Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation

Cristina Roda Sánchez

Supervised by Dr. Víctor Manuel López Jaquero and Dr. Elena María Navarro Martínez

Computing Systems Department
University of Castilla – La Mancha

This dissertation is submitted for the degree of Doctor of Philosophy

Albacete, 2017

This thesis has been funded by the Spanish Ministry of Education, Culture and Sport thanks to the FPU scholarship (FPU12/04962).
To my parents, María Felisa and Luis, and my siblings, Luis and Natalia.

To my grandmas.
The Analytical Engine has no pretensions whatever to originate anything. It can do whatever we know how to order it to perform.

Ada Lovelace

The important thing is not to stop questioning. Curiosity has its own reason for existing.

Albert Einstein

According to Darwin’s Origin of Species, it is not the most intellectual of the species that survives; it is not the strongest that survives; but the species that survives is the one that is able best to adapt and adjust to the changing environment in which it finds itself.

Leon C. Megginson
DECLARATION

This dissertation is the result of my own work and does not include any outcome of work done in collaboration except where it has been specifically indicated in the text. It has not been previously submitted, partial or totally, to any university or institution for any degree, diploma, or other qualification. Moreover, I hereby declare to be one of the main authors of every work used in this thesis by compendium of publications, including the following ones which have been published on journals with impact factor:


Albacete, May 2017

Signed: Cristina Roda Sánchez
SUMMARY

Context-aware systems are becoming more and more popular nowadays as they interact with users in a transparent and ubiquitous way. Healthcare, and namely tele-rehabilitation, is one of the main application domains that takes advantage of this context-awareness paradigm. These systems are able to adapt their behavior depending on the surrounding context (user, platform, environment). However, those adaptations do not always consider possible changes at every level in the system. On one hand, some systems are able to evolve the Software Architecture (SA) dynamically, by modifying the configuration of architectural components at runtime, i.e. adaptation at architectural level. But many times, such systems neglect total or partially the context of use, and hence also any possible change related to the Human-Computer Interaction (HCI) part of the system. On the other hand, there are other systems able to adapt their interaction capabilities, such as the user interface or the interaction modality, to new conditions at runtime, but disregarding any possible adaptation over the software architecture. Therefore, it is obvious that there is a clear need to attempt to align adaptation at both architectural and HCI level. This PhD thesis aims at shedding light on this issue by considering adaptation in those two facets (i.e. SA and HCI) and its alignment while developing context-aware systems for tele-rehabilitation of people.

RESUMEN

Hoy en día, los sistemas sensibles al contexto están siendo cada vez más y más populares gracias a su forma transparente y ubicua de interactuar con los usuarios. El cuidado de la salud, y, concretamente, la tele-rehabilitación, es uno de los principales dominios de aplicación que se beneficia de este paradigma de sensibilidad al contexto. Estos sistemas son capaces de adaptar su comportamiento, dependiendo del contexto que les rodea (usuario, plataforma, entorno). Sin embargo, estas adaptaciones no siempre tienen en cuenta posibles cambios a cualquier nivel del sistema. Por un lado, algunos sistemas son capaces de evolucionar la Arquitectura Software (AS) dinámicamente, modificando la configuración de los componentes arquitectónicos en tiempo de ejecución, es decir, realizan una adaptación a nivel arquitectónico. Pero, muchas veces, dichos sistemas descuidan total o parcialmente el contexto de uso, y, por tanto, también cualquier posible cambio relativo a la parte Interacción Persona-Ordenador (IPO) del sistema. Por otro lado, existen otros sistemas capaces de adaptar sus capacidades de interacción, como la interfaz de usuario o la modalidad de interacción, a las nuevas condiciones en tiempo de ejecución, pero ignoran cualquier posible adaptación sobre la arquitectura software. Por tanto, es obvio que hay una necesidad clara de intentar alinear la adaptación, tanto a nivel arquitectónico, como a nivel IPO. Esta tesis tiene como propósito arrojar algo de luz sobre esta cuestión, considerando la adaptación en estas dos facetas (AS e IPO) y su alineación durante el desarrollo de sistemas sensibles al contexto para la tele-rehabilitación de personas.
ACKNOWLEDGEMENTS

I would like to thank my dear supervisors, Víctor and Elena, for giving me their support from the first time I decided to start this long journey. Thank you, Víctor, for your advices and your expertise, without them this thesis had not been the same. Thank you, Elena, for everything. I have no words to describe how grateful I am to you. You were the first person who trusted on me and encouraged me to get into the fascinating world of researching. I greatly admire and appreciate you both.

I would like to thank all LoUISE research group members to make me feel like another member of the group.

I would like to thank all lecturers and colleagues of the UCLM and other institutions for all the moments we spent together and the experiences exchanged, especially, Kike, Emi, Luis, Gregorio, Julia, Carlos and Javi.

I would like to thank my dear I3A colleagues and friends, namely Miguel, Miguel Ángel, Alex, Jonatan, Félix, Pedro, Susana, Gabi, Juanen, Marina, Estefanía, Celia, Gabriel, Pedro, Julio, Javi and, especially, Elena and Arturo. Thank you, Elena, for all the advices you have brought to me and your immeasurable kindness. Thank you, Arturo, for making me grow as a person and for being my best critic, my confidant and my support during all these years.

I would like to thank my dear Austrian colleagues for making me feel like I was home during my stay in Vienna, namely Uwe, for being my supervisor there, Kostas, Philipp, Gerhard, Christoph, Srdjan, Daniela, Thomas, and especially Edisa. Thank you, Edisa, for your help since the first time I arrived and for all the funny moments we shared.

I would like to thank all my friends that, in one way or another, have contribute to make me the person I am today. Especially my dear high school friends, Marta, Ana, Irina, Elena, María, Jessica and Mercedes. Thank you all for being my support and my way of scape when I needed it. I appreciate you so much.

Finally, I would like to thank all my family for always staying there when I need them. Thank you to my parents for showing me the correct way of life. I cannot express how much I appreciate your effort to make me the person I am today. Thank you to my brother and my sister for your unconditional love, your support and your kindness. I love you all so much.
## ABBREVIATIONS AND ACRONYMS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABI</td>
<td>Acquired Brain Injury</td>
</tr>
<tr>
<td>ADD</td>
<td>Architectural Design Decision</td>
</tr>
<tr>
<td>ADL</td>
<td>Architecture Description Language</td>
</tr>
<tr>
<td>ADR</td>
<td>Architectural Design Rationale</td>
</tr>
<tr>
<td>AI</td>
<td>Artificial Intelligence</td>
</tr>
<tr>
<td>AK</td>
<td>Architectural Knowledge</td>
</tr>
<tr>
<td>AmI</td>
<td>Ambient Intelligence</td>
</tr>
<tr>
<td>FIS</td>
<td>Fuzzy Inference System</td>
</tr>
<tr>
<td>HCI</td>
<td>Human-Computer Interaction</td>
</tr>
<tr>
<td>IM</td>
<td>Interaction Modality</td>
</tr>
<tr>
<td>LD</td>
<td>Linked Data</td>
</tr>
<tr>
<td>MAS</td>
<td>Multi-Agent System</td>
</tr>
<tr>
<td>QoA</td>
<td>Quality of Adaptation</td>
</tr>
<tr>
<td>SA</td>
<td>Software Architecture</td>
</tr>
</tbody>
</table>
CONTENT

Chapter 1. Introduction ............................................................................................................. 1
  1.1 Justification ....................................................................................................................... 1
  1.2 State of the Art and Research Questions ......................................................................... 4
    1.2.1 Context handling in context-aware systems for tele-rehabilitation ........................ 4
    1.2.2 Quality of the adaptation process ............................................................................ 4
    1.2.3 Evolution at architectural level driven by AK ......................................................... 5
    1.2.4 Alignment of adaptation at architectural and HCI level ......................................... 6
    1.2.5 Research questions .................................................................................................... 6
  1.3 Methodology and Work Plan ............................................................................................ 7
  1.4 Structure of the Thesis ..................................................................................................... 9

Chapter 2. Results .................................................................................................................... 11
  2.1 RQ 1 – Alignment of Adaptation at Architectural and HCI Level .............................. 11
  2.2 RQ 2 – Evolution at Architectural Level Driven by AK .......................................... 11
  2.3 RQ 3 – Quality of the Adaptation Process ...................................................................... 12
  2.4 RQ 4 – Context Handling in Context-Aware Systems for Tele-Rehabilitation .......... 12
  2.5 Collaborations .................................................................................................................. 13
  2.6 Summary of Publications ............................................................................................... 13

Chapter 3. Evolution Styles: Using Architectural Knowledge as an Evolution Driver .......... 17

Chapter 4. A Comparative Analysis of Linked Data Tools to Support Architectural Knowledge .............................................................................................................. 43

Chapter 5. Towards Considering Quality of Adaptation in ISATINE .................................. 53

Chapter 6. Towards the Characterization of Interaction Quality ........................................ 57

Chapter 7. Contextualizing Tasks in Tele-Rehabilitation Systems for Older People .......... 61

Chapter 8. Towards an Architecture for a Scalable and Collaborative AmI Environment .... 77

Chapter 9. An Interactive Fuzzy Inference System for Teletherapy of Older People ........ 91

Chapter 10. A Multi-Agent System for Acquired Brain Injury Rehabilitation in Ambient Intelligence Environments ................................................................. 111


Chapter 13. Variability as a Driver for the Adaptation of Tele-Rehabilitation Context-Aware Systems .................................................................................................................. 229

Chapter 14. Discussion and Future Work ............................................................................. 261
  14.1 Discussion and Conclusions ......................................................................................... 261
  14.2 Ongoing and Future Work ......................................................................................... 264

References ............................................................................................................................... 267
Chapter 1
Introduction

This chapter presents the different issues addressed in this thesis. Section 1.1 presents the rationale of this dissertation for solving the different problems encountered. Section 1.2 presents a brief description of the state of the art, as well as the research questions proposed in this thesis. Section 1.3 presents the methodology and work plan followed to conduct this thesis. Finally, Section 1.4 presents the structure of the thesis.

1.1 Justification

Recently, a growing tendency has emerged which is transforming the environment where the user interacts into a system smarter than ever before. These smart environments could be the user’s home, an office or even a hospital where the system interacts with the user in a transparent and ubiquitous manner. These kinds of systems are called context-aware systems since they are aware of the surrounding context of use, usually by means of sensors in charge of gathering a great variety of data from the user, the platform he/she is interacting with, and the environment where such interaction is taking place. Context-aware systems [35] exploit the context to increase the fit of a service to the user’s needs beyond the user centred evaluation. Due to this, a new paradigm has risen up, Ambient Intelligence (AmI), with the aim of developing those pervasive systems sensitive to context that can adapt their behavior to the new conditions at runtime. As Ducatel et al. state [14], AmI promotes the development of intelligent systems “embedded in an environment that is capable of recognizing and responding to the presence of different individuals in a seamless, unobtrusive and often invisible way”.

Thus, AmI systems become transparent as people do not perceive their complexity neither their presence, and are intelligent to react in a proactive and sensitive way [2] at the same time. These two characteristics have had a great impact because it has made technology usable for people who, otherwise, would have been probably computer illiterate. In this sense, older people and people with some kind of impairment, such as those who suffer Acquired Brain Injury (ABI), can take
advantage of this smart approach. Namely, tele-rehabilitation for these two groups constitutes one of the main application domains for exploiting both AmI and context-aware systems due to its high level of variability. Tele-rehabilitation can be defined as [5] “the application of telecommunication, remote sensing and operation technologies, and computing technologies to assist with the provision of medical rehabilitation services at a distance”. By developing tele-rehabilitation systems that can adapt their behavior based on the surrounding context, we will be able to offer bespoke therapies for improving health, and hence the quality of life of those vulnerable people.

Given the great variability these people show, the system must be adapted according to the surrounding content of use. However, such adaptation should provide a certain degree of quality (Quality of Adaptation, QoA), as well as a user interface or a software product in general should do with the aim of producing a pleasant adaptation experience. It means that the adaptation framework should offer some type of mechanism for evaluating the adaptation after its application in order to know whether it has been appropriate or not. The most widely-accepted adaptation process is the one proposed by Dieterich et al. [13], but it suffers from a relevant shortcoming, as QoA is not tackled at all, since it is only focused in the execution of the adaptation. All these statements constitute one of the main foundation of this thesis: facilitating adaptation tasks at HCI level through an appropriate context management, focusing on tele-rehabilitation of older people and people with ABI, and providing a certain degree of QoA.

It is worth noting that software development has been dealing with many challenges since its inception, such as system complexity, non-functional qualities, maintenance operations, distributed production, frequent personnel changes, etc. [32]. Moreover, software companies with high maintenance costs are increasingly demanding flexible and easy-to-maintain designs [32]. In this context, Software Architecture (SA) has become a valuable asset that enables software companies to achieve a great variety of goals by representing and communicating the system structure and behavior to all stakeholders of the system [65]. Until recently, SA research has been focused on Architecture Description Languages (ADLs) in order to describe SA elements and form following the model of SA proposed by Perry and Wolf [39] in 1992. These authors described the SA of a system as a set of architectural (or design) elements that have a particular form:

\[
\text{Software Architecture} = \{\text{Elements, Form, Rationale}\}
\]

The architectural elements can be processing elements, data elements, or connecting elements. Processing elements are components that supply the transformation of data elements that contain the information used and transformed. Finally, connecting elements, such as procedure calls or messages, constitute the glue that holds the different pieces of the architecture together. Moreover, the architectural form consists of weighted properties and relationships, indicating its importance or the necessity of selecting among alternatives, some of which may be preferred over others. Properties define constraints on the architectural elements according to the architect’s decisions, while relationships limit how such elements may interact and how they are organized with respect to each other in the architecture [39].

However, until recently, not enough attention has been paid to the third element of Perry and Wolf’s model: the rationale. The rationale [39] captures the motivation for choosing an architectural style, elements and form, i.e. it explains the satisfaction of the system constraints, determined by considerations ranging from basic functional aspects to non-functional ones, such as economics, performance or reliability. Nowadays, the importance of Architectural Design Decisions (ADDs) and their Architectural Design Rationale (ADR) is widely recognized and they have become essential aspects in Architectural Knowledge (AK). In addition, the modelling, managing, and
sharing of AK has also become a significant research focus [65], being key for a proper exploitation and management of the SA.

Regardless of the architecture chosen, extra challenges arise during software development due to changes in the requirements, the technology, etc. As Bersoff et al. already stated in the eighties [5], “no matter where you are in the system life cycle, the system will change, and the desire to change it will persist throughout the life cycle”. Thus, software evolution emerges here as an essential feature of every developed system [8][37] to adapt it to market trends, technological advances, or simply new customers’ requirements or needs. In this sense, when software evolution is carried out, SA emerges in a natural way as one of the cornerstones that should be considered from two different points of view: as an artifact for the evolution, as it helps the architect plan and restructure the system, i.e. it helps in the task of changing the system; and as an artifact of the evolution, because SA has to be changed as well.

Usually, as aforementioned, the evolution of the SA is applied when new requirements arise during the system life cycle, so that the SA is modified in order to meet them, and those changes remain stable over time. However, sometimes, as Garlan et al. state [18], organization of components and connectors of a SA may change during execution. In this case, the SA should be evolved dynamically, by modifying such configuration of architectural components at runtime, i.e. adaptation at architectural level. Therefore, every software system should provide certain mechanisms for facilitating the SA evolution over time, and also at runtime. Here AK can be seen as such ideal evolution driver. The architecture rationale and design decisions (i.e. the AK) are critical in evolving software systems, so they should be captured in some useful form to aid the evolution process. This constitutes another main foundation of this thesis: facilitating evolution tasks at architectural level with AK as the main driver, and represented in an adequate manner.

Therefore, there is no doubt about the importance of SA for a developed system, so that AmI and context-aware systems should also pay attention to their SA and the way it is designed, always reflecting and taking into account the context of use. By doing so, context-aware systems will provide more robustness and completeness to the adaptation process, since they will consider both architectural and interaction aspects. Nevertheless, there is no much evidence in the literature about SA proposals that deal with dynamic evolution according to the context of use. Most of these architectural proposals are focused on evolving the system from the point of view of SA, and the possible configurations of its components. Consequently, such architectural approaches do not treat context-awareness in an explicit way during the adaptation process, neglecting the context in some way, and hence also the adaptation from the point of view of HCI. On the other hand, there are also HCI approaches centered on context-aware adaptations, which neglect in some way the SA, and hence also any possible change of the SA of the system during the adaptation process.

Therefore, there is a clear need to provide an overview about the different approaches used in context-aware systems in order to align the adaptation at architectural and HCI level. Moreover, another need is to identify those gaps encountered up to now with regard to the deficiencies and weaknesses of systems when dealing with context of use and its different dimensions (user, platform and environment). These findings will help the research community by providing guidelines for modelling and managing context, and its different dimensions, as well as for knowing how the SA should consider such context in order to provide a good adaptation experience to the final user. This constitutes the final foundation of this thesis: aligning adaptation at both architectural and HCI level.
1.2 State of the Art and Research Questions

This section presents an overview of the state of the art from the different points of view that have been considered throughout the development of this thesis, as well as the research questions proposed.

1.2.1 Context handling in context-aware systems for tele-rehabilitation

As part of the foundation of this thesis related to facilitate adaptation tasks at HCI level, there is an issue to be considered: the appropriate handling of context. As aforementioned, the context management in the domain of tele-rehabilitation of older people and people with ABI is crucial due to its high variability. It is obvious that both context and its correct handling constitute a cornerstone of context-aware systems whose behavior is adapted to the surrounding conditions. Ambient Intelligence (AmI) paradigm has emerged with the aim of developing those systems sensitive to context. AmI technologies are expected to be sensitive, responsive, adaptive, transparent, ubiquitous and intelligent [10]. Therefore, tele-rehabilitation systems for older people and people affected with ABI are excellent candidates to take advantage of such AmI approach.

A natural relationship between AmI and MAS arises in this context, since agents provide an effective way to develop AmI systems as they are reactive, proactive and exhibit an intelligent and autonomous behavior [3]. Furthermore, apart from MAS, Artificial Intelligence (AI) mechanisms, such as Neural Networks or Fuzzy Inference Systems (FISs), can also be embedded in AmI environments to make them more intelligent, adaptable, energy efficient and suitable to the user’s needs, as reflected in [20][21]. Some works illustrate the perfect integration of all these approaches together, i.e. AmI and AI with MASs, in healthcare domain, such as the work presented in [61].

Namely, in tele-rehabilitation, the power provided by FISs can be exploited to facilitate the need for bespoke therapies. These fuzzy systems have a reasoning mechanism that performs inference operations using fuzzy rules to make decisions under uncertain conditions. In the literature, there are some works that use FISs for physical rehabilitation, such as [4][24][76]. However, they do not allow therapists to select the variables to be used by the FIS and to know how they should be described in order to design bespoke therapies. As far as we know, the only system that offers some degree of flexibility in defining FISs is the one proposed in [62]. Nevertheless, this system is specifically designed for differential diagnosis and therefore cannot be applied to tele-rehabilitation. In short, to the best of our knowledge, there is only a proposal presented in [8] about physical rehabilitation which exploits AmI and FISs. It proposes an intelligent component to manage the robot behavior, but it is not focused on the performance of the tasks prescribed in a particular rehabilitation therapy. Furthermore, it does not take advantage of using a MAS, combined with AmI, able to manage context information specified during the design.

From the foregoing, two main conclusions can be drawn. Firstly, the existing systems that exploit fuzzy rules for rehabilitation do not offer therapists with facilities to describe FISs necessary for the design of bespoke therapies. Secondly, not many works are focused on providing a MAS, combined with AmI and FISs, able to control the performance of all the tasks that a patient is doing during a rehabilitation therapy, and able to manage context information specified during the design process by using a context metamodel to facilitate adaptation tasks at runtime.

1.2.2 Quality of the adaptation process

As previously established, providing the adaptation process with quality (i.e. QoA) is another issue to be tackled in order to facilitate adaptation tasks at HCI level. Adaptation at HCI level arises with
the aim of constructing software systems and user interfaces suitable for any type of user, any type of platform, and able to work under any situation. For example, when the screen brightness is automatically adapted to current lightning conditions, or when the user interface is rearranged when we rotate the device. For the purpose of guiding designers in the definition of such adaptation capabilities based on the context, different adaptation frameworks/processes have been proposed, such as [9][38]. Nevertheless, the most widely-accepted adaptation process is the one proposed by Dieterich et al. [13], but it is only focused on the adaptation execution. Therefore, it does not assess the adaptation in order to know if it has been appropriate or not after its execution. Thus, QoA is often ignored by the vast majority of adaptation frameworks.

1.2.3 Evolution at architectural level driven by AK

There are compelling arguments [6] for the exploitation of the rationale while the SA is being modified. However, the use of the rationale, most of the time, is simply as a documentation artifact. Some works highlight the importance of dealing with SA evolution by using AK, such as [27][65][66]. However, to the best of our knowledge, only two approaches [67][68] actively exploit the AK for such aim. The former enables the architects to predict the impact of change before it happens, and the latter is oriented to validate the evolution process. Nevertheless, none of them are explicitly focused on guiding the change process. In this sense, there is an obvious relationship between software evolution and architectural styles for guiding the evolution process, although it has not always been made explicit. Several authors [17][64] have tried to exploit this intuition, developing a new concept: evolution style. In short, where the original architectural style describes the constraints to be fulfilled by a set of systems, in an evolution style, this set of systems would be the set of potential configurations of an evolutionary system, i.e. the style defines the constraints to be fulfilled by any possible evolution of the system. However, in none of these works, the evolution decisions are influenced by the existing AK.

Furthermore, as aforementioned, this SA evolution is often applied when new requirements arise, during the system life cycle, so that the SA is modified in order to meet them. Those system changes remain stable over time, but sometimes, as Garlan et al. state [18], organization of components and connectors of a SA may change during execution. In this case, the SA should be evolved dynamically, by modifying such configuration of architectural components at runtime, i.e. adaptation at architectural level. Some works in the literature deal with this SA dynamic evolution by proposing an architecture-based self-adaptive software. For example, the work presented in [19] proposes the Knowledge-Based Architectural Adaptation Management approach for the dynamic management and reasoning over adaptation policies and relevant system knowledge in order to drive the SA adaptation process. Nevertheless, this adaptation process does not consider the AK at all, which constitutes a general tendency in the literature. Thus, in none of all these previous works, the SA evolution is influenced by the existing AK, so that the AK do not participate as an active driver for guiding the evolution process. For facilitating such use, the system should include an explicit and appropriate representation of the AK.

One of the key issues that arises whenever AK is introduced as an active participant in the development and evolution of a software system is the selection of the best supporting tool to describe it. In this context, the proposals can be classified into two main categories: i) those that evaluate AK management tools in the context of software development processes, such as [22][65]; and ii) those that evaluate the usefulness of documenting AK, such as [6][15][16][28]. However, none of them is focused on assessing the usability effectiveness of a given visualization technique for AK representation, except the work presented in [63], which is the only one that deals with visualization techniques for representing AK. Nevertheless, this last work focuses on presenting
and classifying which visualization techniques are currently supported by different tools, but they are not evaluated at all to provide a practical guideline that reflects its usability in terms of AK manipulation. Therefore, AK can be a suitable evolution driver, but it should be represented in an adequate manner.

1.2.4 Alignment of adaptation at architectural and HCI level

After reviewing the literature, we have realized that adaptation is addressed from different points of view. On one hand, most of the SA proposals base the evolution on reconfiguring architectural elements, neglecting the context of use in some way, and hence also the adaptation from the point of view of HCI. On the other hand, many HCI works are focused on context-aware adaptations, but overlook partial or totally the software architecture of the system, and hence also any possible change on it. In order to align adaptation at both architectural and HCI level, an analysis of the literature has been performed to find which research works highlight the use of context, and its relationship with the software architecture in existing context-aware systems. Chapter 11 presents a systematic mapping study that has collected such related works about SAs for context-aware systems. This mapping study has served to complete the state of the art of this dissertation.

1.2.5 Research questions

After taking a glance at the state of the art, some deficiencies and limitations have been encountered, namely:

- The existing systems that exploit fuzzy rules for rehabilitation do not offer therapists with facilities to describe FISs necessary for the design of bespoke therapies.
- Not many works are focused on providing a MAS, combined with AmI and FISs, able to control the performance of all the rehabilitation tasks that a patient is doing during a therapy, and able to manage context information specified in the design process to facilitate the adaptation at runtime.
- Most of the adaptation processes are focused on the adaptation execution and they dismiss the assessment of the adaptation in order to know if it has been appropriate or not after its execution.
- There is no much evidence in the literature about proposals explicitly focused on AK as an active driver for guiding the evolution process.
- Many works are focused on presenting current visualization techniques supported by different tools in order to classify them, but they are not evaluated to provide a practical guideline that reflects its usability in terms of AK manipulation.
- There is no much evidence in the literature about SA proposals that treat evolution dynamically, based on the context of use.
- There are HCI studies focused on context-aware adaptations which neglect partial or totally the SA, and also any possible change on it during the adaptation process.

Considering the previous shortcomings and needs, the Research Questions (RQs) of this thesis have been established as follows:

- RQ 1: What is the overview of SAs for context-aware systems to align adaptation at both architectural and HCI level?
- RQ 2: Can the AK be considered as a driver for software evolution? And for dynamic evolution? If so, how can it be visualized to facilitate evolution tasks at architectural level?
- RQ 3: How an adaptation process can be provided with a certain degree of quality to facilitate adaptation tasks at HCI level?
• RQ 4: How the context of use in a high variability application domain as tele-rehabilitation can be dealt with to facilitate adaptation tasks at HCI level?

The main purpose of this thesis is to provide best practices and useful guidelines for assisting designers, stakeholders and practitioners in the task of adapting a system at every level, i.e. at both architectural and HCI level. Those adaptation tasks should be based on the surrounding context of use, but it is often neglected or even forgotten. Therefore, the main challenge during the process of answering these RQs has been to deal with evolution and adaptation issues in two different research fields, such as Software Architecture and HCI. Each one of them has its own point of view when changing the system, and architectural and HCI changes are usually tackled as separated concerns. Hence, the alignment of adaptation at both architectural and HCI level in the same software system is often missed.

1.3 Methodology and Work Plan

The design science research methodology [23] has been followed to perform this dissertation. The seven research guidelines of this methodology have been applied as follows:

1. *Design as an artifact.* This research has produced several artifacts: guidelines to select the most suitable visualization techniques and Linked Data (LD) tools to support AK; guidelines to provide quality to an adaptation process; models, metamodels and a MAS to manage properly the context of use in tele-rehabilitation systems, as well as its high variability; an editor for creating bespoke therapies; and artifacts to align the adaptation at architectural and HCI level, such as a glossary of useful terms related to context-aware systems, the most frequently SA patterns and styles used in context-aware systems, as well as guidelines for architectures dealing with context.

2. *Problem relevance.* The problems this thesis addresses are related to domains that are becoming more and more popular, ensuring the problem relevance. These domains are context-aware systems, tele-rehabilitation, older people and people with ABI.

3. *Design evaluation.* The research presented in this thesis has been evaluated through well-defined and widely extended empirical strategies proposed by Wohlin et al. [73], such as experiments, surveys and case studies in order to assess the different artifacts developed.

4. *Research contributions.* As stated in the first point, several artifacts have been developed and they constitute the main research contributions of this thesis with the aim of providing adaptation facilities at architectural and HCI level.

5. *Research rigor.* During the research process, different statistical techniques [34], Software Engineering methods [29][30][72][73], international standards [25][26], and widely used guidelines and works to perform systematic studies [7][31][33][40][69][70][71][74][75] have been followed to provide rigor to this thesis.

6. *Design as a search.* This thesis follows an iterative process with the main purpose of improving the adaptation process in context-aware systems, taking into account both architectural and interaction aspects. Thus, the different results have been assessed and improved in an iterative manner to achieve the previous objective.

7. *Communication of research.* The results of this research have been presented to experienced audience regarding technology, and publications about the research artifacts have been published in relevant journals and venues related to the topic of this thesis.

8. The work plan of this thesis is shown in

. The different activities performed by the candidate of this thesis along with the collaboration with the LoUISE research group are explained below:
• **State of the art.** This activity was carried out over the course of the whole thesis in order to collect research works related to context modelling and AmI systems, quality of adaptation, AK and software evolution. Furthermore, the state of the art was completed thanks to the mapping study presented in Chapter 11 about SAs for context-aware systems.

• **RQ 2 – Evolution at architectural level driven by AK.** At the first stage of this thesis, AK was identified as a suitable evolution driver. Moreover, several approaches related to the specification and visualization of AK were analyzed. As a result of this study, an empirical evaluation was performed to determine which proposal was the best alternative to represent AK. Different Linked Data tools were also analyzed aiming at supporting the description of the AK. These initial works constitute the basis for facilitating evolution tasks at architectural level.

• **RQ 3 – Quality of the adaptation process.** It was noticed that every adaptation process should provide a certain degree of quality at every stage. In this sense, a quality metamodel was defined for characterizing QoA at every stage of an adaptation process. This quality metamodel serves as a basis for evaluating the adaptation which will avoid the rejection of the adaptation by the user. Therefore, these activities constitute the basis for facilitating adaptation tasks at HCI level.

• **RQ 4 – Context handling in context-aware systems for tele-rehabilitation.** Mainly during the last two years of this thesis, a context metamodel was defined in order to guide the adaptation of the system in different dimensions (user, platform and environment). Furthermore, such metamodel was applied to different case studies related to tele-rehabilitation, namely physical rehabilitation. This context metamodel was also integrated into a FIS for providing adaptation capabilities in tele-rehabilitation systems. A MAS was defined as well to support such context treatment for adaptation in tele-rehabilitation systems. Finally, a user model, an interaction model and a feature model for modelling and controlling dynamic variability were also proposed, focused on people with ABI. All these results also constitute the basis for facilitating adaptation tasks at HCI level.

• **RQ 1 – Alignment of adaptation at architectural and HCI level.** After analyzing evolution and adaptation at architectural and HCI level as a separation of concerns, it was noticed that there was a need to align both adaptation aspects to provide better adaptation experiences to the final user. Therefore, a systematic mapping study was conducted to find the different architectural patterns and frameworks used in context-aware systems, as well as to know how context was included in those architectures. The maturity of these architectural proposals and the maturity of the adaptations were also studied. These results constitute the basis for facilitating adaptation tasks at both architectural and HCI level, and for future work to cover the different gaps encountered in SAs for context-aware systems.

• **Support.** A comparative analysis of LD tools was performed and several prototypes were developed, derived from a research project (Enriching UsiXML for the Development of Post-WIMP User Interfaces, TIN2012-34003) and collaborations, in particular, a therapy editor for Microsoft PixelSense. In addition, all models and metamodels presented in this thesis were specified by using Microsoft Visual Studio in order to facilitate implementation tasks. The feature model proposed to model context variability in tele-rehabilitation domain was also developed by using an Eclipse plugin, called FeatureIDE. This framework has allowed us to specify and validate different feature configurations.

• **Validation.** Thanks to the participation of the candidate in a research project (TIN2012-34003), the different proposals of this thesis have been applied to the tele-rehabilitation of older people and people with ABI. Furthermore, the case studies proposed to validate the results have been validated by experts in those domains [1][11].
• **Diffusion.** During the entire process of this thesis, several works have been published or submitted to national and international venues, both conferences and journals, for reaching a wide dissemination among the research community.

• **Stay.** At the beginning of the 4th year of this process, the candidate performed a research stay at the University of Vienna, namely at the Faculty of Computer Science, with Dr. Uwe Zdun, a full professor and head of the research group Software Architecture. During this stay, different tasks were carried out with the aim of completing the state of the art of this thesis in a rigorous, formal and systematic way. As a result, a systematic mapping study has been performed in order to discover relevant works of SAs for context-aware systems (Chapter 11).

![Table 1. Work plan of the thesis](image)

<table>
<thead>
<tr>
<th></th>
<th>MSc</th>
<th>1st year</th>
<th>2nd year</th>
<th>3rd year</th>
<th>4th year</th>
</tr>
</thead>
<tbody>
<tr>
<td>State of the art</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RQ 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RQ 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RQ 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RQ 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Support</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Validation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Diffusion</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stay</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 1.4 Structure of the Thesis

This introduction has presented the justification and main foundations of this thesis (Section 1.1), the state of the art and research questions proposed (Section 1.2), as well as the methodology and work plan followed (Section 1.3). Next, Chapter 2 shows the results obtained with regard to every research question (Sections 2.1 to 2.4), and the different collaborations in research projects (Section 2.5), as well as a summary of the publications that take part of this thesis (Section 2.6).

Then, Chapter 3 to Chapter 13 present in detail the most relevant works with their entire content, both articles that have been already published, as well as others submitted or to be submitted in the near future. Finally, Chapter 14 concludes this dissertation with a discussion and the main conclusions, as well as the ongoing and future work derived from it.
Chapter 2

Results

Throughout this chapter, the results obtained regarding each research question previously described are presented in Sections 2.1, 2.2, 2.3 and 2.4, respectively. Then, Section 2.5 presents the articles published thanks to collaborations in a research project (TIN2012-34003). Finally, a summary of all the articles published, submitted or to be submitted is depicted in Section 2.6.

Notice that the three publications [12][58][53] that endorse this dissertation, along with the other relevant articles presented in detail from Chapter 3 to Chapter 13, are related to the four research questions previously defined in Section 1.2.

2.1 RQ 1 – Alignment of Adaptation at Architectural and HCI Level

As aforementioned, there was a need to provide an overview about the different approaches used in context-aware systems in order to align the adaptation at architectural level and at HCI level. This overview about SAs for context-aware systems was presented in [55] (Chapter 11). In addition, some gaps were detected regarding the deficiencies and weaknesses of systems when dealing with context and its different dimensions (user, platform, environment). These findings will help the research community by providing guidelines for modelling and managing context and its different dimensions, as well as for knowing how the SA should consider such context in order to provide a good adaptation experience to the final user.

2.2 RQ 2 – Evolution at Architectural Level Driven by AK

The very first work done in this thesis was focused on how to provide a proper description of the SA in order to facilitate the software evolution process. In this sense, the idea presented in [37] was centered on describing the SA by means of the automatic application of architectural styles, following a model-driven approach. Then, the effort went into considering the AK as a valuable
asset of the evolution process, and how it could be seen as a useful evolution driver provided that it was represented in a useful form [12] (Chapter 3).

An empirical evaluation [48][49][50] (Chapter 12) was carried out to discover which visualization technique used for representing AK was the most adequate one. Furthermore, a comparative analysis of different Linked Data (LD) tools [46][47] (Chapter 4) that support the representation and management of AK was also performed. By following these guidelines offered by these evaluations, the description of the AK will be enhanced to facilitate evolution tasks at architectural level.

2.3 RQ 3 – Quality of the Adaptation Process

As aforementioned, another objective of this thesis was to provide the adaptation process with high quality levels. With this aim, some works [42][44] (Chapter 5) were focused on offering proposals for achieving such Quality of Adaptation (QoA). The concept QoA was clarified to better understand what this term means when applied to the several stages of an adaptation process. By doing so, it can be proposed what techniques or metrics best fit each adaptation process stage, contributing to a better understanding of what QoA is and how it can be actually used in an adaptation process.

Other works [43][45][41] (Chapter 6) were focused on characterizing the interaction quality by means of a survey. We identified the different terms that both persons either related or not to the interaction field use to describe quality. This survey showed that a gap exists among the views that both groups use to refer to interaction quality. This gap should be considered to understand the quality and then to offer meaningful information about QoA, both at every stage of an adaptation process and during the communication with the user.

2.4 RQ 4 – Context Handling in Context-Aware Systems for Tele-Rehabilitation

As aforesaid, another objective this thesis has striven for is facilitating adaptation tasks at HCI level through an appropriate context handling in order to provide older people and people with ABI with tele-rehabilitation facilities. For this purpose, some metamodels were proposed for modelling different aspects useful in tele-rehabilitation environments. These aspects were related to the context of use (user, devices/sensors and environment), and the interaction between the user (patient) and the different devices, as well as specific features of rehabilitation therapies (activities, steps, gestures, postures, etc.). A Fuzzy Inference System (FIS) metamodel was also defined to support the creation of fuzzy rules suitable for adapting the patient’s therapy to the context of use. Additionally, the architecture of a Multi-Agent System (MAS) was specified for guiding such adaptation process in tele-rehabilitation systems, always considering the context of use. Next, these works are further detailed.

The work presented in [57] proposes a therapy metamodel which takes into consideration some context information about the patient, the rehabilitation activities he/she has to perform, or even his/her gestures or postures during the rehabilitation process. This therapy metamodel was designed to guide the adaptation in tele-rehabilitation systems at HCI level. The work in [58] (Chapter 9) was one step forward and, besides a therapy metamodel, it also proposed a FIS metamodel for creating FISs and fuzzy rules for controlling transitions between rehabilitation activities and adapting therapies to the context of use.
After a brief review of context models presented in [60], the work in [56] (Chapter 7) proposed a new context metamodel for modelling the context of tele-rehabilitation systems, as well as a task metamodel. It also described how to contextualize those tasks involved in a tele-rehabilitation process to offer more efficient adaptation mechanisms. The architecture of a MAS was presented in [52][54] (Chapter 8) for guiding the adaptation process of tele-rehabilitation systems, considering the context of use. The approach presented in [53] (Chapter 10) enhanced and extended the previous ones since it presented a more detailed design of the MAS, and an improved version of the context metamodel. Note that most of these works included a case study to validate the proposals.

Finally, the work presented in [51] (Chapter 13) focused on modelling the user of the tele-rehabilitation system to provide a separation of concerns regarding the other dimensions of context (platform and environment), given its relevance for the adaptation process. Therefore, a user model was proposed for modelling the body parts, as well as the different movements every part may perform for facilitating the adaptation in physical rehabilitation activities. Some other features related to ABI were also modelled in case the user (patient) of the system presented any impairment, either physical, cognitive or emotional. Apart from this user model, we modelled the input and output channels the user could potentially use, given her capabilities and skills, to interact with the different available devices, and vice versa. Thus, an interaction model was proposed that enables the specification of both the Interaction Modalities (IMs) the user and the likely devices that may be part of the system.

The work presented in [51] (Chapter 13) also illustrated a feature model for creating bespoke activities for people with ABI. This proposal aimed at modelling and controlling the high context variability usual in ABI application domains. Thus, according to the characteristics of the patient, the system will offer a configuration of features, including the most appropriate association activity for treating cognitive deficiencies in people with ABI. All these models proposed will facilitate adaptation at HCI level by following a model-based approach.

2.5 Collaborations

During the development of this thesis, the candidate was also collaborating in a research project (TIN2012-34003) about the construction of post-WIMP user interfaces for tele-rehabilitation. All the collaboration work was related to the fourth research question. Therefore, it has contributed to answer such question about context management in tele-rehabilitation systems, focusing on older people and people with ABI. Therefore, the outcomes of these collaborations have provided clear alternatives to facilitate adaptation tasks at HCI level.

Due to the fact that all this collaboration work has been already presented in Section 2.4 with regard to the fourth research question, in this section they are only cited in order to let the reader know which specific publications were carried out as result of this collaboration. Namely, these works were: [52], [53], [54], [56], [57], [58], [59] and [60].

2.6 Summary of Publications

Table 2 shows the list of articles published, submitted or to be submitted as result of this thesis. Notice that the three publications that endorse this dissertation by compendium of publications are highlighted by an asterisk before the title. Along with each publication, the research questions it is related to are also represented. Lastly, the last column shows the chapter where the work is included in this document. Note that only papers found more relevant to describe the achievements of this
thesis are detailed in a separated chapter of this dissertation (Chapter 3 to Chapter 13). Publications related to collaborations are denoted with a C in the RQ column.

Table 2. Publications during this thesis

<table>
<thead>
<tr>
<th>Publication</th>
<th>RQ</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Journals</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>* Arturo C. Rodríguez, Cristina Roda, Francisco Montero, Pascual González, Elena Navarro, An Interactive Fuzzy Inference System for Teletherapy of Older People [58] (Cognitive Computation, Q2)</td>
<td>4, C</td>
<td>9</td>
</tr>
<tr>
<td>* Cristina Roda, Arturo C. Rodríguez, Víctor López-Jaquero, Elena Navarro, Pascual González, A Multi-Agent System for Acquired Brain Injury Rehabilitation in Ambient Intelligence Environments [53] (Neurocomputing, Q1)</td>
<td>4, C</td>
<td>10</td>
</tr>
<tr>
<td>Cristina Roda, Víctor López-Jaquero, Francisco Montero, ¿Qué Entendemos por Calidad de Interacción? [41] (Revista Novática)</td>
<td>3</td>
<td>-</td>
</tr>
<tr>
<td>Cristina Roda, Elena Navarro, Víctor López-Jaquero, Variability as a Driver for the Adaptation of Tele-Rehabilitation Context-Aware Systems [51] (to be submitted, KBS, Q1)</td>
<td>4</td>
<td>13</td>
</tr>
<tr>
<td><strong>International Conferences</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elena Navarro, Carlos E. Cuesta, Dewayne E. Perry, Cristina Roda, Using Model Transformation Techniques for the Superimposition of Architectural Styles [37] (ECSA, CORE A)</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>Arturo C. Rodríguez, Cristina Roda, Francisco Montero, Pascual González, Elena Navarro, A Collaborative System for Designing Tele-Therapies [57] (IWAAL)</td>
<td>4, C</td>
<td>-</td>
</tr>
<tr>
<td>Cristina Roda, Elena Navarro, Carlos E. Cuesta, A Comparative Analysis of Linked Data Tools to Support Architectural Knowledge [46] (ISD, CORE A)</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Cristina Roda, Víctor López-Jaquero, Francisco Montero, Towards Considering Quality of Adaptation in ISATINE [44] (Interacción)</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Title</td>
<td>Authors</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Avanzando Hacia la Consideración de la Calidad de Adaptación en ISATINE (Interacción)</td>
<td>Cristina Roda, Víctor López-Jaquero, Francisco Montero</td>
<td>3</td>
</tr>
<tr>
<td>A Multi-Agent System in Ambient Intelligence for the Physical Rehabilitation of Older People</td>
<td>Cristina Roda, Arturo C. Rodríguez, Víctor López-Jaquero, Pascual González, Elena Navarro</td>
<td>4, C</td>
</tr>
<tr>
<td>Towards the Characterization of Interaction Quality (Interacción)</td>
<td>Cristina Roda, Víctor López-Jaquero, Francisco Montero</td>
<td>3</td>
</tr>
<tr>
<td>Hacia la Caracterización de la Calidad de Interacción (Interacción)</td>
<td>Cristina Roda, Víctor López-Jaquero, Francisco Montero</td>
<td>3</td>
</tr>
<tr>
<td>Contextualizing Tasks in Tele-Rehabilitation Systems for Older People</td>
<td>Arturo C. Rodríguez, Cristina Roda, Pascual González, Elena Navarro</td>
<td>4, C</td>
</tr>
<tr>
<td>Towards an Architecture for a Scalable and Collaborative AmI Environment</td>
<td>Cristina Roda, Arturo C. Rodríguez, Elena Navarro, Víctor López-Jaquero, Pascual González</td>
<td>4, C</td>
</tr>
<tr>
<td>Técnicas de Visualización para Conocimiento Arquitectónico: una Evaluación Empírica</td>
<td>Cristina Roda, Elena Navarro, Carlos E. Cuesta, Dewayne E. Perry</td>
<td>2</td>
</tr>
<tr>
<td>Modelos de Contexto en el Desarrollo de Interfaces Post-WIMP: una Revisión Crítica</td>
<td>Arturo C. Rodríguez, Cristina Roda, Elena Navarro, Pascual González</td>
<td>4, C</td>
</tr>
<tr>
<td>An Interactive Fuzzy Inference System for Teletherapy of Older People</td>
<td>Arturo C. Rodríguez, Cristina Roda, Francisco Montero, Pascual González, Elena Navarro</td>
<td>4, C</td>
</tr>
<tr>
<td>Evaluating Visualization Techniques for Architectural Knowledge: Experimental Material</td>
<td>Cristina Roda, Elena Navarro, Carlos E. Cuesta, Dewayne E. Perry, Javier Jaén</td>
<td>2</td>
</tr>
<tr>
<td>Analyzing Linked Data Tools for SHARK</td>
<td>Cristina Roda, Elena Navarro, Carlos E. Cuesta</td>
<td>2</td>
</tr>
</tbody>
</table>
Chapter 3

Evolution Styles: Using Architectural Knowledge as an Evolution Driver


**Type of venue:** International Journal (IF: 1.320, Q2)

*This publication endorses this dissertation by compendium of publications.*
Evolution styles: using architectural knowledge as an evolution driver

Carlos E. Cuesta¹, Elena Navarro²*,†, Dewayne E. Perry³ and Cristina Roda⁴

¹Department of Computing Languages and Systems II, ETS Ingeniería Informática, Campus de Móstoles, Rey Juan Carlos University, 28934, Madrid, Spain
²Computing Systems Department, University of Castilla-La Mancha, Avda. España s/n, 02071, Albacete, Spain
³Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX USA
⁴Vector Corp., Parque Científico y Tecnológico de Albacete, Paseo de la Innovación 3, Centro de Empresas de I+D, 02006, Albacete, Spain

ABSTRACT

Software evolution is an increasingly challenging and compelling concern for every developed software system because of the changes in the requirements, the technology, etc. When software evolution is carried out, software architecture emerges as one of the cornerstones that should be considered from two different points of view: as an artifact for the evolution, as it helps the architect plan and restructure the system, and as an artifact of the evolution, because it has to be modified as well. This paper focuses on the second point of view—that is, on the evolution of the software architecture, but taking into account architectural knowledge as a key driver of the process. Given that architecture rationale and design intent are critical in evolving software systems, it is imperative that they be captured in some useful form to aid that evolution process. We present a new approach for evolution styles that extends them by considering in their description the architectural knowledge as a valuable asset of the evolution process. Copyright © 2012 John Wiley & Sons, Ltd.

Received 29 July 2011; Revised 26 September 2012; Accepted 10 October 2012

KEY WORDS: software architecture evolution; architectural knowledge; evolution styles

1. INTRODUCTION

Software evolution is an essential feature of every developed system [1, 2]. There are always compelling arguments that lead us to change the developed system to adapt it to market trends, technological advances, or, simply, new customers’ requirements. This feature was already pointed out in the eighties when the first law of software engineering was stated by Bersoff et al. [3]: ‘No matter where you are in the system life cycle, the system will change, and the desire to change it will persist throughout the life cycle’; even previously, Lehman had stated the first law of software evolution [4, 5], a work that would be later extended and developed several times [4, 6, 7]. Therefore, as the need for change will last over the whole life cycle of the systems, the introduction of proper processes, techniques, and tools that help us deal with it becomes an essential part of the software development and maintenance process.

When we tackle software evolution, software architecture (SA) emerges as one of the cornerstones that should be considered from two different points of view: as an artifact for the evolution and as an artifact of the evolution. That is, SA is an artifact that can be used for evolving software as it acts as a shared mental model of a system expressed at a high level of abstraction [8], helping the architect plan and restructure the system by abstracting away from technological issues [9]. This alternative has been named architecture-based software evolution. But SA is also one of the results of the evolution because

*Correspondence to: Elena Navarro, Computing Systems Department, University of Castilla-La Mancha, Avda. España s/n, Albacete, Spain.
†E-mail: elena.navarro@uclm.es

Copyright © 2012 John Wiley & Sons, Ltd.
it may also evolve to be consistent with the changes as they emerge. Therefore, the introduction of mechanisms, techniques, processes, etc. that guide the architect while evolving the SA becomes a critical issue to maintain its consistency, quality, etc.

In the very beginning of the SA area of research, the architectural design rationales (ADRs) that explain the SA specification were understood as very valuable assets of the architecture process. Perry and Wolf, in a well-known article [10], defined the SA as a model composed of elements, form, and rationale. The first item refers to the description of components and what would be later defined as connectors; the second item refers to constraints in their properties and relationships; and the third one was defined as the motivation for the choice of style, form, or elements, which explains why this choice satisfies the system requirements. However, the rationale has been scarcely considered until recently when a group of researchers highlighted again its importance [11, 12]. It can be seen as a computational structure, composed of small assets of design knowledge, tracing back to some requirements and forward towards an implementation: the extended discourse of the system’s design, defining our architectural knowledge (AK). AK is therefore composed of architectural elements, requirements, and a number of design assets. There are several ways to represent them—for example, we talk about architectural design decisions (ADDs) and ADRs, which comprise a concrete decision in the process and the reasoning behind it. When only the final architecture is described, all this information is unrepresented design knowledge [13]; but now architecture includes this information as part of the rationale.

When we take into account the importance of a rationale to understand why the system is the way it is, it becomes a challenging question to evaluate what the implications are while evolving the system. Bratthall et al. [14] dealt with this question by carrying out an experiment with 17 subjects from both industry and academia and concluded that most of the interviewed architects stated that by using the ADRs they could shorten the time necessary to carry out the change tasks. Interviewed subjects also concluded that the quality of the results was better using ADRs when they had to predict changes on unknown real-time systems. Therefore, there are compelling arguments for the exploitation of the rationale while the SA is being evolved. However, this use of the rationale turns it into a passive actor of the evolution process as it is simply used as a documentation artifact. It is at this point where this work focuses its attention: can a rationale become an active actor of the evolution that drives its application? In this work, we describe how this question can be answered positively by means of evolution styles [5]. Concretely, an extension to these styles, which we have called AK-driven evolution styles (AKdES), is presented, which considers the introduction of ADDs and ADRs in their description to guide the architect while evolving the SA.

This paper is structured as follows. After this introduction, Section 2 describes the necessary background about software evolution and AK. Section 3 presents our proposal, AKdES, which intends to combine the novel concept of evolution styles with the basis provided by AK. Section 4 introduces a specific process, ATRIUM [15], as a proof of concept to show the applicability of our proposal. Section 5 demonstrates by means of a case study how our approach can be put into practice. And finally, Section 6 describes our conclusions and further research.

2. RELATED WORK

The implications of the SA in software evolution have been highlighted largely in the literature [16–18]. Most of the works focus their efforts on the exploitation of the SA as a key driver for the evolution, that is, what have been called an artifact for the evolution. It is worth noting that these works exploit the SA specification as the AK of the system. For instance, an interesting work in this area is the Evolution Tailored with AK framework proposed by Noppen and Tamzalit [19]. In this work, a new concept is introduced named architectural trait, which refers to the properties the architect wants to consider during the evolution no matter the evolution pattern he wants to apply. These architectural traits are defined by identifying a set of components and connectors that the architect wants to examine to determine their relevance in the evolution to be performed. Therefore, the use of architectural traits entails turning the SA evolution into an evaluation process that assesses the relevance of the architectural traits with regard to the architecture space that the architect retrieves from the architecture
knowledge base. However, this architecture knowledge base does not describe design decisions (DDs) and design rationales (DRs) per se but different SA specifications that can be reused to address different needs. Therefore, although it exhibits some similarities to our approach, it also exhibits clear differences.

Another related work that also considers the exploitation of a knowledge base to enact the software evolution has been presented in the area of architecture-based self-adaptive software. For instance, the Knowledge-Based Architectural Adaptation Management approach [20] introduces as first-class architectural entities both adaptation policies and relevant system knowledge, which are dynamically managed and reasoned over by an expert system to drive the adaptation process carried out by the framework. The system knowledge refers to architectural observations that come from either events of the system itself or external information communicated to the system and can be related to structural changes or potential problems. An architectural adaptation manager uses these architectural observations to determine which adaptation responses must be generated—that is, which structural modifications or adaptation policy changes must be performed. In a similar way to the previous work, this work does not deal with ADDs or ADRs.

Several approaches have focused on the evolution of the SA where they consider SA as an artifact of the evolution. Several points of view have been considered for this evolution. Some works have focused on how the quality factor evolvability can be introduced and considered during the SA design. For instance, the adaptability evaluation method [21] specifies adaptability requirements and guides the architect in their consideration during the SA design by means of a set of guidelines and in their analysis to determine whether these requirements are met. Another work focuses on the SA evolvability evaluation during the final phase of its design. In this category, a well-known proposal is the SA analysis method [22], which evaluates the SA relative to several scenarios that describe likely future changes to the system, helping to estimate the amount of work to carry them out.

Finally, another alternative for dealing with SA evolution is by using AK. As stated by Avgeriou et al. [23], the AK turns into an integrated representation of SA specification, ADDs, and ADRs. Most of the works [24–26] in this area have been related to the ADD modeling to describe their structures, traceability relationships, etc. Other works [27, 28] have focused on the definition of supporting tools for AK management. Many approaches have also highlighted the existing close relationship between AK and SA evolution. For instance, Burge, Carroll, and McCall stated in [29] that as the rationale describes the history of how and why the system has been modified over time, it ‘should be captured for the change’ to determine where problems have usually happened and where they are likely to emerge. Other approaches [30, 31] have also highlighted that the necessary work to specify AK really pays off when the system is in its maintenance and evolution phases by helping to reduce the costs of these phases.

However, despite the importance of AK during the evolution process, to the best of our knowledge, only two approaches expressly exploit the AK as an active actor of the evolution process. One is that presented by Tang et al. [32]. They presented a graphical model, named Architecture Rationale and Element Linkage, which is described as a Bayesian belief network, to describe the causal relationships between ADD and architectural elements [33]. Using this representation, three probability-based reasoning methods can be applied: (i) predictive reasoning is used to determine which design elements could be affected by a change if some architectural elements were to change; (ii) diagnostic reasoning is introduced to determine what could be the causes for those architectural elements to change; and (iii) combined reasoning is used to combine the previous results and reason about what the likely changes to the system are. This work is very interesting in that it enables the architects to predict the impact of change before it happens. Therefore, this work can be considered as complementary, as our work is oriented to guiding the evolution process.

Another one is that proposed by Tibermachine et al. [34]. They proposed the introduction of a family of languages, called architectural constraint languages, that allow the specification of the architectural choices associated to the decisions that are made during the development process. There are two main constituents of the architectural constraint languages: (i) the core constraint language, a modified version of the Object Constraint Language, which provides navigation operations, operators, and usual quantifiers; and (ii) a set of Meta-Object Facility [35] meta-models to describe abstractions found in the main modeling languages that are used at different stages of the life cycle. Each pair of
Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation

both elements is called a profile, being defined a profile for each stage of the development process and modeling language. For instance, authors proposed in this approach that a profile could be used to formalize the architectural constraints related to the architectural design stage and the Acme Architecture Description Language [36]. The main advantage of this proposal is that architectural decisions, such as architectural styles [37] or design patterns used for the system, can be described by means of the formal language core constraint language and be related to the specific architectural elements. It means that whenever a change is performed, the conformance of the architectural model can be checked after its evolution. Therefore, this approach is more oriented to validation rather than guidance of the evolution process.

The notion of architectural style, as originally introduced by Perry and Wolf, is a concept ‘which abstract elements and formal aspects from various specific architectures’ [10]. It is a prescriptive rather than a descriptive concept; the same system or configuration may simultaneously comply with several style definitions, that is, the resulting architecture would have several styles at the same time. The idea is that the style defines a series of constraints—the range of systems that fulfill these constraints gathers the members with that style. As the style becomes more specific, this range is smaller, and the prescription becomes almost a description.

There is an obvious relationship between software evolution and architectural styles, although it has not always been made explicit. The style guarantees that the architecture holds a series of properties—and usually, these properties are high-level features that are maintained during the system’s evolution. Therefore, different configurations of an evolving architecture have the same style—that is, in general terms, the architectural evolution respects the defined style. Disregard of architecture and constraining styles lead to architectural drift and erosion [11].

In recent times, several authors have tried to exploit this intuition, developing a new, although related, concept—that of evolution style. In short, where the original architectural style described the constraints to be fulfilled by a set of systems, in an evolution style, this set of systems would be the set of potential configurations of an evolutionary system—that is the style defines the constraints to be fulfilled by any possible evolution of the system. Where the original notion prescribed the architecture, this variant prescribes its evolution.

Tamzalit and others [19, 38, 39] have developed a series of papers on the topic. Their notion of evolution style uses the original concept of architectural style to delimit the range of changes to a given system. They intend to provide generic transformations (evolution patterns) that describe these changes as series of steps (evolution operations). To be generic, these transformations have been defined to fulfill the constraints of an architectural style—therefore, evolution is within (or towards) a style. These concepts have been developed on top of graph transformations, at least at the level of a proof of concept. Although Tamzalit’s work explicitly refers to the impact of evolution styles on AK [19, 38], none of these proposals make use of existing AK to define or even influence those styles. Much of this information is system specific, and this makes it even more relevant to our work. Our proposal intends exactly to fill this gap, that is, to define evolution styles that make explicit use of AK.

Inspired by their work, Garlan et al. [9] have also made some developments in this context. Their evolution styles are directly defined as transformations on styles: they provide an initial style, a target style, and possibly several sequences of intermediate evolutionary steps, known as evolution paths. These paths define different ways to evolve from the initial to the target style, using architectural operations to generate ‘evolving’ intermediate styles. They also define a set of path constraints, which are specified using temporal logic and predefined evaluation functions. This makes it possible for the architect to perform certain analysis. Our approach, AKdES, is to some extent similar to those mentioned—the style will be described as a sequence of steps. But the defining feature is that our evolution decisions are influenced by the existing AK. In the Garlan et al. approach, path constraints are bound to make decisions using just structural information, such as the number of nodes—thus the need for predefined functions. However, in our approach, any decision ever made is still available, so our evolution decisions can take even past decisions into account. In particular, our approach is able to detect when the conditions that led to a decision in the past do not hold anymore and to decide if this change in conditions is enough to cause, or least to enable, an evolutionary step. Our approach is designed not to depend on any existing description.
language or platform, although it has to be adapted to any specific approach. Our only assumption is that we use structured AK (SAK)—that is, that the system includes an explicit representation of AK.

Table I summarizes the conclusions presented in this state-of-the-art section. Every existing proposal is compared with the rest of them (including our proposal, AKdES) using a set of features. Namely, the second column states if the proposal has explicit support for AK (or some comparable concept, in the case of partial support), whereas the third one indicates if this AK is structured. The fourth one relates to evolution and states if this proposal has an explicit support for evolution using some structural concept. The fifth one indicates if the proposal uses explicit AK to evaluate if some evolution must happen, whereas the sixth states if the proposal includes explicit evolutionary steps as the result of that evolution. Finally, the last column indicates if the kind of evolution it supports is significant at the architectural level.

As can be observed, Table I makes it easy to identify four different categories. The first group (the first four proposals) is composed of methods that do not use explicit AK, but some comparable abstraction; except for Knowledge-Based Architectural Adaptation Management, all of them use this knowledge just for evaluation purposes, but they do not trigger an evolution process as the result of this evaluation. Only two of them consider this evolution at the architecture level. The second group (next two proposals) includes methods that have explicit AK, and this is evaluated for evaluation purposes, but it does not include explicit evolution structures or processes. The third group (next two proposals) joins the existing definitions of evolution styles; almost reversing the second group, they include support for evolution structures and processes, but they do not use AK to make their decisions. Finally, the last group (including our own proposal, AKdES) is defined by having explicit AK support, and it uses this knowledge to trigger an evolution process, supported in turn by explicit structures (i.e., styles), acting at the architecture level. In summary, this can be found at the intersection of the second and third groups—including both AK and evolution styles.

3. PROPOSAL: COMBINING EVOLUTION STYLES WITH ARCHITECTURAL KNOWLEDGE

3.1. Architectural knowledge as an evolution driver

As noted earlier, the notion of evolution style has been initially defined [39] as both a variation and an extension of the concept of architectural style, as originally conceived by Perry and Wolf [10]. Styles express the central idea of the prescriptive view of SA—that is, the architecture prescribes a set of design conditions, which are specified as a set of architectural constraints: every configuration that fulfills these constraints complies with the corresponding architectural style. The notion has become popular, and it is also considered from a descriptive point of view—where it is usually expressed as a vocabulary set of architectural elements and a set of integration constraints, mainly of a topological nature.

Table I. A comparison of existing proposals considering both architectural knowledge and evolution to some extent.

<table>
<thead>
<tr>
<th>Architectural knowledge</th>
<th>Structured AK</th>
<th>Evolution structures</th>
<th>AK-based evaluation</th>
<th>AK-based evolution</th>
<th>Architecture-level evolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETAK</td>
<td>(traits)</td>
<td>×</td>
<td>√</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>KBAAM</td>
<td>(observations)</td>
<td>×</td>
<td>√ (policies)</td>
<td>√</td>
<td>√ (dynamic)</td>
</tr>
<tr>
<td>AEM</td>
<td>(requirements)</td>
<td>×</td>
<td>×</td>
<td>√</td>
<td>×</td>
</tr>
<tr>
<td>SAAM</td>
<td>(scenarios)</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>AREL</td>
<td>√ (CCL)</td>
<td>√ (BBN)</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Tamzalit</td>
<td>×</td>
<td>√ (patterns)</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Garlan</td>
<td>×</td>
<td>√ (paths)</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>AKdES</td>
<td>√</td>
<td>√ (BBN)</td>
<td>√ (paths)</td>
<td>√</td>
<td>√ (styles)</td>
</tr>
</tbody>
</table>

√, full support; √, partial support; ×, not supported.

Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation

Styles are also obviously related to evolution, as they can also constrain it. This can be considered from both sides of the coin. First, a dynamic architecture, by definition, also specifies an architectural style. That is, the architecture evolves (describes several different configurations), but at the end of the day, it continues being one and the same entity. Thus, the whole sequence of intermediate configurations has to maintain a set of basic constraints, which should capture the essence of the evolving architecture. By definition, this set of constraints specifies an architectural style—as previously noted [40].

On the other hand, we can take the opposite view. Rather than deduce the constraints of the style from the stable part of the architectural evolution, we can prescribe the boundaries of that evolution using the notion of style. Therefore, from this point of view, the architectural style is a set of constraints that the system must fulfill at every stage during its evolution. The system can evolve, in principle, in any way that does not explicitly violate any of the rules described in the style. Indeed, this meaning has been implicitly present from the first definition of the concept [10]—we should just emphasize that the architectural style constrain not only the current architecture but also any future version of it.

Thus, the notion of evolution style emerges not exactly as a new concept—but as a specific variant of the generic notion of architectural style; indeed, it is presented that way in the original proposal [38]. Of course, there are differences—notions that we specifically introduce to harness even further the evolution scheme. In particular, the different proposals defining the notion of evolution style provide additional constraints for the evolutionary process. The common idea is to introduce some kind of restriction about the way in which this evolution may happen, that is, about the sequence(s) of intermediate steps that can be carried out.

As already mentioned, the existing proposals [9, 38] that define the notion of evolution style have used different ways to constrain this evolution, which are specified in the following:

- Tamzalit’s approach, as presented in [38], is conceptually able to use topological, behavioral, or communication-oriented constraints. However, this proposal so far has only dealt with topological rules. These evolution styles have a syntactic part and a semantic part; but only the first one is relevant to constrain evolution. The syntactic specification consists of a header, which defines the evolution task, and an optional competence, which provides an implementation method (often deferred to external technologies). The header provides three essential constraints for the style: a precondition, a postcondition (i.e., a goal), and more importantly, an invariant. These conditions are expressed using predicate logic or a comparable formalism (e.g., Object Constraint Language).
- The Garlan et al. approach [9] is similar to the preceding one, although it has a more operational flavor. As already noted, it begins with an initial style and has a target style; the way to evolve from the first to the last describes an evolution path, possibly several ones. The emphasis on evolution is clearer in this case, as constraints are imposed to evolution paths, rather than to the base architecture. Therefore, the style still conforms to the original definition [10], but here, it is clearly evolution itself that is being constrained. The operational nature is defined by construction—paths are divided in discrete evolution steps; and their constraints are described using a temporal logic.

Therefore, not only are both proposals similar in structure and scope, they have also been specifically conceived to constrain evolution using just topological information. This is not bad in itself—in fact, in the context of SA, this is both a coherent choice and a usual practice. But, at the same time, it does not exploit all the potential of the concept.

As already noted, the growing body of research on AK makes it possible to obtain and use much more information about the architecture, which is not limited to topological constraints anymore. For instance, this information includes many of the choices made during architectural design (i.e., design decisions); therefore, when the circumstances that led to a certain decision vary, the potential to perform a change in the architecture appears. Obviously, this provides even better opportunities for evolution than just topological information, as it is based in the system’s own specific requirements.

Then, our proposal, AKdES, is to use AK in the context of evolution styles, both to constrain and to trigger evolution in SAs. Therefore, in our proposal, evolution styles are similar to those in previous
proposals (particularly to those of the Garlan et al. approach) but explicitly include AK techniques. In fact, this combination is also implicit in the original definition of style [10], considering the role of the rationale. However, it has never been discussed before in this context. Tamzalit’s approach has an explicit interest for ‘evolution reuse’, and it even makes an explicit reference to the exploitation of AK [38]. However, this is just a side reference, and existing work concentrates just on topological constraints.

The central idea of our proposal is summarized in Figure 1. As already noted, it is inspired to some extent by the Garlan et al. approach, although many details are different. We can say that our proposal is based on that of Garlan et al. as much as their proposal is based on Tamzalit’s. This means that there are many commonalities, but the approach itself is quite different, and it is supported by very different technologies.

Figure 1 depicts a concrete step in the evolution of an architectural model. Each ‘frame’ represents a different moment in time, building a temporal sequence as indicated by the time arrow. The model in every frame describes the current architecture (configuration) and the relevant AK. In this example, the AK is structured; the names of the AK relationships from the decision tree have been taken from [41]. Namely, the first frame (at time t) depicts a configuration with two components (right oval), along with the decision tree, which led to this configuration. This tree shows also that a certain decision (A) has influenced our final choice, implying the selection (constrains) of configuration C instead of the alternative B, which is rejected (inhibits). The rejected configuration (B) would have led to a three-component architecture (left oval). This is the initial situation.

Then, something happens: an evolution step, as reflected by the central arrow (Δ). In short, that change means that our previous A decision is modified. And then, the new situation appears as a frame in time t+1. The configuration has now three components, and the decision tree has been altered. Now a new decision (A′) is inhibiting the old choice (still named C), and now the B alternative is selected (our choice is now constrained to this specific configuration). Therefore, an evolution step has happened.

In short, our proposal intends to define evolution styles in the context of a SAK-based system, similar to Garlan’s proposal [9], as sequences of such steps, evolving the system from the configuration in time t to the situation in t+n, and defining a consistent and reusable evolution process in between.

3.2. Defining evolution styles

Our definition of an evolution style is constructive: rather than providing a direct definition, we start by describing the elements that compose it until we reach a final form. Then, we use a bottom-up, rather than a top-down approach. Our main purpose is knowledge reuse, not only in the architecture but even in its evolution—indeed, the main goal of architectural styles and even of knowledge itself.

Figure 1. Architectural knowledge to evolve software architecture.
This section provides an abstract definition for an evolution style, and it can be applied to any process that fulfills the requirements summarized in Section 3.6. However, when we choose a specific process, we can obtain a much concrete definition. We outline ours in Section 4. Then, in our proposal, an evolution style is composed of these elements:

- **Evolution conditions**: a (potentially) system-wide condition that triggers a change, that is, requiring an evolution step. In short, the identification of a situation that should be handled by evolving the system. Many of these evolution conditions must be understood as changes on the initial assumptions, and therefore, it can affect previously existing decisions (ADDS). Section 3.3 discusses these elements in some more detail.
- **Evolution decisions**: a choice made as reaction to an evolution condition. This choice must be stored as part of the AK, possibly using a different category than other design decisions. Note that this decision will still be related to other architectural decisions, using the usual relationships.
- **Evolutionary steps**: what happens when a decision is chosen—that is, evolution happens. Note that this definition depends on the decision, not on the specific architecture. Therefore, a single evolutionary step may imply several technological steps at the architectural level—even a full reconfiguration; but just a single decision is used.
- **Evolution patterns**: a sequence of evolution decisions, possibly chained within a logical reasoning, which could be described as an evolution rationale. A pattern defines a sequence of steps: every decision creates an alternative branch and can lead us to a new evolution decision, until a termination condition is met. The result of the application of such a pattern results in the form of a decision tree within the AK structure. Evolution paths in the Garlan et al. approach [9] can be considered as a related construction of these (or a variant case).
- **Evolution styles**: a set of evolution patterns that are conceptually related—that is they affect either the same set of features or a set of related features that, together, could achieve some combined effect. Note that this set of patterns need not to be connected, and of course, it includes the unitary set as a special case; that is, the simplest evolution style consists of a single pattern.

Therefore, the structure of an evolution style in our proposal can be quickly summarized as such: an evolution style is a set of conceptually related evolution patterns. An evolution pattern describes the answer to a specific evolution condition, which triggers a decision process where a different configuration is chosen among several alternatives. Once the alternative configuration has been selected (an evolution decision is taken), the system executes an evolutionary step, modifying the AK structure in the process. If this step satisfies the evolution condition, the process finishes; if not, the evolution may continue.

Figure 2 depicts a representation of an evolution pattern or, more precisely, the effects of applying a certain evolution pattern in the AK structure. Our definition is constructive, and the notion of an evolution pattern is inherently dynamic. So, to understand this picture, we should consider it as the final result of applying a pattern consisting of three steps, that is, the situation at time $t+3$.

![Figure 2. A presentation of an evolution pattern and its effects.](image-url)
The process develops as follows. First, an evolution condition (represented by the top-left oval) is detected, and the application of the pattern is triggered. Then, a certain ADD (ADD.1) is chosen, and as a consequence, the architecture evolves from the initial version (not shown in the picture) to a different configuration (ARCH.1). This is the first evolutionary step, and the decision to perform this step is captured in the first evolution decision (ED.1). Therefore, in the first version, ED.1 implies the choice of ADD.1, and hence, they would be initially related by a constrains relationship.

Then, we assume that the evolution condition requires further evolution, and a second evolutionary step must be performed. As a result of this, the previous decision (ED.1) changes, and a new one (ED.2) is made. The change on ED.1 means that now ADD.1 is rejected, and therefore, the relationship between them changes to inhibits (as shown in Figure 2). This step is similar to the presentation made in Figure 1. At the end of the second evolutionary step, the decision ED.2 implies the choice of a new ADD (ADD.2) and of the corresponding configuration (ARCH.2).

The evolution process continues; and a third evolution step is executed. Then, like in the previous case, a new decision (ED.3) is made, and the previous one (ED.2) changes. This is exactly the situation as depicted in Figure 2: now, ED.2 inhibits ADD.2 and constrains (i.e., implies) ED.3 instead. The decision ED.3 implies the design decision ADD.3, which leads to the final architecture (not depicted in Figure 2). Then, the termination condition (bottom-right oval) is met, and the application of the pattern finishes.

Of course, many different patterns may appear; also, many evolution paths can be taken—the emphasis here is that every time a pattern is applied, its effects modify the AK structure. This makes it possible to trace the application of previous patterns and possibly to abstract from these structures, to obtain new patterns from pre-existing sequences of evolutionary steps.

3.3. Evolution conditions: detecting the need for evolution

By integrating AK in the evolution process, we intended to assist the architect in detecting when and how the architecture should evolve. AK can be used for evolution in three ways:

- AK can help to detect when a change is allowed.
- AK can help to detect when a change is required.
- AK can help to determine which evolution style should be chosen to evolve the system, among several potential candidates.

In this section, we focus on the first option. Much of the research on architectural evolution has focused on how we can do an architectural reconfiguration, but less emphasis has been put on deciding when to do it. The reason is, probably, that much of the relevant information was not explicit (it was ‘unrepresented design knowledge’ [24, 26, 27])—but now, this information is included in the AK.

The decision to evolve is semi-automatic, in the sense that a human has usually the last word. But our reason to trigger the evolutionary step can often be encapsulated in a logical formula, an evolution condition. When this condition is true, then evolution should usually happen.

Evolution conditions are described as a logical expression—or alternatively as plain text—which can gather and combine any number of the following factors:

- Situations in the current system architecture—that is, mostly structural or topological features (e.g., number of connections), as in many other proposals
- Situations in the external context—that is, something that happens on the outside, including human intervention—and therefore unpredictable by its own nature
- Active decisions—that is, an architectural decision made in the past and included in the AK, which is still directly affecting the current SA. Questioning if such a decision is still justified considering the current context, and even possibly revoking it, is a standard method for triggering a change to the system.
- Past decisions—that is any decision made in the past and included in the AK, which was either taken or rejected. This past decision can still affect active decisions by means of AK relationships (as seen in Figure 1 and, particularly, in Figure 2) or can be used as a ‘memory log’ to decide upon the current decision, accessed by means of traceability.
Evolution conditions, described in this way, are able to access more information about an existing system than many other approaches—indeed, they are potentially able to know as much about the system as the architect himself. Therefore, they provide the means to define every single step within our evolution style.

3.4. Evolution decisions: linking to architectural knowledge

As already noted, our approach to software evolution relies on the assumption that the system to evolve has an explicit architectural rationale (i.e., the evolutionary system has also an explicit specification). In the ideal case, this architectural rationale is not just a plain text description (which would be useful nonetheless), but a structured representation that we are able to handle—that is what we have called SAK. Again, neither the language nor the concrete representation is important: the only strong requirement is to be able to access and use this information.

The SAK assumes an explicit representation of AK and that this representation includes a set of internal relationships—i.e., it is structured. The set of relationships may vary with the architectural language or platform. There are several proposals in the literature: to a certain extent, the set of relationships is not as important as the structure (the network) they define [42], which could even constrain our future evolution. Nevertheless, for the remainder of the discussion, we will use the set that we ourselves described in [41] and is also used in Figures 1 and 2.

Essentially, we just need to know that constrains is a direct relationship (decision A implies decision B) and that inhibits is an inverse relationship (decision C hinders but does not forbid decision D). There are other relationships, more prominently excludes, but in this article, we just need to refer to these two.

The connection between the evolution process and the SAK is provided by evolution decisions, as defined in Section 3.2 and exemplified in Figure 2. In a sentence, any decision about the system is an ADD, and it deals with the system itself; but a change in the architecture defines an evolution decision, that is, a decision about ADDs. Of course, evolution decisions are not strictly necessary: the AK network can describe the same ADDs without using them—but their purpose is to serve as a link to AK and also to assist in the definition of evolution styles.

Therefore, the style is built bottom-up: the architect, while working with a concrete system, identifies a significant step (i.e., a concrete decision) in evolution, which can be abstracted from the specific AK to a generic situation. Then, he gathers sequences of such steps, and these define patterns. And finally, a set of related patterns defines an evolution style.

As in many other cases, these styles are created out of scattered fragments of knowledge—but they are still built in a way that guarantees conceptual coherence. Evolution styles must be applied to a system—or, more precisely, into the AK of a system. This will be known as the base system for the style. As styles are conceived to be generic, they must be adequately adapted and parameterized—but once they are superimposed to specific decisions and components, they can be useful in many different contexts.

3.5. Applying an evolution style

The evolution style is conceived to be applied as part of a semi-automatic process: every step in every pattern corresponds to an evolution decision—and both in architecture and in evolution, those decisions are usually taken by humans [43]. In that sense, any evolution, that is, a change to the AK and therefore to the associated architecture, has to be explicitly approved by the architect.

Evolution styles would be useful even if they were purely documental; however, we must assume some kind of automatic support. For instance, the existing model-driven support for design decisions and architectural styles that is present in the Morpheus toolset [44] can be easily extended to describe this kind of structures and to provide assistance to the human architect. The resulting system would be able to suggest a potential evolution to the architect, and he would just need to accept it—then, evolution will happen.

The application of an evolution style, with the support of an automatic system, is simple once we have evolution patterns in the form of an AK decision tree—just like the one in Figure 2. First, the system must detect an evolution condition: this can be performed automatically if the condition is a logical expression; when it is provided as plain text, it is the architect who must decide about it. Then, the sequence of evolutionary steps is followed in the original order: the first ED is considered, and the corresponding
ADD is applied. Hence, the first step is performed. If the evolution condition has been satisfied, the process can stop here; otherwise, the second ED is applied, the previous ADD is inhibited, and a new ADD is taken. This is repeated, step by step, until the termination condition is met. The termination condition can be quite simple: for instance, just that the architectural pattern has finished.

Of course, in the few cases when human intervention is not required, evolution styles might have everything they need to take a decision—hence, they could automatically perform the full evolution process without further assistance, using the same model-driven techniques referred to earlier. However, this kind of situation is rare, and it does not describe the general case, so it will not be considered in the following.

3.6. Requirements to apply the proposal

As aforementioned, the proposal has been defined (so far) in an abstract way, so that these ideas can be used in many contexts, no matter the specific process we use for evolution. However, in the previous sections, a number of ideas have been assumed to be able to apply it. Therefore, the process must at least comply with the following requirements:

- **SAK**: the approach consists in exploiting this AK; hence, it must be available. As already noted, our proposal takes advantage of the structure of this knowledge—therefore, SAK must be assumed. The elements of this knowledge must include ADDs and rationales (ADR).
- **Process description**: our process must at least cover development until the architectural design stage; from this and the previous point, we can safely assume that the process provides support and stores information from requirements to the architecture.
- **Degree of automation**: the complexity of this approach makes automation a necessity, rather than an option. As we are dealing with several kinds of models (requirements and architecture), their evolution, and their transformations, we also require a model-driven development (MDD)-based approach.

In summary, our abstract proposal can be applied to any development process that stores SAK from requirements to architecture and that is able to handle this information and process by means of an MDD-based approach.

4. PROOF OF CONCEPT: ATRIUM FOR EVOLUTION

To validate the approach, we have selected a process that allows us to deal with AK and to manipulate its models in an easy way, ATRIUM [15]. It has been designed for the concurrent definition of requirements and SA, providing automatic/semi-automatic support for traceability throughout its application. In the following sections, we explain why ATRIUM has been chosen, briefly introduce ATRIUM, and present our approach.

4.1. Reasons to choose ATRIUM

As noted in Section 3.6, our abstract definition of architectural styles can be applied to any development process that fulfills three requirements. ATRIUM does comply with these requirements, and therefore, it is able to use the ideas in our proposal. The reasons are specified as follows:

- **ATRIUM** provides explicit support for AK, as explained in [44]. In fact, it has been extended to provide a very complex structure of AK in a simple way, using just three AK relationships [41]. Therefore, it does support SAK and, in fact, has been explicitly chosen for the richness and detail of its SAK support.
- **ATRIUM** was specifically designed to support the development process from the requirements to the architectural stage. The details of this process will be summarized in the next section.
- **ATRIUM** is an MDD process [45] and, in fact, was designed as such from the start. First, it provides support for the process itself, from requirements to architecture [15]. Second, it also supports the management of AK using MDD-based techniques [44]. Hence, this support can also be extended to support evolution.
In the rest of the article, and for the sake of clarity, we will assume that we are using the ATRIUM-specific version of our proposal, rather than the most abstract one. However, our conclusions should hold for any other process that fulfills the noted requirements.

4.2. Describing ATRIUM

Figure 3 shows, with the use of SPEM 1.1 [46], the main activities of ATRIUM. These activities must be iterated over to define and refine the different models. These activities are described as follows:

- **Modeling requirements**: This activity allows the architect to identify and specify the requirements of the system to be by using the ATRIUM goal model, which is based on Knowledge Acquisition in Automated Specification [47] and the non-functional requirement framework [48]. This activity uses as input both an informal description of the requirements stated by the stakeholders and the ISO/IEC 25010:2011 Software Product Quality Requirements and Evaluation [49]. The latter is used as a framework of concerns for the system to be. In addition, the architectural style to be applied is selected during this activity.

- **Modeling scenarios**: This activity focuses on the specification of the ATRIUM scenario model, that is, the set of architectural scenarios that describe the system’s behavior under certain operationalization decisions [50]. Each ATRIUM scenario identifies the architectural and environmental elements that interact to satisfy specific requirements and their level of responsibility.

- **Synthesize and transform**: This activity has been defined to generate the proto-architecture of the specific system [44]. It synthesizes the architectural elements from the ATRIUM scenario model that builds up the system to be, along with its structure. This proto-architecture is a first draft of the final description of the system that can be refined in a later stage of the software development process. This activity has been defined by applying model-to-model transformation techniques [51], specifically, using the QVT-Relations language [52] to define the necessary transformations. It must be pointed out that ATRIUM is independent of the architectural meta-model used to describe the proto-architecture because the architect only has to describe the needed transformations to instantiate the architectural meta-model he deems appropriate. However, the necessary transformation to generate PRISMA architectural models [53] has been already defined.
4.3. Architectural knowledge in ATRIUM

As presented in [41, 44], DDs and DRs are introduced from the very beginning of the software development process, specifically, from the requirement stage, thanks to their specification in the ATRIUM goal model. Goal, requirement, and operationalization are the building blocks of this model, as shown in Figure 4. Goals constitute expectations that the system should meet. Requirements are services that the system should provide or constraints on these services. Finally, an operationalization is a description of an architectural solution, that is, an architectural design choice for the system to be to meet the users’ needs and expectations. They are called operationalizations because they describe the system behavior to meet the requirements, both functional and non-functional. For this reason, two key attributes are included while they are described: DD and DR. A seamless transition is performed from requirements to operationalizations by means of the contribution relationship, to specify an operationalization contributing to/preventing the satisfaction of a requirement facilitating the automatic analysis of architectural alternatives [15].

One of the main advantages of AK management is the capability to explore the reasoning in the SA by exploiting the network of AK. To provide ATRIUM with this facility, several relationships were defined in its meta-model to allow the analyst to describe the AK as a network [41]. As shown in Figure 4, these relationships were first specified on operationalizations, as they are in charge of describing both the DDs and the DRs and can be defined as follows:

- **Constrains** is a binary and unidirectional relationship with positive semantics. Let us consider A and B operationalizations, describing different design decisions. Having a constraint relationship from A to B means that B’s design decision cannot be made unless A’s design decision is also made.
- **Inhibits** is a binary and unidirectional relationship used to specify negative semantics. Let us consider A and B operationalizations, describing different design decisions. Having an inhibition relationship from A to B means that if A’s design decision is made, it hinders the making of B’s design decision.
- **Excludes** is a binary and unidirectional relationship with stronger negative semantics than inhibits. Let us consider A and B operationalizations, describing different design decisions. Having an exclusion relationship from A to B means that if A’s design decision is made, it prevents B’s design decision from being made.

As presented in [44], one of the advantages of ATRIUM is that it facilitates the generation of the DDs along with the proto-architecture, so that each architectural element is related to the set of DDs that motivated its specification and the DRs that justify those decisions. Figure 4 shows (part of) the

![Figure 4. Describing architectural knowledge in ATRIUM.](image-url)
It can be observed that every ArchitecturalElement is related to a set of DesignAssets that describe both its DDs and DRs. As can be observed in Figure 4, the DesignAssets can be related by means of constrains, excludes, and inhibits relationships in a similar way to the operationalizations in the goal meta-model.

It is worth noting that the main difference is between operationalizations in the goal model and DesignAssets in the architectural model. The former are in charge of specifying all the DDs and DRs that were analyzed during the specification of the system, that is, they describe the reasoning carried out to evaluate which were the best alternatives for the system. The latter describe the reasoning behind the current specification of the system, that is, why the system has its current specification. Therefore, they help to maintain AK from different perspectives.

4.4. Describing evolution styles in ATRIUM

Considering the constructive definition of evolution styles, as provided in Section 3.2, it is quite clear that most of the relevant notions are already present in ATRIUM, and hence, they do not require any extension of the standard framework. Therefore, they are described as follows:

- **Evolution conditions**: The condition that triggers an evolution process. It can be described as plain text or using a logical formula. Particularly in the second case, the automatic support in ATRIUM can be extended to provide some automatic detection of the condition; but most of the time, this would be conceived and decided by the architect himself (i.e., by human intervention). In summary, there is no need to explicitly describe these conditions as part of the model.

- **Evolution decisions**: These provide the decision to perform an evolutionary step. They are a special case of design decisions, as they are decisions on decisions. Just like conventional ADDs describe information and choices about architectural elements, these evolution decisions describe choices about ADDs themselves. As already noted in Section 3.4, they are provided to serve as a link between the AK and the evolution process—but they are not strictly necessary in terms of the AK. However, their role is very important in describing an evolution style, as they provide the basic skeleton for this structure—in the form of a decision tree. Therefore, ATRIUM provides explicit support for these elements, in the form of the EvolutionAsset entity. This is defined in the architectural meta-model, as shown in Figure 5, as a special case of DesignAssets.

- **Evolutionary step**: It describes an action, that is, the evolution from a situation to another situation. Hence, it does not require any explicit representation. However, every such step leaves a definite trace in the structure of the AK. Just consider the abstract process as defined in Section 3.2 and seen in Figure 2. Essentially, for every step, a decision is made (captured as an EvolutionAsset). This decision affects a certain ADD (captured as a DesignAsset, which in turn relates to several ArchitecturalElements) and depicts this influence using a constrains relationship. The sign of this relationship can eventually be modified (as already seen) by further decisions.

- **Evolution pattern**: As a pattern is a sequence of evolutionary steps, it is not necessary to provide any additional concept to describe this notion. In fact, provided that every step is captured by the
three elements construction mentioned in the previous point (an EvolutionAsset, a DesignAsset, and the relationship between them), a pattern is shown as the sequence of such triplets—where every evolution decision is expanded as an additional branch, therefore taking the form of a decision tree. Again, the structure provided in Figure 2 depicts such a decision tree, and there is no need to provide any special construct to capture it—apart from giving it a name so it can be reused.

- **Evolution style:** Similarly, an evolution style is just a set of evolution patterns—therefore, it does not require any special construction either.

In summary, once the EvolutionAsset is provided, it suffices to be able to use already existing concepts (in particular DesignAssets and their relationships) to constructively build up a reusable representation for an evolution style—once it is parameterized—to abstract this definition from the concrete elements it affected in its first occurrence. The only special requirement of evolution patterns and styles is a distinctive name and a mapping (to apply the generic pattern to specific elements)—the rest is already provided.

Figure 5 shows how EvolutionAsset is described in the architectural meta-model. As it is defined as an extension, it inherits not only all its attributes, helping to define decisions and rationales about other DesignAssets, but also its relationships, which are used to establish how it affects other DesignAssets.

Moreover, although EvolutionAsset has been defined only at the architectural meta-model, it could be also defined in the goal meta-model. Thus, the architect could exploit model-to-model transformations to generate these elements in the goal model in an automatic way. This alternative could be helpful to carry out the evolution by taking into account both the AK and the requirements of the system. This alternative would be of interest to describe evolution conditions in the goal model, although the implications of its use are currently under evaluation.

The application of these evolutionary steps, in ATRIUM, follows the same lines as the generic definition provided in Section 3.2. First, an evolution condition is detected, and then an evolution decision (documented as an EvolutionAsset) is made; this evolution causes a concrete choice in the architecture, which is captured as a DD within a DesignAsset, and provides the corresponding relationship. The sequence defined by this process defines a decision tree by combining such triplets, therefore creating the equivalent of a structure that can be traversed and reused.

### 5. EXAMPLE: EVOLVING A CLOUD ARCHITECTURE

To illustrate the concepts introduced through the previous sections, in this section, we present a practical case for an evolving architecture—and how the management of AK leads to this evolution and to the definition of evolution styles.

Instead of a trivial example, we present a real-world case study, including a complex architecture with a certain set of features and that faces a complex problem. The purpose is twofold: first, to describe a problem in an interesting context, showing that our approach is not just a ‘lab construct’ and can still be applied within a non-controlled environment; and second, to show the actual power of these concepts, which is not perceived until applied to a complex problem.

Another interesting feature of this example is the reason to evolve, which is increasing costs. Therefore, the change is not required for some technical reason (and the system is complex enough to have plenty of these), but to fulfill a real-world necessity. This is relevant because this is the kind of situations that can only be adequately described when the AK is made explicit.

#### 5.1. Initial situation: the cloud-based radio station system

The example we present in this article has been developed in the context of cloud computing [54]. The reasons behind this choice are the current emphasis on this approach, which helps to situate the practicality of our proposal and also the fact that cloud architectures are of great importance for cloud-based applications—that is, the architectural level is particularly significant for its function, and therefore, its evolution is relevant for the system as a whole.
Perhaps the most important feature of cloud computing, and undoubtedly what distinguishes it from other related proposals, such as software as a service (SaaS), is scalability (also known as elasticity). A cloud-enabled application is executed within an elastic environment, which means that the application is able to automatically react to an increasing demand of resources—when they are necessary, simply there are more resources available.

Without loss of generality, let us conceive a cloud-enabled application as a set of independent services related to each other by means of queues and managed by specific controllers. This setup is sometimes referred to as ‘the canonical cloud architecture’ [55]. Most of the management policies have to be either distributed in the architecture or managed within those controllers.

Our example describes a cloud-based, on-demand ‘radio station’ that broadcasts on the Internet by using streaming techniques. This station has stored a large set of programs, which are uniquely identified. When a listener (a user) tunes into the station, he requests for some specific program; the system returns by providing the URI of a streaming server, where the user can now listen to the requested program. The process ends when the broadcasting finishes and every used resource is set free.

As depicted in Figure 6, the system is composed by three kinds of services: a database service (DBS), a streaming service (SS), and a data storage service (DSS). These services are managed by three controllers, respectively known as the tuner, the monitor, and the terminator. Each one of them has its own queue to receive and store requests.

Every time that a listener enters the station, he searches for a certain program—that is, he triggers a request on the tuner’s queue. The tuner locates the requested program in the database (DBS), and then the monitor creates a new instance of the streaming service (SS). This SS obtains the recording of the program from the storage (DSS) and starts to stream its contents on a newly created URI. Once the program has finished, the monitor notifies the terminator to destroy the old SS, releasing the associated resources.

Of course, the actual system is more complicated; this paper simplifies the presentation to concentrate on the issues related to evolution, which are our main concern here.

5.2. Brief discussion on the example

These services are conceived as software services ‘in the cloud’ within an archetypical cloud platform. This means that all these services (our DBS, SS, and DSS) are implemented and exported using the SaaS approach, where users access them as clients of a service. However, as already noted, using a service-oriented approach is not enough. In fact, these services are usually designed almost in the same way as conventional applications: they are cloud applications mostly because of where they are deployed—they are scalable and resilient because this is supported by the underlying platform. But this platform is also service oriented, and it is presented in the form of an infrastructure as a service’ (IaaS). In summary, our application is defined as a set of user-level (SaaS) services, which are in turn supported by a set of system-level (IaaS) services.

Figure 6. Radio station system at runtime: version 1.0.
Both the ‘canonical cloud architecture’ and our specific example, as presented earlier and in Figure 2, are ‘mixing’ both kinds of services, so the architecture might seem a bit complex. Of course, they are just services: our architecture is just a set of interacting services—and having a well-defined workflow, which is the case, that makes it quite simple.

For the sake of clarity, a short explanation is introduced in the following. Our user-level services (DBS, SS, and DSS) are SaaS services; every one of them runs over a lower-level IaaS service. Our controllers (tuner, monitor, and terminator) are effectively controlling these underlying services and how they provide resources to the top-level ones: thus, they have to be considered IaaS themselves, but their function defines our application’s scalability. Finally, the different queues (T, M, and TR) are services themselves—in fact, instances of the same service. Their existence is almost mandatory for cloud architectures: elasticity implies that the number of clients of a service, at a given moment, can exceed the capability of any service. Queues are therefore provided to ensure that no request (or response) becomes lost. Queuing services work effectively as the ‘connectors’ in this architecture, and obviously, it is safe to classify them as IaaS, as they serve as the basic infrastructure.

Therefore, at the infrastructure (IaaS) level, we just need a storage service, a computing service, a generic database service, and a queuing service, with their corresponding controllers, to respectively allocate our DSS, many SS instances, the DBS, and all the queues at the user level. When the popular Amazon AWS platform is used, for instance, they would have been managed by S3 storage, EC2 computing, the SimpleDB service, and SQS queues [55]—our SaaS services would run on top of these.

Of course, there might be more elements—for example, the billing subsystem has been deliberately omitted. However, it must be present in any cloud-based application: every service in the cloud costs money. Of course, we would assume its function, that is, the customer still is charged, but it has not been included in the architecture to simplify the presentation.

5.3. Detecting the need: excessive dynamic allocation

Let us suppose that the presented system is satisfactory in terms of efficiency and functionality: the radio station works as expected, and the user experience is positive. The chosen program is broadcast, and performance is right, even during occasional (and sudden) peaks of audience.

However, after some time, it is clear that the system is too expensive: elasticity costs money. A new listener implies an access to the DBS, a new computational instance of the SS, and one or several loadings from the DSS. Each one of these steps is chargeable and gets included in the bill. But this also means that if the station achieves success, it will have many listeners, who will cause also many expenses. Usually, our income is expected to cover these expenses, but sometimes, this is not the case—for many reasons. Therefore, we are in a curious situation. From a technical point of view, the system can grow as much as desired: the elastic environment guarantees that there are no scalability issues. But the growth rate can still be a problem, in this case, from the business point of view.

This way of working, however, is considered standard in current cloud architectures, probably because of the low prices of current cloud providers. However, it is obvious that it is not very efficient: every time a new listener accesses the station, a new instance of the SS must be created. In terms of functionality, that is the perfect solution, but it is obviously a waste of resources. In fact, this argument is also found in the well-known performance anti-pattern, excessive dynamic allocation [56]. This anti-pattern criticizes the practice of creating a new instance of an object or server to provide a service to a new client—even comparing this (in the best known metaphor for the pattern) with creating a new gas pump in a gas station every time that a new car requests service. The anti-pattern was originally in the context of Web systems and therefore has a number of similarities to our situation.

This need for the system to evolve can only be detected by a human—but examining the AK structure is also of great help. Sometimes, this process can even be partially supported by automatic tools. For instance, part of our previous work in the Morpheus toolkit [57] adds the capability to assist in the detection of anti-patterns (like the one mentioned earlier) by checking the network of relationships in the AK. In terms of our proposal, this means to decide upon the evolution condition. As we noted, this condition can be expressed in logical terms or using plain text. In our example,
it is easy to quantify that condition (i.e., the cost rises over a certain limit), and it is also simple to relate this rise of the costs to a performance problem. Therefore, to explore performance anti-patterns, such as excessive dynamic allocation, should be a logical choice on the part of the architect.

Once the architect has detected the evolution condition, an evolution process begins. Then, an evolutionary step is planned. The first time, this step and the associated decision will be performed manually; but it will be also captured as part of an architectural pattern—and ultimately, it will be available for future reuse as part of a larger evolution style, as described in the next section.

5.4. Describing the solution: defining an evolution style

In this subsection, we will describe the reasoning we use to describe the evolution of the architecture. This rationale is going to be explicitly captured as part of the AK, including the evolutionary steps—and therefore, it will be used to build an evolution pattern as defined in Section 3.2.

To simplify the presentation, we have chosen the same graphical format to that in Figure 2. In fact, they are very similar; but the reader must have in mind that this version describes actual changes in the cloud architecture, whereas the previous one just provided generic terms for any decision.

First, the evolution condition, requiring the architecture to evolve, is detected (Section 5.3). The first time this condition is evaluated, it leads to an evolution decision, that is, the EvolutionAsset EA.1 in Figure 7. The architect considers, as a first alternative, to use the queue and serve the requests with a first come, first serve policy—so listeners have to wait to be served. This choice defines an alternate branch, that is, a new DesignAsset DA.1, and it is captured within the AK structure as a standard ADD. This ADD provides a new configuration (AA.1), which implements the request queues.

Therefore, the decision EA.1 causes an evolutionary step, where the architecture evolves from the initial version described in Section 5.1 to the alternative configuration AA.1. However, it is soon obvious that this solution is not satisfactory—in terms of the evolution condition, it is possible that costs have diminished; but an excessive response time would cause the loss of listeners.

And then, the evolution process continues. A new evolution asset (EA.2) must be taken, and the first consequence is that ED.1 changes, now rejecting (i.e., inhibits) the previous choice, AB.1.

The new decision EA.2 leads the architect to consider a different alternative: the corresponding ADD will be labeled as DA.2. Now, the architecture (AA.2) to be considered uses a bounded number of instances of the SS service. It provides a fixed cost, apparently fixing our problem, but it has the negative outcome of limiting the number of simultaneous listeners—which causes, in the end, the same problems that the AB.1 decision already had. Therefore, this second evolutionary steps does not provide the final solution.

The architect considers now a different approach (EA.3). Many users would be interested exactly in the same program, and even at the same time: therefore, instead of creating a specific server every time, we consider sharing the same server instance to serve all these listeners. This alternative (DA.3) is therefore our next choice—implying that the previous one (DA.2) becomes inhibited and the

Figure 7. Final decision tree for the application of the evolution style.
evolutionary step towards a new architecture (AA.3). However, the shared instance would have a limited capability, as we are not using the full power of the elastic environment anymore. Then, we would only be able to serve a limited number of listeners, and the situation becomes similar to what happened in DA.2. Therefore, this solution is also rejected, and the evolution process continues. However, this time we are already close to a satisfactory solution.

Finally, the architect considers a related solution (EA.4): to use a relay system. It is still true, as considered earlier, that many listeners would like to access the same program. Then, instead of connecting all of them to the same server, we provide additional servers. For this, we need a fourth kind of service, which was not present in the original architecture, the 
relay service (RS). This service simply connects to the live stream of some existing SS and starts broadcasting exactly the same contents it is receiving, using its own URI. An RS can safely serve a certain number of clients, and therefore, the workload is distributed among several instances of these services. Therefore, this leads us to a new design decision, in which the selected branch (DA.4) relies on the relay approach to fix the problem. As we will see later, this defines the final step in this specific evolution, ultimately leading to the final architecture (Figure 8).

This process is summarized in Figure 7. As we already indicated in Section 3.2, this figure does not depict the actual evolution pattern—instead of that, it describes the structure of the AK (i.e., the decision tree) resulting from its application. But it is clear enough to serve also as a representation of the evolution pattern we have just defined. This can be also an evolution style, if this is the only pattern considered; in fact, this is just a matter of scale.

We must take into account that it does not actually matter if these evolutionary steps are actually performed, providing changes in the architecture, or if they are just considered as (detailed) alternatives during the system’s design. In both cases, they must be included as part of our AK; even rejected solutions are important, as they need not to be considered again. This is exactly what ‘unrepresented design knowledge’ meant, and it is the main reason for capturing AK.

In summary, the process begins with a single evolution decision (EA.1), which leads to other decisions (EA.2–4), and every time suggests an alternative solution (DA.1–3). This defines a logical chain of decisions leading from the initial question (EA.1) to the final choice (EA.4). Every decision defines an evolutionary step: every alternate branch would propose a different architecture (AA.1–3), so if a different decision had been made, the result would have been quite different. But following our logical chain of decisions (all of them related), we end up accepting the final solution (DA.4), which is the result of a sequence of evolutionary steps: and this is exactly our definition of an evolution pattern.

An evolution style, as we already defined (3.2) will be a set of such evolution patterns, which would have a common goal. The particular case of a style with a single pattern is also allowed, so we can also consider this example as a full evolution style, defining the change from Figure 7 to Figure 8.
5.5. Final situation: result of applying the evolution style

Although just tangentially related, two important comments must be made about the chosen solution: first, about the way in which this single choice (‘to use a relay service’) can affect the whole architecture; and second, to explain why this is a sensible solution.

First, note that we need just one SS instance, the original; the first RS just needs to access the stream provided by this instance and begins streaming itself. As more instances of the same program are requested, another instance of the RS could be necessary. This second RS does not need to connect to the original SS, but to the first RS (Figure 8). Thus, the relay system creates a chain of relays, each one able to serve several clients and also the next relay.

So, we have the following: first, the original architecture (Figure 8) has evolved from a client–server style to a hybrid architecture, which includes a peer-to-peer chain of relays (Figure 8); and second, the workflow has to be adapted so that the monitor creates a new RS instance when a new listener needs it and provides the listener with the corresponding URI.

This solution is not an obvious solution: the system still creates many server instances. And the RS also consumes resources, so it also costs money. However, this solution is cheaper: first, fewer instances are created; and second, every instance is also cheaper. The RS is probably less complex than the SS; but even more importantly, the RS need not use the data storage at all, as it obtains data from the other service. Therefore, it is purely computational.

5.6. The resulting evolution style in ATRIUM

As already noted in Section 4.4, there are not many special constructs to describe the evolution style we have just described in ATRIUM, once the notion of evolution decision has been provided. Of course, the basic part of ATRIUM stays unaltered; therefore, we have the standard elements to describe the requirements (a goal model), the architecture (a component model), and the AK. The model of the latter includes all the relevant elements as defined in the ATRIUM meta-model (and as already exposed in Section 4.4), independent of their relationship to the evolution process. Therefore, every design decision will be captured in a DesignAsset including both the decision and its rationale, directly related to the implementation provided as an aggregate of (one or many) ArchitecturalElements; and to the corresponding requirements in the goal model, as traced through the corresponding operationalization.

Therefore, considering that the rest of the architecture is already provided, for our specific example in Figure 7, the extension in the AK structure caused by our evolution style would include the following:

- Four DesignAssets, one to describe every alternative branch (DA.1–4), apart from those that already existed previously.
- In the minimal case, at least four ArchitecturalElements to describe every alternative configuration (AA.1 to AA.3 and also the final architecture). In fact, every architecture has many of these elements, and several of them (as those describing the DBS, the DSS, and so on) are similar or even identical in the different versions.
- The four EvolutionAssets, one to describe every evolution decision (EA.1–4).
- And of course, the corresponding relationships: three inhibits, from each EA.1–3 to DA.1–3, and four constrains, from every EA to the next one, and from EA.4 to DA.4.

Note that applying this evolution style, once it has been stored, does not require any special provision on the part of ATRIUM—it just has to be able to traverse the decision tree and to generate the corresponding version of the architecture once the relevant operationalization has been selected. Both capabilities are already present in the current version of the Morpheus toolkit; the only issue is that these operations must be applied dynamically, during the system’s runtime.

Please note that ATRIUM covers several stages in the specification phase, from initial requirements to the proto-architecture [15]. Hence, the process itself has little to do with many well-known issues of dynamic systems (such as state transfer), as they would be completely dependent on the implementation technology. For instance, when ATRIUM is used to generate PRISMA components, it would be the PRISMA technology that is in charge of performing these actions.
In summary, the example shows how our management and use of the AK make it possible to decide how to evolve from the original architecture to the final approach—and in the process, how to define an evolution style to reuse this knowledge.

6. CONCLUSIONS AND FURTHER WORK

The SA is a critical driver in the development and evolution of software systems. Further, we claim that architecture knowledge is an equally critical driver. We offer two main conclusions: first, that AK can be considered itself as an evolution driver, in the sense that it provides much information of particular relevance to the evolution process; and second, that much of the evolution process itself can be captured as part of the AK, using evolution concepts at the architectural level.

To simplify the reuse of this knowledge, we also propose a specific structure that captures the information and decisions related to architectural evolution: the evolution style. We describe the structure of this style and the process to define and reuse it. To illustrate the feasibility and usefulness of this approach, we describe in some detail the use of this evolution style in the context of a cloud-based application example and provide a clear perspective of the involved notions.

Future work in this context includes the full integration of this (generic) proposal into a specific approach. At present, we are already including these notions in ATRIUM [57], a process that defines and manages SAs from the requirements phase. Incorporating evolution styles into ATRIUM (and its associated toolset) will be just a matter of extending the AK network to include the new kinds of relationships (those related to evolution) and to provide an additional cycle in its model-driven development process, which would apply evolution styles on top of the current architecture and its associated knowledge.

Further work is related to the evaluation of AKdES. Although we have already examined their practicality thanks to the use of ATRIUM, we are in the process of evaluating it from an empirical point of view. In this context, we are assessing existing proposals, such as DESMET [58], that guides us in the evaluation of AKdES. However, the major concern at this point is that there is no standard method that AKdES can be compared with. This drawback turns this evaluation into a very challenging future work.

ACKNOWLEDGMENTS

This work has been partially supported by grants (PEII09-0054-9581) from the Junta de Comunidades de Castilla-La Mancha and also by research grants (TIN2008-06596-C02-01, TIN2012-31104 and CS2007-022 by Program CONSOLIDER INGENIO 2010) from the Spanish Ministry of Science and Innovation. Professor Perry is supported in part by NSF CISE grants IIS-0438967 and CCF-0820251.

REFERENCES


Copyright © 2012 John Wiley & Sons, Ltd.
3. Evolution Styles: Using Architectural Knowledge as an Evolution Driver

ARCHITECTURAL KNOWLEDGE AS AN EVOLUTION DRIVER

41

46. OMG. Software process engineering metamodel (SPEM), version 1.1 formal/05-01-06. 2005.

AUTHORS’ BIOGRAPHIES:

**Carlos E. Cuesta** is an Associate Professor of Software Engineering at Rey Juan Carlos University, in Madrid (Spain), where he is also the current Academic Secretary (Deputy Dean) at the School of Computer Science and Engineering. Previously, he has held positions as Teaching Assistant (1997–2002), Lecturer (2002–2003), and Associate Professor (2003–2006) at the Department of Computer Science in the University of Valladolid, also in Spain. Since he moved to Rey Juan Carlos University (2006), he has been the Director of the Master and Doctorate Programs on Information Technologies and Computer Systems (2006-2010), the Associate Dean of Academic Affairs, and Head of Studies (2010–2012) of his School. He has been Program Co-Chair of the second Joint meeting of two major conferences in Software Architecture: the Tenth Working IEEE/IFIP Conference on Software Architecture and the Sixth European Conference on Software Architecture (WICSA/ECSA 2012). Previously, he was the Conference Chair for the first edition of the European Conference (ECSA 2007). He is also an active member of the ECSA Steering Committee (continuously since 2006); he also participates in many other Program Committees and editorial efforts. He is a founding member of the VorTIC3 research group at Rey Juan Carlos University and has been previously linked to the ISSI group at Polytechnic University of Valencia (Spain). His main research interests are related to Software Architecture and also include many bindings to Service Orientation, Reflective and Self-Adaptive Systems, Model-Driven Development, Invasive Composition, and Concurrency.

**Elena Navarro** is an Associate Professor of Computer Science at the University of Castilla-La Mancha (Spain). Prior to this position, she worked as a researcher at the Informatics Laboratory of the Agricultural University of Athens (Greece) collaborating in the CHOROCHRONOS project funded by the Training and Mobility of Researchers program of the EU. Previously, she served as a staff member of the Regional Government of Murcia, at the Instituto Murciano de Investigación y Desarrollo Agrario y Alimentario, collaborating in the INTERREG II Project funded by the EU. During her master degree studies at the University of Murcia, she was a holder of several research scholarships funded by the Regional Government of Castilla-La Mancha and the National Government. She obtains her bachelor’s degree and PhD degree at the University of Castilla-La Mancha and her master’s degree at the University of Murcia (Spain). She is currently an active collaborator of the LoUISE group of the University of Castilla-La Mancha. Her current research interests are Requirements Engineering, Software Architecture, Model-Driven Development, and Architectural Knowledge.
Dewayne E. Perry is the Motorola Regents Chair of Software Engineering at The University of Texas at Austin (UT Austin) and the director of the Empirical Software Engineering Laboratory (ESEL). The first half of his computing career was spent as a professional programmer and a consulting software architecture and designer. The next 16 years were spent as a software engineering research MTS at Bell Laboratories in Murray Hill, New Jersey. He has been at UT Austin since 2000. His research interests include empirical studies in software engineering, software architecture, and software development processes. He is particularly interested in the process of transforming requirements into architectures and the creation of dynamic, self-managing, and reconfigurable architectures. He is a member of the ACM SIGSOFT and IEEE Computer Society, has been a coeditor-in-chief of Wiley’s Software Process: Improvement and Practice as well as an associate editor of IEEE Transactions on Software Engineering, and has served as organizing chair, program chair, and program committee member on various software engineering conferences.

Cristina Roda is a software engineer at Vector Corp. She received her MSc in Computer Science from the University of Castilla-La Mancha at the Superior Polytechnic School of Albacete, Spain, in 2012. Previously, she finished her bachelor’s degree in Computer Science at the University of Castilla-La Mancha at the Superior Polytechnic School of Albacete in 2010. Her research interests are in Software Engineering, Model-Driven Development, Software Architecture, and Architectural Knowledge.
Chapter 4

A Comparative Analysis of Linked Data Tools to Support Architectural Knowledge

Citation: Cristina Roda, Elena Navarro, Carlos E. Cuesta, “A Comparative Analysis of Linked Data Tools to Support Architectural Knowledge”, 23rd International Conference on Information Systems Development (ISD 2014), 399-406, 2-4th September, 2014, Varazdin, Croatia

Type of venue: International Conference (CORE A)
A Comparative Analysis of Linked Data Tools to Support Architectural Knowledge

Cristina Roda
cristinarodasanchez@gmail
University of Castilla - La Mancha
Albacete, Spain

Elena Navarro
elena.navarro@uclm.es
University of Castilla - La Mancha
Albacete, Spain

Carlos E. Cuesta
carlos.cuesta@urjc.es
Rey Juan Carlos University
Móstoles, Madrid, Spain

Abstract
Architectural Knowledge (AK) has been an integral part of Software Architecture specification since its original inception, but it has not been explicitly managed until recently. It can be described as a computational structure composed of design decisions and rationales. Recent research emphasizes that availability must be complemented by an effective use of this information. We propose the use of Linked Data techniques to define and manage AK, thus achieving flexible storage and scalable search. Our approach suggests storing the network of decisions in RDF format to be retrieved efficiently by means of SPARQL queries. As a side effect, many different AK structures can be described in this way, which then becomes a general format to describe AK. Using this approach, this work analyses some significant features regarding AK of several Linked Data tools, in order to determine which ones are the best/worst for sharing and reusing AK as Linked Data.

Keywords: Architectural Knowledge, Linked Data, Ontology, RDF, SPARQL, AK Network.

1. Introduction
In recent times, Architectural Knowledge (AK) has become one of the most popular topics in the Software Architecture (SA) field. AK has an organizational perspective, but it can also be described as a computational structure, composed of assets of design knowledge, tracing back to specific requirements and forward towards an implementation: that is, it describes the extended history of the system’s design [10]. Then, from this perspective, AK is composed of architectural elements, requirements, and a number of design assets. When the specification just provided the final architecture, all this information was unrepresented design knowledge [21]; but now the architecture includes this information as part of the rationale.

Although AK was considered as an integral element of the SA specification since the very beginning of the discipline [14], it has been only recently [3] that both researchers and practitioners have become aware of its importance. The importance of AK goes beyond merely documenting the architecture; it is one of the vehicles for ensuring the quality of the software development process. For instance, Bratthall et al. [4] carried out a survey with several subjects from both industry and academia. They concluded that most of the interviewed architects considered that by using the AK they could shorten the time required to perform the change-tasks. Interviewed subjects also concluded that the quality of the results, when they had to predict changes on unknown real-time systems, was better using AK. Ozkaya et al. have also concluded, during their interview study [13], that the difficulties during both the initial phases and the evolution of systems are not only due to the availability
of AK, but depend on its effective use. However, despite these difficulties, interviewed architects have also remarked that they perceive AK to be essential when evolution happens. Feilkas et al. [6] have also carried out a case study on three industrial information systems highlighting the relevance of AK as one of its research questions. Authors have detected that one of the problems in these projects was that developers were not aware of the intended architecture because the AK was not properly described. Moreover, the case study also exposed that both a machine-readable form of the AK and the introduction of automatic analysis techniques are keys to achieve architecture-awareness in the development team.

We believe that a relevant conclusion can be extracted: it is a must to facilitate the management of AK, not only during the development process, but especially during the evolution of systems, in order to provide them with certain quality levels. The proposal presented in this work paves the way to satisfy this need by using a Linked Data (LD) [2] approach. In general, Linked Data [2] is referred to use the Web to create typed links between data from different sources. In this manner, the main purpose of our approach is to provide architects with abilities for both documenting the AK and exploiting it, regardless of the architecture-centric practices already used by their organization.

Therefore, our initial hypothesis is that LD provides a good basis for software architects to describe and manage the AK, and that this technology provides an adequate toolset which can be exploited by them in a flexible way. In this line of research, it is followed the design science research method [7] that involves the design of novel artifacts whose use and performance let improve/understand the behavior of Information Systems. In this work, we have selected some LD tools which seem to be particularly relevant, and examine their features from an AK perspective. Our purpose is to decide (a) whether these features seem to be adequate enough for supporting AK, thereby justifying this approach; and (b) which one of these tools, if any, seems to be more promising from an AK perspective.

Our approach tries to contribute to the emerging Organizational Social Structures (OSS) [18] perspective. An OSS can be seen as a dynamic interplay of people, e.g. stakeholders or developers. From this perspective, we can find two alternatives: Communities of Practice (CoPs), where people share/learn a common practice, and Strategic Communities (SCs), where people share experience and knowledge with the aim of achieving strategic business advantage. Both of them have a common need: the need of sharing knowledge to achieve their goals. This work intends to establish a foundation for them by means of the LD approach.

The remainder of this paper is organized as follows. Section 2 outlines some Linked Data tools that are able to manage AK as Linked Data. All of these tools are afterwards analysed in Section 3, according to different features, in order to find out which ones seem to be the best/worst for sharing and reusing AK in a Linked Data format. Finally, Section 4 presents our conclusions and future work.

2. Linked Data Tools

This section presents some of the most widely used Linked Data tools (a more detailed description about their features, installation, etc., can be found in [16]). Initially, they have been selected with the following requirements: they can handle LD and they must have a wide support. Furthermore, all these tools have been selected because they have support for almost all the features that we have considered in the analysis presented in Section 3. Some of these features are of vital importance to manage AK, such as the SPARQL query language, RDF and OWL schemas, RDF input data format, and RDF/XML as serialization format. In particular, these four features are compulsory in our analysis so that those tools that do not support them will be penalized regarding to their total score.

2.1 Virtuoso

It is a single data server, developed by OpenLink Software, that offers functionality for both a traditional Relational data management server and a Linked Data server [12]. Namely, it provides RDF, Relational and XML data management, document and Linked Data server, as
well as a web application server, among others. In order to use this server with Linked Data, we may use OpenLink Data Spaces (ODS) \cite{11}\cite{12} that allows one to establish and manage data on the web, as an extension of the emerging Semantic Web. It includes the platform ODS Briefcase which enables users to control file access rights, content-based search and metadata. In addition, all resources are exposed as RDF data sets, so that the file server functionality can be exploited by means of the SPARQL query language for the Semantic Web. ODS Briefcase offers several features especially suited for managing LD, such as (i) uploading RDF files and validating their format according to a particular syntax, e.g. XML; (ii) editing these files; (iii) consuming data uploaded to the server by means of SPARQL \cite{11}; and (iv) showing query results in different formats (HTML, XML, JSON, Javascript, NTriples, RDF/XML or spreadsheet). In addition, it is able to define SPARQL queries using a specific graphical representation, and to run them using OpenLink iSPARQL \cite{11}.

2.2 Linked Media Framework (LMF)

LMF \cite{17} is an easy-to-setup application server which packages Semantic Web technologies to offer advanced services. This framework consists of two main elements: LMF Core and LMF Modules. The LMF Core component is a LD server that exposes data following LD Principles. In addition, it also offers a highly configurable Semantic Search service and a SPARQL endpoint. Moreover, some other elements which can be used are \cite{17}: (i) LMF Semantic Search, that offers a highly configurable Semantic Search service; (ii) LMF Linked Data Cache, which implements a cache to the LD Cloud, to be used transparently when querying the contents of the LMF. In the case of querying a local resource that links to a remote resource in the Linked Data structure, the remote resource will be retrieved in the background and cached locally.

2.3 Apache Jena & Fuseki

Apache Jena \cite{19} is a Java framework for building Semantic Web applications, and specially Jena is a Java API for these kind of applications that can be used to create and manipulate RDF graphs. Jena provides a collection of tools and Java libraries to develop semantic web and LD applications, tools and servers. Namely, Jena includes an API for reading, processing and writing RDF data in XML, N-triples and Turtle formats. It has also an ontology API for handling OWL and RDFS ontologies and a rule-based inference engine for reasoning with RDF and OWL data sources. It is able to efficiently store large numbers of RDF triples on disk. It has a query engine compliant with the latest SPARQL specification and a server to allow these RDF data to be published so that they can be used by other applications using a variety of protocols. In addition, it provides constant classes for well-known schemas (RDF, RDFS, RDFa, Dublin Core or OWL) and also has some methods for reading and writing RDF as XML. On the other hand, Fuseki is a SPARQL server that offers services for SPARQL update and file upload to a selected dataset, validators for SPARQL query and update, and for non-RDF/XML formats.

2.4 TopBraid Suite

TopBraid Suite \cite{20} offers semantic technology applicable in several scopes, such as to connect data, systems and infrastructures or to build flexible applications from LD models. All components of the suite work within an open architecture platform built specifically to implement W3C standards for the integration and combination of data obtained from diverse sources. These components are TopBraid Composer, an Eclipse plug-in that provides complete support for developing and managing ontologies and LD; TopBraid Ensemble \cite{20}, a semantic web application assembly toolkit for rapid configuration and delivery of dynamic business applications, suitable to create model-driven applications; and TopBraid Live (TBL), a server to deploy flexible, model-driven applications and dynamic, on-demand integration of data from diverse sources.
2.5 **Sesame**

*Sesame* is an open source Java framework for storing and querying RDF data, similar to Jena (Section 2.3). This framework is fully extensible and configurable with respect to storage mechanisms, inference engines, RDF file formats, query result formats and query languages. The core of the Sesame framework is the *RDF Model API*, which defines how the building blocks of RDF (statements, URIs, blank nodes, literals, graphs and models) are represented. Sesame also provides the *Repository API*, which describes a central access point for Sesame repositories. Its purpose is to give a developer-friendly access point to RDF repositories, offering various methods for querying and updating the data in an easy way. Additionally, Sesame supports the use of SPARQL for querying memory-based and disk-based RDF stores, RDF schema inference engines, as well as explicit support for the most popular RDF file formats and query result formats.

2.6 **Mulgara**

*Mulgara* [9] is a scalable open source RDF datastore written in Java, under the Open Software License 3.0. This tool can be considered akin to a relational database, as the information can be stored and retrieved via a query language. But unlike a relational database, Mulgara is optimized for the storage and retrieval of RDF statements, i.e. subject-predicate-object. Some of its main features are native RDF support, multiple databases per server, a simple SQL-like query language (similar to SPARQL), large storage capacity or low memory requirements. Moreover, Mulgara provides mechanisms for ensuring *reliability* (full transaction support, clustering and store level fail-over, permanent integrity), *connectivity* (using SOAP, Jena, etc.), *manageability* (near zero administration, web based configuration tools) and *scalability* (via XA Triplestore engine) of our system.

2.7 **RedStore**

*RedStore* [8] is a lightweight RDF triplestore written in C, which uses the Redland library, a set of free software libraries that provides support for RDF. It supports, in addition to native persistence and in-memory storage, a variety of storage backend adapters, including MySQL, Postgres and Virtuoso. In native mode, RedStore uses hashtables to store RDF data. Its main features are: SPARQL over HTTP support, a built-in HTTP server, support for a wide range of RDF formats, and a test suite for unit and integration testing.

2.8 **Callimachus**

*Callimachus* [1] is a framework for data-driven applications based on LD. It enables web developers to quickly create web applications based on LD, as they only need a web browser to develop a data-driven application. In addition, Callimachus uses either Sesame (Section 2.5) or Mulgara (Section 2.6) for RDF storage, AliBaba (a RESTful object-RDF library), and a proprietary template-by-example technique to view and edit resources. One of the most interesting features of Callimachus is that it is able to execute queries using RDF itself.

3. **Feature Analysis**

This section presents a detailed analysis of the LD tools from the previous section in order to compare their features and determine what the best ones are. This analysis was performed using the examples of AK networks described in [http://goo.gl/NJD2Ft](http://goo.gl/NJD2Ft) to illustrate the power of LD. Some of these features have been chosen because we consider them of vital importance for solving the following deficiencies which often make even more difficult the exploitation of AK:

i. There is not a standard for representing AK, but multiple approaches.

ii. Not every decision and rationale throughout the lifecycle (the history of AK) is recorded.
iii. There is not a standard language for querying AK.
iv. There is not a scalable solution able to manipulate the historical AK.

Therefore, the features selected from the point of view of managing AK as LD are:

- **Data persistence**: indicates if the LD tool provides persistence for its stored data. It is related to deficiency (ii), as this feature is desirable to access the history of AK over time.
- **Query languages**: identifies the different query languages that can be used to manipulate LD. It is related to deficiencies (iii) and (iv), as we want to store AK and the volume of such information can rapidly increase, it is also desirable to have a query language that allows us to navigate AK efficiently.
- **Supported schemas/vocabularies** by the LD tool, like XML or RDF. It is related to deficiency (i), as we are looking for a standard schema to represent AK, such as RDF.
- **Federated queries**: indicates if the LD tool supports data searching across multiple servers. It is related to deficiencies (iii) and (iv), as AK may be stored in different locations, following a LD approach.
- **Input data formats** supported by the LD tool in order to store and manage LD. It is related to deficiency (i), so that the standard for storing AK could be RDF.
- **Query output formats** provided by the LD tool when a query is executed. It is related to deficiency (iii), as we are looking for a standard query language that provides different output formats when querying AK.
- **RDF serialization formats** supported by the LD tool, like RDF/XML, Turtle, etc. It is related to deficiency (i), as we want to represent our AK with a standard such as RDF.

On the other hand, we also take into account some technical features that are relevant when choosing a software tool. They are not directly related to the AK field and its management as LD, but to the software itself. In this way, features selected from the point of view of software are:

- **Type of tool**: specifies the type of the analyzed LD tool.
- **Interaction UI** (User Interface): indicates whether the LD tool has a friendly UI.
- **License**: informs of the type of license that the LD tool has.
- **Security**: indicates how the LD tool guarantees that the information is always safely stored.
- **SDK**: indicates if the LD tool can be used to create other applications.
- **Complexity**: how complex is the Installation, Start-up and Data management of the tool.

A detailed analysis is presented in [16], providing a thorough analysis of all these features (both from the point of view of managing AK as LD and from the point of view of software) with regard to each tool. Notice that almost all LD tools are able to provide federated queries, as they all support SPARQL 1.1 [22]. This language can be used to express queries across diverse data sources, regardless of whether the data are stored natively as RDF or retrieved as RDF via middleware. **SPARQL 1.1 Federated Query extension** has been created to execute queries distributed over different SPARQL endpoints.

In order to decide which the best/worst LD tools are with regard to the analysed features, the following formula (which maximizes the global contribution) is used:

\[ \text{Total score} = \text{QL1} \cdot \text{SS1} \cdot \text{IDF1} \cdot \text{RSF1} \cdot \left( \text{IUI} + \text{DP} + \text{QL2} + \text{SS2} + \text{FQ} + \text{IDF2} + \text{QOF} + \text{Li} + \text{Sec} + \text{RSF2} + \text{SDKS} + \text{CI} + \text{CSU} + \text{CDM} \right) \]

Each one of these parameters is calculated using the rules presented in Table 1, namely in column **How to score**. Parameters QL1, SS1, IDF1 and RSF1 are compulsory features, and therefore their values will be either 1 or 0, depending on their support. The reasons to consider them as necessary features are the following:

- QL1 establishes that the tool supports at least SPARQL as a query language, due to the fact that it is the standard query language used in LD.
- SS1 determines if the tool supports RDF, because this is the standard representation schema accepted for LD. It also determines that the tool supports OWL, which is widely
used for reasoning in the context of LD, because it enables to connect different datasets, located in different data stores, using different schemas. Moreover, both RDF and OWL can help architects to obtain more complete answers for queries over LD, as stated in [15].

- IDF1 establishes that the tool provides RDF as input data format, as it is the standard input data format for LD.
- RSF1 indicates that the tool provides RDF/XML as a RDF serialization format, as it has been defined by the W3C in the original specification.

Initially, we did not consider these four compulsory parameters within the formula, but only the sum of the remaining ones. Finally, we noticed that this was a wrong choice, given that some tools obtained better scores than others, despite they lacked an adequate support for LD, e.g. lack of schemas for RDF and OWL. This led us to mark them as compulsory. This way, the tool with the highest score must be considered as the best analysed LD tool and conversely, the one with the lowest score will be considered the worst. Notice that the remaining features are still considered equally important; therefore they are equally scored, namely 1 point per each feature at most –so the score of each feature ranges between 0 and 1. Moreover, in order to facilitate this normalization, some of these features, such as QL2 or SS2, are calculated by dividing the number of supported query languages or schemas by the number of query languages or schemas offered by the tool with the highest support.

Table 2 shows the marks for each LD tool. Notice that we have omitted the feature that is

Table 1. Features preferences.

<table>
<thead>
<tr>
<th>Analysed feature</th>
<th>Parameter</th>
<th>Preferences</th>
<th>How to score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of tool</td>
<td>-</td>
<td>(Without preferences)</td>
<td>(Not scoring, only informative)</td>
</tr>
<tr>
<td>Interaction UI</td>
<td>IUI</td>
<td>A User Interface that allows users to interact with the Linked Data tool.</td>
<td>1 If the tool has a UI 0 Otherwise</td>
</tr>
<tr>
<td>Data persistence</td>
<td>DP</td>
<td>A platform that provides data persistence over the time.</td>
<td>1 If the tool provides data persistence 0 Otherwise</td>
</tr>
<tr>
<td>Query languages*</td>
<td>QL1</td>
<td>A tool that supports, at least, SPARQL.</td>
<td>1 If the tool supports SPARQL 0 Otherwise (Number of supported query languages)/5</td>
</tr>
<tr>
<td></td>
<td>QL2</td>
<td>The more query languages supported, the more preferable.</td>
<td></td>
</tr>
<tr>
<td>Supported schemas/vocabularies*</td>
<td>SS1</td>
<td>A tool that supports, at least, RDF and OWL as schemas.</td>
<td>1 If the tool supports RDF and OWL as schemas 0 Otherwise (Number of supported schemas)/13</td>
</tr>
<tr>
<td></td>
<td>SS2</td>
<td>The more schemas are supported, the more preferable.</td>
<td></td>
</tr>
<tr>
<td>Federated queries</td>
<td>FQ</td>
<td>A tool that supports federated queries.</td>
<td>1 If the tool provides federated queries 0 Otherwise</td>
</tr>
<tr>
<td>Input data formats*</td>
<td>IDF1</td>
<td>A tool that supports, at least, RDF as an input data format.</td>
<td>1 If the tool supports RDF as an input data format 0 Otherwise (Number of input data formats)/10</td>
</tr>
<tr>
<td></td>
<td>IDF2</td>
<td>The more input data formats supported, the more preferable.</td>
<td></td>
</tr>
<tr>
<td>Query output formats</td>
<td>QOF</td>
<td>The more query output formats supported, the more preferable.</td>
<td>(Number of query output formats)/17</td>
</tr>
<tr>
<td>License</td>
<td>Li</td>
<td>A tool with an Open Software license.</td>
<td>1 If the tool has an Open Software license 0 Otherwise</td>
</tr>
<tr>
<td>Security</td>
<td>Sec</td>
<td>A tool with security services.</td>
<td>1 If the tool has security services 0 Otherwise</td>
</tr>
<tr>
<td>RDF serialization formats*</td>
<td>RSF1</td>
<td>A tool that supports, at least, RDF/XML as a RDF serialization format.</td>
<td>1 If the tool supports RDF/XML as a RDF serialization format 0 Otherwise (Number of RDF serialization formats)/10</td>
</tr>
<tr>
<td></td>
<td>RSF2</td>
<td>The more RDF serialization formats are supported, the more preferable.</td>
<td></td>
</tr>
<tr>
<td>SDK support</td>
<td>SDKS</td>
<td>A tool that provides SDK support.</td>
<td>1 If the tool has SDK support 0 Otherwise</td>
</tr>
<tr>
<td>Complexity</td>
<td>CI</td>
<td>It is desirable a Low complexity in all cases.</td>
<td>0.2 Low 0.1 Medium 0 High</td>
</tr>
<tr>
<td>Start-up</td>
<td>CSU</td>
<td></td>
<td>0.3 Low 0.15 Medium 0 High</td>
</tr>
<tr>
<td>Data management</td>
<td>CDM</td>
<td></td>
<td>0.5 Low 0.25 Medium 0 High</td>
</tr>
</tbody>
</table>

(*) These features are compulsory.
not scoring, i.e. Type of tool. As a result, the first position is for Sesame (9.35), the second is for Apache Jena and Fuseki (8.82), then Virtuoso (8.58), Callimachus (7.3), TopBraid Suite (6.98) and Mulgara (5.43). The last ones are RedStore (0) and Linked Media Framework (0), which have no score because they do not support OWL (a compulsory feature represented as parameter SS1). As we can see, the best LD tools, between the analysed ones and according to our ranking, are Sesame and Apache Jena & Fuseki. In this sense, these two tools are really similar, given that both provide a Java framework to manage LD. Their strength with respect to the others comes from the large number of supported schemas and vocabularies and RDF serialization formats, in addition to the SDK support, which is exclusive of them.

4. Conclusions and Future Work

As already indicated, AK provides the basis to guide architects in many decisive processes, such as evolution [5], and to achieve certain levels of quality during these processes. However, as mentioned in Section 3, there are several shortcomings that can prevent organizations from exploiting AK, such as the variety of formats for documenting and representing it, or the difficulties in meeting its requirements for evolution.

In this paper, we suggest the use of LD techniques to solve these shortcomings. These techniques allow one to define and query AK in an easy and effective way, providing a flexible storage and scalable search. In this sense, our approach recommends storing the network of decisions using RDF, to be efficiently retrieved by means of SPARQL queries. It is worth noting that this proposal does not depend on any particular AK tool or model, or on any RDF triplestore or serialization format: we are considering the benefits of the LD approach itself. Furthermore, we have presented several tools for managing LD and we have analysed some relevant features of these tools with regard to AK, with the purpose of deciding to which extent they provide relevant features, and which ones are the best/worst for sharing and reusing AK as LD. Our results show that the best LD tools for this specific goal, among the analysed ones, are Sesame and Apache Jena & Fuseki. Therefore, given these outcomes, we conclude that the best choice to handle and share AK is to use a LD tool with some SDK support, such as Sesame or Apache Jena & Fuseki, to build some specific tool or interface which simplifies the use of this tool in an AK context. In summary, the LD approach offers important advantages in terms of scalability, as it has been defined to manage great amounts of data thanks to its index-based structure. This facility is being widely used within the open data initiative, to make data available to everyone. This approach makes possible to define and implement a shared repository of AK available to every architect, without compromising specific approaches.

### Table 2. Scores of Linked Data tools.

<table>
<thead>
<tr>
<th>Analysed features</th>
<th>Par.</th>
<th>Virtuoso</th>
<th>LMF</th>
<th>Apache Jena &amp; Fuseki</th>
<th>TopBraid Suite</th>
<th>Sesame</th>
<th>Mulgara</th>
<th>RedStore</th>
<th>Callimachus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interaction UI</td>
<td>UI</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Data persistence</td>
<td>DP</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Query languages</td>
<td>QL1</td>
<td>5/5=1</td>
<td>1/5=0.2</td>
<td>2/5=0.4</td>
<td>1/5=0.2</td>
<td>2/5=0.4</td>
<td>1/5=0.2</td>
<td>2/5=0.4</td>
<td>3/5=0.6</td>
</tr>
<tr>
<td>Supported schemas/vocabularies</td>
<td>SS1</td>
<td>1/3=0.31</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Federated queries</td>
<td>PQ</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Input data formats</td>
<td>IDF1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Query output formats</td>
<td>QOF</td>
<td>7/17=0.41</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>License</td>
<td>Li</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Security</td>
<td>Sec</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>RDF serialization formats</td>
<td>RSF1</td>
<td>1/0.5</td>
<td>5/10=0.5</td>
<td>8/10=0.8</td>
<td>4/10=0.4</td>
<td>9/10=0.9</td>
<td>3/10=0.3</td>
<td>10/10=1</td>
<td>3/10=0.3</td>
</tr>
<tr>
<td>SDK support</td>
<td>SDKS</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CI</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.1</td>
<td>0.2</td>
<td>0</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>CSU</td>
<td>0.3</td>
<td>0.3</td>
<td>0.15</td>
<td>0.3</td>
<td>0.15</td>
<td>0.3</td>
<td>0.15</td>
<td>0.3</td>
<td>0.3</td>
</tr>
<tr>
<td>CDM</td>
<td>0.5</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td>TOTAL SCORE</td>
<td>8.58</td>
<td>6.82</td>
<td>9.35</td>
<td>5.43</td>
<td>0</td>
<td>7.3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Several ideas outline the path for our future work, due to the possibilities offered by LD. One of them is related to the social part of the proposal. Even though OSS are already gaining a growing attention, and practitioners are becoming more aware of the need of sharing the knowledge, we are planning to carry out several case studies in order to evaluate the degree of acceptance of the ideas presented in this work.

Acknowledgements

This work has been funded in part by the Spanish Ministry of Economy and Competiveness under the National R&D&I Program, within Projects DESACO (TIN2008-06596-C02-01), and CoMobility (TIN2012-31104), and also through the FPU scholarship (FPU12/04962).

References

Chapter 5
Towards Considering Quality of Adaptation in ISATINE

Citation: Cristina Roda, Víctor López-Jaquero, Francisco Montero, “Towards Considering Quality of Adaptation in ISATINE”, XV International Conference on Human Computer Interaction (Interacción 2014), 1-3, 10-12th September, 2014, Puerto de la Cruz, Tenerife, Spain

Type of venue: International Conference
Towards Considering Quality of Adaptation in ISATINE

Cristina Roda
LoUISE Research Group, I3A
University of Castilla-La Mancha
Campus, s/n, Albacete, Spain
+34 967 599 200 +2656
cristinarodasanchez@gmail.com

Victor López-Jaquero
LoUISE Research Group, I3A
University of Castilla-La Mancha
Campus, s/n, Albacete, Spain
+34 967 599 200 +2461
victor@dsi.uclm.es

Francisco Montero
LoUISE Research Group, I3A
University of Castilla-La Mancha
Campus, s/n, Albacete, Spain
+34 967 599 200 +2468
fmontero@dsi.uclm.es

INTRODUCTION

The increasing use of applications that make usual tasks easier for the user such as buying airline or theatre tickets, or the variety of people that use a particular application, are some examples that illustrate the necessity for the creation of software applications suitable for any type of user, any type of platform, and able to work under any situation. Unfortunately, it is impossible to design and foresee every situation given during interaction. Here is where adaptation comes to play.

Adaptation to the surrounding context of use of interaction has become a daily routine in the interaction with our devices, i.e. the brightness of our screen is automatically adapted to current lightning conditions, the screen is switched off when we take the phone close to our cheek to talk, and the user interface is maintained. The user interface is the part of our screen that is automatically adapted to current lightning conditions. When we take the phone close to our cheek to talk, the user interface is automatically adapted to current lightning conditions. The screen is automatically adapted to current lightning conditions when we take the phone close to our cheek to talk. Unfortunately, it is impossible to design and foresee every situation given during interaction. Here is where adaptation comes to play.

In this sense, there are some adaptation frameworks that guide the designer in order to produce a pleasant adaptation experience. The most widely-accepted adaptation process is the one proposed by Dieterich in 1993 [1], but it suffers from several shortcomings, considering the most relevant one for us that quality of adaptation is not contemplated at all, i.e. that adaptation process does not allow assessing the adaptation in order to know whether it has been appropriate or not, since it is only focused in the execution of the adaptation.

In this manner, in previous works, we proposed a new framework for adaptation, called ISATINE [2], which addresses some shortcomings found in Dieterich’s process, namely taking into account quality during the adaptation process. Therefore, ISATINE provides a certain level of quality that guarantees an appropriate and accurate adaptation for a given situation. This type of quality is called Quality of Adaptation (hereafter: QoA).

Points of view and types of quality

In order to consider QoA within the ISATINE framework, we should know first what quality is, thus we present some definitions of this usual, but subjective, term. For example, the ones proposed by Crosby [3], Juran [4] or Taguchi [5], as well as more formal definitions such as the one presented by the IEEE in its standard 610.12 [6], or the one within the ISO 9000 family of standards [7].

Apart from this, it is worth noting that quality depends on each user and their business goals. For example (adapted from ISO/IEC 9126-1:2001), a user could identify quality with usability and quality in use; an acquirer with external measures of functionality, reliability, usability, efficiency and quality in use; a maintainer with maintainability; a person responsible for implementing with portability; a developer with internal measures of any characteristics of quality; and finally an evaluator can associate quality to productivity, satisfaction and effectiveness. Apart from these six different points of view for quality, in [8] it is suggested that quality can be described from four different perspectives, i.e. four different types of quality:

- **Expected Quality**, the degree of quality that the quality expert wants to achieve for the final version of the SUS. It is derived from the Expected Quality, so that it is related to the specification too.

- **Wished Quality**, the degree of quality that the client or user needs, i.e. the relevant elements or requirements in the specification of the System under Study (SUS).

- **Achieved Quality**, the quality obtained for a given implementation of the SUS. Ideally, it must satisfy the Wished Quality.

- **Perceived Quality**, the perception of the results by the client or user once the SUS has been delivered.

At this point, we establish relationships between the different points of view about quality and the four types of quality in order to know which points of view attend to each type of quality:

- **Expected Quality** can be treated from the user point of view as it is referred to system requirements (user’s needs), i.e. as it meets the user’s needs. This kind of quality is also related to the acquirer view as it is also conformable to the client needs.

- **Wished Quality** can be treated from the maintainer, person responsible for implementing, developer and evaluator points of view as they represent quality experts who want to achieve final quality regarding to maintainability, portability, internal measures of quality and effectiveness of the system, respectively.

- **Achieved Quality**, as wished quality, can be also treated from the maintainer, person responsible for implementing, developer and evaluator points of view as they represent quality experts who finally achieve quality regarding to maintainability, portability, internal measures of quality and effectiveness of the system, respectively.

- **Perceived Quality**, as expected quality, can be treated from the user and acquirer points of view as it is focused on product quality after the system’s delivery, i.e. the perception of quality by the final user/acquirer. Usability, quality in use, functionality or user experience are concepts associated to this sort of quality.

These six points of view about quality can be also regrouped into two well distinct points of view: **User – Interaction** and **Developer – Adaptation**. **User – Interaction** point of view includes the user and acquirer points of view, while **Developer – Adaptation** point of view includes the maintainer, person responsible for implementing, developer and evaluator points of view.
QoA in ISATINE

In previous work [9], we have defined Quality of Adaptation as the extent to which a set of adaptations produce a user interface to achieve specified goals with usability in a specified context of use. Once we have defined QoA, we can redefine the four types of quality within the context of adaptation:

- **Expected quality of adaptation**, the adaptation goals and requirements that the application is expected to provide.
- **Wished quality of adaptation**, the specification of adaptation and the selection policies to choose between some adaptations in order to achieve a certain degree of quality derived from the expected QoA.
- **Achieved quality of adaptation**, the quality actually obtained for a given implementation of an adaptation and its rationale interpretation.
- **Perceived quality of adaptation**, the quality of adaptation as perceived by either the user or the system.

As we have defined these four types of QoA, it is interesting to relate these types to each stage of ISATINE framework in order to clarify what sort of quality is considered within each phase:

1. **Goals for user interface adaptation** (Goals). This stage is related to expected quality of adaptation as it represents what is globally expected from the adaptation by either the user, the system or a third party. For example, it can be required to maintain the last offer for marketing reasons even if the UI is adapted.

2. **Initiative for adaptation** (Initiative). This stage is related to expected quality of adaptation given that, at this point, the actors involved in the initiation of the adaptation (user, system or third party) express a need for starting the adaptation process because the current state of the user interface does not meet their goals of adaptation in the given context of use.

3. **Specification of adaptation** (Specification). In this phase, the adaptations, which can be carried out to achieve the expected quality of adaptation, are described. Therefore this stage is related to wished quality of adaptation.

4. **Application of adaptation** (Application). This stage is associated with wished quality as the application of the adaptations specified in the previous stage must also be done according to the wished quality of adaptation.

5. **Transition with adaptation** (Transition). It is obvious that the transition from the original user interface to the adapted one must be done considering some criteria of achieved quality as the transition has to fulfill some quality criteria. In this stage, the actor in charge of making the transition should show what adaptation quality has been achieved and how to highlight it to help the actors in resuming their tasks prior to adaptation easily. Therefore, this stage is related to achieved quality of adaptation.

6. **Interpretation of adaptation** (Interpretation). Following transition with adaptation, in this stage an explanation about what is the quality of the adaptation achieved is provided by any actor involved in the adaptation. As a result, this stage is related to the achieved quality of adaptation, since it is what the actors involved in the stage should interpret/explain.

7. **Evaluation of adaptation** (Evaluation). In this phase, it is checked whether the goals of adaptation initially specified are met or not, what can be interpreted as the perceived quality of the adaptation that the final user has. Therefore, this stage is related to perceived quality of adaptation.

Table 1 summarizes in what stages of ISATINE each type of quality is involved in, as well as the two general quality points of view that they are connected to.

<table>
<thead>
<tr>
<th>Quality</th>
<th>Expected</th>
<th>Wished</th>
<th>Achieved</th>
<th>Perceived</th>
<th>Views</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>User - Inter.</td>
<td>Goals</td>
<td>Initiation</td>
<td>Specification</td>
<td>Application</td>
<td>Transition</td>
<td>Interpretation</td>
</tr>
</tbody>
</table>

**CONCLUSIONS**

In this work, the consideration of quality in an adaptation process has been described. Quality of adaptation has been refined to better understand how this term applies to the several stages an adaptation process encompasses. Throughout the adaptation process, how quality of adaptation can be considered has been presented. By doing so, we will be able in the future to propose what techniques or metrics best fit each adaptation process stage, contributing to a better understanding about what QoA is and how it can be actually used in a full adaptation process. Also as future work we aim at providing a complete adaptation quality model that will guide any adaptation designer thoroughly during the design process.

**REFERENCES**


Chapter 6
Towards the Characterization of Interaction Quality

Citation: Cristina Roda, Víctor López-Jaquero, Francisco Montero, “Towards the Characterization of Interaction Quality”, XVI International Conference on Human Computer Interaction (Interacción 2015), 1-2, 7-9th September, 2015, Vilanova i la Geltrú, Barcelona, Spain

Type of venue: International Conference
Towards the Characterization of Interaction Quality

Cristina Roda  
LoUISE Research Group, I3A  
University of Castilla-La Mancha  
Campus, s/n, Albacete, Spain  
+34 967 599 200 +2671  
cristina.roda@uclm.es

Victor López-Jaquero  
LoUISE Research Group, I3A  
University of Castilla-La Mancha  
Campus, s/n, Albacete, Spain  
+34 967 599 200 +2461  
victor@dsi.uclm.es

Francisco Montero  
LoUISE Research Group, I3A  
University of Castilla-La Mancha  
Campus, s/n, Albacete, Spain  
+34 967 599 200 +2468  
fmontero@dsi.uclm.es

ABSTRACT

Interaction Quality is an important issue for any kind of application, and of course the design of the user interface adaptation facilities of an application should consider quality aspects too. Neglecting quality assessment can easily produce applications that do not meet users nor designers expectations. In this sense, previous works have illustrated how to consider quality in a user interface adaptation framework, such as ISATINE, i.e. how to contemplate Quality of Adaptation in each stage of this framework. The many facets of the concept of Quality of Adaptation have been analyzed, and it has also been proposed its decomposition into four types of interaction quality: Expected Quality of adaptation, Wished Quality of adaptation, Achieved Quality of adaptation, and Perceived Quality of adaptation. Furthermore, which of these types of quality are involved in every stage of ISATINE has been also described. However, what is the idea both users and developers have about Interaction Quality? Have they use the same terminology to refer to it? These are the questions that we attempt to answer with this work which presents a study about the characterization of Interaction Quality, namely about Expected and Wished Quality. Thus the aim of this work is to better understand the quality preferences of users, allowing developers to design software products much more attractive for them.

Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation]: User Interfaces – evaluation/methodology, screen design, user-centered design.

General Terms

Design, Human Factors, Theory.

Keywords

User Interface, Interaction Quality, Adaptation Process, Quality of Adaptation, ISATINE, Study

INTRODUCTION

In the last few years, we have witnessed the increasing use of applications that make usual tasks easier for the user, such as buying airline or theatre tickets, or the variety of people that use a particular application. These examples, among others, illustrate the necessity of the creation of software applications suitable for any type of user, any type of platform, and able to work under any situation. However, it is not possible to design and foresee every situation given during interaction. Here is where adaptation comes to play in order to develop such type of context-aware applications, e.g. when the screen brightness is automatically adapted to current lightning conditions or when the user interface is rearranged when we rotate the device. In this manner, different adaptation frameworks/processes to guide designers in the definition of adaptation capabilities have been proposed, such as [1][2]. Nevertheless, in the context of adaptation, the most widely-accepted adaptation process is the one proposed by Dieterich [3], but it suffers from several shortcomings, considering the most relevant one for us that this adaptation process does not allow assessing the adaptation in order to know if it has been appropriate or not, since it is only focused on the adaptation execution. With the aim of addressing these deficiencies found in Dieterich’s adaptation process, in previous work [4], we have proposed another one: ISATINE. Obviously, all adaptation frameworks/processes should provide a certain level of Interaction Quality (i.e. capability of software product to satisfy stated and implied needs when used under specified conditions [5]). This feature of quality guarantees an appropriate and accurate adaptation for a given situation. In the same way that a user interface (and a software application) is created to achieve a specific interaction quality degree, it must happen with the adaptation capabilities of the application. This last type of quality is related to the Quality of Adaptation (hereafter: QoA) and it is referred to the extent to (or increment) which a set of adaptations produces a user interface to achieve specified goals with usability in a specified context of use [4]. However, as seen in previous work [6], currently quality is not considered appropriately in the adaptation process, and thus a good adaptation experience is not always provided to the user. Hence, it has been proposed the integration of QoA into ISATINE, aiming at taking into account Interaction Quality throughout all the stages of the adaptation framework. This consideration of quality will help in improving the adaptation experience at every step of the adaptation process, and not only after the execution of the adaptation, as it usually happens. With this aim, next several types of Interaction quality, regarding QoA, are identified, as well as which of them are involved in each stage of ISATINE. This will allow us to better contextualize the study presented at the end of this work.

INTERACTION QUALITY

There are different types of Interaction quality: Expected quality, Wished quality, Achieved quality and Perceived quality. Regarding QoA, these different perspectives can be defined more specifically as follows:

- **Expected QoA**: adaptation goals and requirements the application is supposed to provide. It is related to ISATINE stages 1 (Goals) and 2 (Initiative).

- **Wished QoA**: adaptation specification and selection policies to choose between some adaptations to achieve a certain degree of quality derived from the Expected QoA. It is related to ISATINE stages 3 (Specification) and 4 (Application).

- **Achieved QoA**: quality actually obtained for a given implementation of an adaptation and its rationale interpretation. It is related to ISATINE stages 5 (Transition) and 6 (Interpretation).
Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation

— *Perceived QoA*, quality of adaptation as perceived by either the user or the system. It is related to ISATINE stage 7 (Evaluation).

All these Interaction quality types are all related to each other. Expected and Wished types of quality are complementary ones. A software product will have quality when Expected and Wished qualities are highly overlapped. Then, this high level of Interaction quality will be perceived by users when Wished and Achieved quality are focused in the achievement of user’s needs. In this sense, it could be interesting to perform a first study, presented in the next section, which shows if there are meaningful differences between Expected and Wished quality in terms of characterization and naming, as we suspect that it could be a clear gap between the perception of people with and without knowledge on Interaction quality.

**CHARACTERIZING INTERACTION QUALITY: A STUDY**

This study identifies a first gap between Interaction quality for users and developers. Next, two dimensions of Interaction quality in ISATINE are analyzed: Expected quality and Wished quality.

**Expected quality**

A total of 31 people without specific experience in computer science completed this study. Those people were recruited between new students of the first course in the Computer Science grade. Approximately 84% of the respondents were male (26) while the remaining 16% were female (5). Respondents’ ages ranged from 19 to 25 (Mean = 19.87, Standard Deviation = 1.41). Results were analyzed in terms of Interaction quality characterization and preferences. A single question was done to participants: *Please, could you write, at least, five words (positive or negative) related to your idea of Interaction quality?* The answers were analyzed by calculating the percentage of repeated words, presented by using a Tag Cloud (see Fig. 1), which indicates the word usage frequency within textual content. The frequency of each word is represented in the tag cloud by increasing the font size and color saturation of that word.

**Wished quality**

In parallel with the previous study, another one was conducted with 24 people with specific experience in HCI and computer science. Those people were recruited among interaction experts and developers. Next, two dimensions of Interaction quality characterization depending on the amount of knowledge in HCI and computer science. Those people were recruited among interaction experts and developers. Then, this high level of Interaction quality will be perceived by users when Wished and Achieved quality are focused in the achievement of user’s needs. In this sense, it could be interesting to perform a first study, presented in the next section, which shows if there are meaningful differences between Expected and Wished quality in terms of characterization and naming, as we suspect that it could be a clear gap between the perception of people with and without knowledge on Interaction quality.

**CONCLUSIONS**

A gap between the definition of *Expected* and *Wished* quality in the ISATINE framework was identified, studied and analyzed by using a survey. This study identifies the different terms that both persons knowledgeable and not knowledgeable in interaction use to describe quality. Finally, it is shown that there is a gap between the views that both groups express about the notion of Interaction quality. So, we must consider this gap in the understanding of quality to provide meaningful information about QoA at every stage of ISATINE. Many of the results in this study are consistent with previous findings in the literature in terms of users and developers. However, the current study provides more detailed information about Interaction quality characterization (criteria). Future research should examine a wider range of people, as well as the design features which make certain kinds of adaptations more accepted than others, i.e. those adaptations with better QoA.

**REFERENCES**


Chapter 7

Contextualizing Tasks in Tele-Rehabilitation Systems for Older People

Citation: Arturo C. Rodríguez, Cristina Roda, Pascual González, Elena Navarro, “Contextualizing Tasks in Tele-Rehabilitation Systems for Older People”. 7th International Work-Conference on Ambient Assisted Living. ICT-Based Solutions in Real Life Situations, (IWAAL 2015), 29–41, 1-4th December, 2015, Puerto Varas, Chile

Type of venue: International Conference
Contextualizing Tasks in Tele-Rehabilitation Systems for Older People

Arturo C. Rodriguez¹, Cristina Roda¹, Pascual González², and Elena Navarro²
¹ Computer Science Research Institute (I3A), Albacete, Spain
{Arturo.Rodriguez,Cristina.Roda}@uclm.es
² Computing Systems Department, University of Castilla-La Mancha, Albacete, Spain
pgonzalez@dsi.uclm.es, elena.navarro@uclm.es

Abstract. Nowadays, one of the most important issues in developed countries is the progressive aging of the population. Thus, governments are irrevocably forced to invest more and more money to take care of their citizens. Regarding healthcare, the attention is focused on those aspects derived from the physical and cognitive problems associated to older adults. Fortunately, fields of research, such as Gerontechnology, are showing promising results for improving quality of life of older people. These works have given rise to remarkable advances in tele-rehabilitation due to the appearance of new technologies and a better understanding of users and their diseases. However, to optimize the development process of these new systems and to take into account user’s features and the surrounding environment, existing modeling languages must evolve. Tele-rehabilitation systems cannot behave in the same way with every user and under every condition, rather they must be able to adapt themselves to the user needs, according to the condition of the environment. In this work, a context meta-model is presented which allows analysts to specify users’ features, devices and the environment, as well as relevant states for the system. Moreover, the relationship between context and task model is also addressed by a CSRML-based task meta-model.

Keywords: Gerontechnology · Tele-rehabilitation · Task model · Context model · CSRML · MDA · Context data state · Task contextualization

1 Introduction

The increase in average life expectancy and the low birth rate in developed countries are causing a progressive aging of population. For instance, a study [1] of the National Institute of Demographic Studies of France shows how the pyramid of population have evolved towards a more uniform distribution in the last 100 years, which implies an older population growth. The European Commission Information Society and Media [2] has also highlighted this problem and warns that the situation is becoming unsustainable. The demographic changes and the increase of chronic diseases force healthcare systems around Europe to make a higher investment.

© Springer International Publishing Switzerland 2015
I. Cleland et al. (Eds.): IWAAL 2015, LNCS 9455, pp. 29–41, 2015.
DOI: 10.1007/978-3-319-26410-3_4
New fields of research, such as Gerontechnology [3], can be a part of the solution to this problem. Gerontechnology aims to develop products, environments and services for improving quality of life of older people in terms of physical and cognitive problems and providing them with new opportunities in their personal life.

A direct consequence of this field is the advance in tele-rehabilitation systems. Tele-rehabilitation can be defined as [4] “the application of telecommunication, remote sensing and operation technologies, and computing technologies to assist with the provision of medical rehabilitation services at a distance”. However, these advances must be accompanied of improvements in the development process. The definition of new languages or the modification of existing ones are becoming a real need when talking about modeling such systems and taking these models as part of the development process.

It is of vital importance to take into consideration context information while modelling a tele-rehabilitation system for older people at design-time. Monitoring context information allows a system to be aware of changes in (i) the environment that surrounds it (e.g. changes regarding illuminance or noise levels); (ii) the devices that are integrated into it (e.g. the temperature of a device); and (iii) the users (e.g. their heart rate, EEG or corporal temperature) that interact with it.

Interaction between older people and computers is sensible to all these changes, thus a tele-rehabilitation system should be able to self-adapt, taking into account these changes, in order to facilitate a friendly interaction. Furthermore, systems that involve physical rehabilitation for older people should guarantee an adequate healthcare service. In this sense, it would be highly recommended to provide these systems with mechanisms for adapting their behavior according to, for instance, the heart rate of the patient. This is one of the most critical parameters regarding older adults because, as Palatini et al. state [5], an elevated heart rate is a risk factor for cardiovascular death in older men. This work focuses in context modeling and how the context affects to the tasks performance. As a result, a context meta-model and an extension of a Task meta-model, based on CSRML (Collaborative Systems Requirements Modelling Language), are proposed. After this introduction, Sect. 2 presents a review of previous work in context modeling, its relation with task modeling and a new proposal of context meta-model. Then, in Sect. 3, a Case Study is presented to support our proposal. Finally, conclusions and future work are described in Sect. 4.

2 Towards a New Context Meta-Model for Contextualizing Tasks

Tele-rehabilitation services are considered a good example of the new-look systems that make use of emerging technologies, such as gesture or voice recognition. This kind of systems requires many context information. Therefore, suitable languages that allow analysts and designers to define and model this information are becoming critical. Dey [6] defines context as “any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves”. Dey’s definition is widely accepted, although Soylu et al. [7] find it too
generic. Taking into account Dey’s definition of context, it can be stated that a *context model* represents the information of the context that surrounds the system as well as of the system itself.

Over the last few years, numerous studies about context modeling have been proposed and, in spite of the variety of context interpretations and how they are modeled, some common aspects arose from them, namely *environment*, *devices* and *user*. For example, an initial proposal [8] focused on concerns, like Location and the ability to detect nearby devices. On the other hand, Rodden et al. [9] included new concepts to context modeling, identifying five dimensions of context: Infrastructure context, Application context, System context, Location context and Physical context. However, all these proposals do not offer a language to model the context. UsiXML [10] tackles the context modeling as part of a model-driven user interface development process. It proposes a meta-model that is instantiated for specifying the entities that are relevant for the system at hand, as well as their capabilities and their interaction zones. However, UsiXML does not provide support for modeling dynamic context information (i.e. information that changes at runtime). On the contrary, the aim of this context meta-model is to describe the structural context information of the system. Another context meta-model was proposed by Vale and Hammoudi [11] that was applied to context-aware development within a model-driven development process. This meta-model is based on ontological concepts and describes the context at two abstraction levels that correspond to M2 and M1 layers within Model-Driven Architecture (MDA). Despite these two levels of abstraction, context information in M1 is restricted to four concepts: Time, Location, Device and Profile. The use of just four concepts makes the description of context too specific, hampering a more detailed specification of other dimensions related to the user or the environment. Moreover, it also ignores the actual relationship between dynamic pieces of information and devices needed to measure them. This is an important issue that Schmidt had already stated in [12]. Hence, most of the existing proposals for modeling context are influenced by the type of application for which they have been defined because of the difficulties encountered while attempting to standardize a meta-model that should cover all possible dimensions of context.

As mentioned above, some concepts as *environment*, *user* and *device* are recurrent in the literature. They can be considered as *context elements* in a context model. Moreover, in many cases, these elements are modeled by describing their *capabilities* or set of features. However, Tele-rehabilitation systems must be aware of context information that changes at runtime. This makes necessary to include the following concepts: (i) *context data*, that represent features of a *context element* that changes dynamically and (ii) *state* of a context data, that represent current value, condition or range of values within which the context data is. In order to address this problem, in this work a context meta-model is proposed, shown in Fig. 1, whose main elements are the following:

- **ContextModel.** Represents an instance of the context model.
- **ContextElement.** It is used to specify an element whose description is relevant for the system. It can be a *User*, a *Device* or the *Environment*. 
• **Capability.** It is used to specify a set of features related to a context element. For instance, GPS capability of a mobile Device, vital signs of a User, or ambient conditions. The features that define a Capability are called **ContextData.**

• **ContextData.** It represents features of the context that change at runtime, and this change directly affects the system itself. For instance, the value associated to the feature Heart Rate, regarding the Capability Vital Signs for a concrete User, should change over time as the heartbeat of the user is constantly changing. This dynamic information is not directly related to Literals, but to States.

• **State.** It is employed to establish a value, range of values or condition of dynamic data. When the current value (measured at runtime) of the dynamic data matches that State (i.e. it matches a concrete value or condition) or it falls within a range of values, the system behavior will be affected at runtime. This concept is similar to the **Input Range** defined by Schmidt [12].

• **AtomicState.** It is used to denote a condition that cannot be decomposed. An atomic condition is always related to a concrete ContextData. For instance, the ContextData “Heart Rate” may be related to a range or condition relevant to the system, such as GREATER_THAN 180 bpm.

• **CompositeState.** It is used to specify complex conditions that are composed of two or more Conditions related by means of logical operators. For instance, LOWER_THAN 180 bpm AND GREATER_THAN 120 bpm.

![Proposed context meta-model](image-url)
Sensoring. It is to specify relationships between context information and the device that measures it. By means of the RefreshTime attribute, it can be also specified the frequency of measurement.

Literal. It is used to specify types, values and measurements for specific context data (e.g. IntLiteral: 180 bpm).

2.1 Contextualizing Task Models

As aforesaid, the context meta-model shown in Fig. 1 enables analysts to specify the agents that are a part of the system and interact with it, their attributes, and the conditions of those attributes that will affect the system behavior. Once these conditions have been specified, a relationship between tasks and context conditions may be established in order to identify what context conditions may affect the triggering of a concrete task.

As aforementioned, in the older people tele-rehabilitation field, a detailed description of user is crucial, as well as the specification of the environment conditions that affect the interaction tasks. Moreover, the context also directly influences how tasks are carried out. While reviewing literature, we may find out classical task meta-models, as for instance ConcurTaskTrees (CTT) [13]. It is used to specify the tasks of a system independently of both the platform and the interaction modality. CTT allows analysts to hierarchically specify the system tasks and the temporal relationships among them. However, in later steps of the development lifecycle, it may be desirable to contextualize the task model (i.e. to add context information to the task model, such as the context situations that may trigger a concrete task or the agents that perform it) in order to determine how these tasks have to be carried out and how the system should react under specific context conditions from the interaction point of view. For example, a tele-rehabilitation system may decide to initiate a system task (e.g. Reduce Exercise Intensity) according to a high heart rate of the user, or change how it provides the user with feedback (e.g. change from acoustic to haptic interaction modality in a noisy environment). Therefore, a contextualized task model may give rise to more precise interaction models within a model-driven development process.

Context information must be measured by sensors or devices that make up the system but, at the same time, become actors with a concrete role that may need to communicate, collaborate or coordinate between them and the system in order to carry out some interaction tasks. In a context-aware system, sensors can be seen as agents that interact with the system and other agents to perform concrete tasks. Hence, two main conclusions can be drawn:

1. It is necessary to model these interaction tasks taking into account the agents involved.
2. It is necessary to differentiate individual tasks (i.e. those tasks that are carried out by a single agent, human or not, that interacts with the system) from collaborative/ coordination/communication tasks (i.e. those tasks that need more than one agent to be carried out while interacting with the system).
This leads us to conclude that the task taxonomy used in the classic task meta-models is not expressive enough, and thus new categories are needed in order to address these two conclusions. A more recent proposal, namely CSRML [14], incorporates new concepts to specify, within its Task Refinement Diagrams, collaboration, coordination and communication tasks of agents and system. CSRML is an extension of the $i^*$ framework [15] for the specification of collaborative systems, extending the meaning of some of its elements. For instance, in CSRML, an Actor playing a Role can participate in individual or collaborative tasks and can be responsible for the accomplishment of a particular goal. Therefore, CSRML provides support to the special requirements of CSCW (Computer Supported Cooperative Work) systems. For this reason, this work presents an extension of the CSRML facilities for task modelling. As can be observed in Fig. 2, a new meta-class called ContextDataState has been included. This meta-class enables analysts to establish a relationship with a condition of the context model. Note that Fig. 2 shows a partial view of the extension of the CSRML task meta-model. Some concepts that are not related to the aim of this paper have been omitted for better understanding. A task can be related to several context conditions which means that the task will start if the context matches with the definition of any of the context conditions. In the following section, a case study is presented in order to show how a tele-rehabilitation system can be modeled using the meta-models proposed in this section.

**Fig. 2.** Partial view of CSRML task meta-model

### 3 Case Study: Contextualized Tasks in Tele-Rehabilitation Systems for Older People

In this section, we present a case study of a Tele-rehabilitation system for older people in order to show how tasks can be contextualized. The tele-rehabilitation system to be modeled is currently under development by the LoUISE research group [16]. This
system allows therapists to create customized rehabilitation exercises for each specific older adult in order to provide him/her a bespoke therapy. At the same time, the system can be aware of context and adapt itself to user needs at runtime. In this case study, we are going to model the Upper Limb Rehabilitation (ULR) task, which refers to rehabilitation exercises that involve the upper extremities of older people’s body. Notice that this task is modeled using the basic elements offered by CSRML, as well as the additional ones for contextualizing tasks.

Figure 3 shows the CSRML Responsibility Diagram (RD) of our case study. This diagram is used to specify the actors that are going to be involved in the system; the roles each actor may assume; and finally, the role responsible for the accomplishment of each task. Figure 3 illustrates that the ULR task is specified as an abstract task (i.e. a task that can be decomposed into more concrete tasks). The RD also depicts that there are six actors involved: User, Computer, Noise sensor, Illuminance sensor, Heart Rate sensor, and Skin Conductance sensor. The User actor plays the Patient role in the Tele-rehabilitation system; the Computer actor plays the UI Adaptor and Virtual Therapist roles, depending on the task it is performing; the Noise sensor actor acts as the Environment Noise sensor role; the Illuminance sensor actor has the Environment Illuminance sensor role; the Heart Rate sensor actor plays the Heart Rate stress sensor role; and finally, the Skin Conductance sensor actor plays the Skin Conductance stress sensor role. Furthermore, this figure also displays that the Virtual Therapist role is the head of the ULR task, so it is in charge of assuring an adequate realization of the ULR task.

![Fig. 3. Responsibility diagram of the Upper Limb Rehabilitation task, using CSRML](image)

Once the roles related to each system actor have been defined, the abstract ULR task can be described in more detail, showing all the tasks that make it up. In CSRML, this is done by means of the Task Refinement Diagram (TRD) which is used to specify tasks, and the roles responsible for performing each task. In addition to abstract tasks, CSRML also distinguishes between non-interaction and interaction tasks. The former are tasks carried out by an agent (human or not) or by the system, without any interaction with other agents or the system. The latter are tasks carried out by agents and the system, involving interaction between them to perform the task. Moreover,
non-interaction tasks can be decomposed into two different types of tasks: user and system tasks, depending on who is involved in them. Thus, if they are performed by a human agent, they are called user tasks (i.e. cognitive tasks, decision processes), otherwise, they are called system tasks (i.e. performed by a non-human agent or by the system itself). Regarding interaction tasks, CSRML defines four types of tasks:

- **Individual task.** It is a task performed by a single actor (not the system).
- **Collaboration task.** It is a task carried out by two or more actors, with different or the same roles, in a collaborative way (i.e. the task could not be done without the explicit participation of each actor). Among these actors, we may find the system and/or agents, both humans and non-humans.
- **Communication task.** It is a task performed by two or more actors, with different or the same roles, to exchange some information. As in the previous case, among these actors, we may find the system and/or agents, both humans and non-humans.
- **Coordination task.** It is a task performed by two or more actors, with different or the same roles, that proceed in a coordinated way. As previously, among these actors, we may find the system and/or agents, both humans and non-humans.

Figure 4 depicts the contextualized TRD regarding the ULR task. This ULR task is decomposed into several subtasks: Start ULR, Execute ULR, Adapt interface and Stop ULR. Except the first one, Start ULR, which is an individual task that involves the patient interacting with the system to indicate his intention of initiating the Upper Limb rehabilitation, the remaining subtasks are also abstract so that they will be explained next, describing how some of them can be contextualized.

### 3.1 Execute ULR

This task is related to the execution of all the exercises including in the Upper Limb rehabilitation and it is also decomposed into three different subtasks, executed in a concurrent way:

1. *Move UL* is an individual task performed by the Patient. It involves he has to move his Upper Limbs while the system is recording his movements;
2. *Monitor stress* is an abstract task that describes the monitoring of the patient’s stress while he is doing rehabilitation movements. For this aim, first, it must be gathered the stress information from Skin Conductance and Heart Rate stress sensors, in a coordinated way; and second, it must be detected the stress level thanks to the collaboration between these type of sensors (they provide the stress information) and the Virtual Therapist which is in charge of interpreting this information from the sensors and determining the corresponding stress level for the patient;
3. *Adapt ULR* is an abstract task performed by the Virtual Therapist. It specifies the adaptation of the ULR exercises to provide a bespoke therapy to the patient by changing the number of repetitions and/or the target area limit. The target area is represented as a rectangle that the Patient’s limbs under rehabilitation should reach while doing a particular rehabilitation exercise.
Regarding the contextualization of this task, the Patient role is associated with the User using the system in order to be aware of his vital signs. This information is relevant to adapt the ULR exercises. Moreover, the roles Skin Conductance stress sensor and Heart Rate stress sensor are related to the Skin Conductance sensor device and the Heart Rate sensor device, respectively. In this sense, the task Adapt ULR has

---

**Fig. 4.** Contextualized Task Refinement Diagram of the *Upper Limb Rehabilitation* task, using CSRML.
been contextualized with three particular context conditions, as seen in Fig. 4: B, C, and D, which will affect the triggering of such task when they become true within the current context. For instance, in a particular moment while the Patient is performing a concrete ULR exercise, his Heart Rate would be 80 bpm (i.e. it is between 60 and 180 bpm) so it satisfies context condition B. Hence the system could decide to launch the Adapt ULR task in order to change the number of repetitions of such exercise, in this case it would be reduced, given that the Patient is becoming a bit stressed.

3.2 Adapt Interface

This task is related to the adaptation of the interface depending on the environment information, namely the interface is adapted with regard to the lighting and the noise. The Adapt interface task is also decomposed into two different subtasks, concurrently executed:

1. Monitor environment is an abstract task to monitor the lighting and noise of the environment where the Patient is performing the ULR exercises. For this aim, first, the system Gathers environment information from Environment Noise and Environment Illuminance sensors, in a coordinated way, and second, it Detects a UI adaptation need thanks to the collaboration between this type of sensors and the UI Adaptor. This is in charge of interpreting this information from the sensors and determining whether it is necessary to adapt the UI.

2. Adapt UI, an abstract task performed by the UI Adaptor. It is defined to adapt the User Interface regarding the lighting and noise levels to provide an adaptive UI by changing the illuminance level (Increasing or Decreasing it) and/or the Interaction Modality (IM) (Activating or Deactivating the acoustic IM).

Regarding the contextualization of this task, the roles Environment Noise sensor and Environment Illuminance sensor are related to the Noise sensor device and the Illuminance sensor device, respectively. Furthermore, there are several subtasks (Increase illuminance level, Decrease illuminance level, Activate acoustic IM and Deactivate acoustic IM) that have been contextualized with four concrete context conditions: F, E, H and G, respectively, as can be seen in Fig. 4.

As in the previous task, these context conditions will determine whether these four tasks, regarding the adaptation of the UI, will be finally done. For instance, while the Patient is performing a concrete ULR exercise, if the lighting level is 60 lux (it satisfies context condition E: Illuminance < 100 lux) and the noise level is 120 db (it satisfies context condition G: Noise > 100 db), then the system could decide to Decrease illuminance level and Deactivate acoustic IM. These tasks are related to context conditions E and G, respectively, to adapt the UI to the current environmental conditions.

3.3 Stop ULR

This can be done by the Patient him/herself (Stop ULR individual task) or by the Virtual Therapist (Cancel ULR system task). With respect to the contextualization of
this task, only one of its subtasks has been contextualized with the context condition A (HeartRate > 180 bpm): Cancel ULR. In this sense, this context condition trigger such task when it is satisfied. For example, while a Patient is performing a concrete ULR exercise, if his Heart Rate becomes 190 bpm, then the context condition A would be satisfied as the value for the Heart Rate is greater than 180. Hence, the system could decide to do the Cancel ULR task in order to stop the exercise since the current condition is becoming too stressful for the Patient.

4 Conclusions and Future Work

In the last few years, new technologies have opened new doors to improve our quality of life. These technologies are especially useful in fields such as Gerontechnology. However, these advances must include new languages that allow us to optimize the development process of Tele-rehabilitation systems. This kind of systems highly rely on the context because they need of the user information, environment conditions and the different agents that participate in doing the tasks. Classic task meta-models lack of a detailed description of concepts, such us collaboration, coordination or communication that are critical in Tele-rehabilitation systems. In order to tackle these issues, a meta-model for specifying dynamic context information has been defined as an extension of the task modeling facilities of CSRML. This extension includes expressive power to specify include relationships with the Context Model so that analysts can contextualize the task models. These relationships enable to define and execute transformations, in a model-driven development process, that can use context information to obtain more accurate interaction models.

A case study has been also presented in this paper to show how our proposal can be used to model context information and how a task model, based on CSRML, can be contextualized. A tele-rehabilitation system has been presented that allows therapists to create rehabilitation exercises to be performed by older people. Concretely, it has been shown how a kind of exercises, ULR ones, can be modeled as a CSRML task. As shown, CSRML provides support for several types of tasks, including those related to the specification of collaborative systems. Furthermore, it has been also seen the contextualization of some tasks with regard to the monitoring of some vital signs of the patient, such as Heart Rate and Skin Conductance, and of some environmental aspects, such as Lighting and Noise. This contextualization of tasks facilitates the achievement of several goals of our system (e.g. provide a bespoke therapy and an adaptive UI). They are needed as the system is designed to be aware of some context conditions that could trigger some tasks at runtime, namely those tasks related to the adaptation of the ULR task to each specific patient and to the adaptation of the UI itself. Moreover, any type of device within our tele-rehabilitation system, such as the Heart Rate sensor, the Skin Conductance sensor, the Illuminance sensor and the Noise sensor can be easily modeled with CSRML by means of actors with a specific role. In this way, the ULR task can be also contextualized in terms of which tasks are performed by which devices. This can be done thanks to the specification of relationships between devices and their corresponding roles. These are in turn related to some particular tasks since the TRD in CSRML provides support for specifying what roles will carry out what tasks.
Therefore, our proposed context meta-model provides useful mechanisms to easily specify context data regarding the user, the environment and the devices (also including those that act as sensors), as well as significant states associated to these context elements that will provoke some changes in the system behavior at runtime. In Tele-rehabilitation, the specification of such states related to, e.g. the patient’s vital signs, is considered a must to provide him/her with a secure therapy, especially for older people.

As future work, one of the next necessary steps is the evaluation of this proposal by analysts and designers within a model-driven development process. This evaluation is currently being designed. On the other hand, there are other context dimensions that may be interesting to include in the context modeling. For instance, the static features regarding different devices, environment and users may be useful for documenting and analyzing the system context of use. Furthermore, the available interaction modalities can be seen as part of the context too. Finally, a tool based on Microsoft DSL-Tools technology is currently under development in order to support these languages.

**Acknowledgements.** This work was partially supported by the Spanish Ministry of Economy and Competitiveness/FEDER under TIN2012-34003 grant, and through the FPU scholarship (FPU12/04962) also from the Spanish Government.

**References**

Chapter 8

Towards an Architecture for a Scalable and Collaborative AmI Environment

Citation: Cristina Roda, Arturo C. Rodríguez, Elena Navarro, Víctor López-Jaquero, Pascual González, “Towards an Architecture for a Scalable and Collaborative AmI Environment”, 14th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS 2016), 311–323, 1-3rd June, 2016, Sevilla, Spain

Type of venue: International Conference
Towards an Architecture for a Scalable and Collaborative AmI Environment

Cristina Roda¹, Arturo Rodríguez¹, Elena Navarro², Víctor López-Jaquero², Pascual González²
¹ Albacete Research Institute of Informatics (I3A), Albacete, Spain
{Cristina.Roda, Arturo.Rodriguez}@uclm.es
² Computing Systems Department, University of Castilla-La Mancha, Albacete, Spain

Abstract. In recent years, much research has focused its attention on Ambient Intelligence (AmI). Its potential applications to smart homes, hospitals, health monitoring or daily life assistance make this paradigm a very promising field of research that can have a great and positive impact in our lives. The combination of AmI environments and Multi-Agent Systems (MAS) has emerged as a perfect solution for the development of this kind of applications. However, there are many challenges to be addressed before such applications can be put into practice. In this paper, we propose an architecture based on MAS aimed to build rehabilitation systems for people with Acquired Brain Injury (ABI) and explain how this architecture has been applied for the development of Vi-SMARt: a system for defining and planning therapies for people with ABI, and to control and evaluate their rehabilitation process.

Keywords: Acquired Brain Injury, Multi-Agent System, Ambient Intelligence, Event-Driven Architecture, SOA, Collaborative Rehabilitation Therapy

1 Introduction

For some time now, Acquired Brain Injury (ABI) has become an epidemic problem in our society with a growing impact every year. ABI has been even named as “the Silent Epidemic”. Toronto ABI network [1] states that people with ABI have suffered “damage to the brain that occurs after birth and which is not related to congenital disorders, developmental disabilities, or processes that progressively damage the brain” but to different causes such as skull-brain trauma, degeneration of the blood vessels, meningitis, brain tumours, etc. People affected with ABI increasingly demand solutions that offer them with a relearning process in order to recover, not only their physical skills, but also their cognitive abilities. In this sense, the inherent characteristics of Ambient Intelligence (AmI), transparency and intelligence, have turned it into one of the best approaches to face up to the impairment derived from ABI.

AmI provides [2] digital environments that proactively, but sensibly, support people in their daily lives. In this context, as Ayala et al. state in [3], AmI systems offer Ambient Assisted Living (AAL) services in an imperceptible way, commonly embedded in
daily devices, such as smartphones, or as sensor devices. As AmI focuses on the development of context-aware systems that integrate different devices to recognize the context and act accordingly, these systems can react rapidly and immediately to needs reflected by people with ABI, especially while they are performing their rehabilitation therapy.

For such people who have suffered a brain injury, rehabilitation is the process of regaining lost skills and abilities to respond to stimuli (visual, auditory and haptic), or developing coping mechanisms to replace those skills. In this context, we have developed a novel AmI system for the rehabilitation of persons with ABI, as part of the Vi-SMARt project (Virtual, Social, Multi-sensorial and Adaptive systems for the Rehabilitation of people with ABI). This system allows therapists to design therapies that can be adapted to each specific patient and his/her abilities. In particular, these therapies not only stimulate the visual sense, but also include other sensory stimuli, mainly auditory and haptic ones. Thus, people with ABI can perform bespoke rehabilitation therapies and follow a suitable schedule according to their diagnostic. Furthermore, these therapies can be performed in different physical places since Vi-SMARt is composed of modules aimed to perform each type of rehabilitation therapy. Vi-SMARt is also intended to make possible a collaborative rehabilitation therapy because, as stated in [4], it facilitates the rehabilitation process. This feature implies an important challenge since the design of different interrelated rehabilitation modules must be addressed.

The usage of Multi-Agent Systems (MAS) in an AmI environment emerges as a natural and perfect combination because agents are reactive, proactive and exhibit an intelligent and autonomous behavior. For this reason, Vi-SMARt is intended to follow a Multi-Agent architecture, whose general overview was previously presented in [5]. Notice that the design of our MAS was initially planned for controlling and managing rehabilitation therapies carried out by older adults, but it has been modified to be used by people with ABI. This architecture makes Vi-SMARt extensible, since new types of ambient variables, devices and functionality can be added to the existing system; adaptable to consider the inter-personal and intra-personal variances of people with ABI; and scalable, so that the performance of the rehabilitation processes is not affected by the number of connected users.

In this paper, we present the different challenges that have emerged during the Vi-SMARt design and how its architecture has been designed to address them. This paper is structured as follows. First, in the next section, we give an overview of some systems that follow different architectural styles used in combination with MAS to support all features demanded by an AmI environment. Section 3 states several challenges that must be solved by Vi-SMARt and its architecture. Section 4 presents the software architecture of Vi-SMARt, describing the architectural styles it includes. Finally, in Section 5, some conclusions and future lines derived from this work are shown.

2 Related Work

If we take a glance at the literature, we can find many works, such as [6][7][8], that have proposed or studied different rehabilitation systems or approaches with the aim
of treating people affected with ABI. The current trend regarding this kind of systems focuses on the use of AmI to support the rehabilitation process, taking into account different variables that can affect both the patient and the environment. In this sense, several studies have been published, such as the one presented in [9]. AmI approach turns rehabilitation systems into context-aware systems thanks to the integration of different devices to recognize the context and act accordingly. By doing so, they are able to rapidly react to the needs shown by people with ABI while they are carrying out their rehabilitation activities.

Furthermore, it has been reported that MAS are particularly good at modeling real-world and social systems, where problems are solved in a concurrent and cooperative way without the need of reaching optimal solutions. This is why the natural relationship between AmI and MAS is being widely exploited. For example, the work done by Gascueña et al. [10] shows how to implement AmI applications using the INGENIAS Development Kit (IDK) to develop them following the MAS approach. IDK supports both the specification of MAS models and a set of facilities for code generation. Another proposal that combines AmI and MAS is the one proposed by Corchado et al. [11]. They have developed an intelligent environment, called GerAmi that integrates MAS and other technologies, such as mobile devices, to facilitate the management of geriatric residences.

Other approaches, in addition to that of MAS, have been used in the development of AAL systems. Becker [12] identifies several ones, such as:

- **Service-Oriented Architecture** (SOA), whose main benefit is to support the separation between the contract and its implementation. In this context, applications in SOA are built on services, where a *service* is [13] an implementation of a well-defined functionality. Such services can then be consumed by clients in different applications or processes, facilitating the *reuse* of such services.

- **Peer-To-Peer** (P2P) [14] is a networking technology that has a serverless operation, enabling several systems to share resources and communicate directly with each other. This approach moves away from the conventional centralized server approaches, avoiding bottle necks and single points of failure. In this sense, a pure P2P system consists of equal peer nodes that lend the same functionality, acting as both clients and servers. Notice that, in heterogeneous systems made up of small sensor nodes up to powerful servers, this architectural style will not be applicable for the whole system but rather for groups of devices of the same type. Moreover, this approach solves a usual shortcoming of centralized approaches: *scalability*, as this is not limited by the features of the server.

- **Event-Driven Architecture** (EDA) follows the publish-subscribe pattern, so that some elements can promptly react to the notifications of events produced by other elements, being an *event* a significant change within our system. Note that, in EDA, the event producers are not aware of either the type or the number of event consumers, that is, a low coupling between them. This considerably contributes to both the *extensibility*, as new consumers can be easily developed, and *scalability* of the systems.
In this sense, there are some related works that follow some of these architectural trends, combined with Multi-Agent. The architectural style most frequently used together with the MAS approach is SOA. For example, in [15] it is presented FUSION® with the aim of facilitating the development of AAL systems, exploiting the use of intelligent agents integrated into SOA platform, so that, most of the characteristics of the system are distributed into remote and local services. Similarly, another work presented in [16] proposes a methodology for integrating a FIPA-compliant agent platform with the Open Services Gateway initiative (OSGi) service oriented framework. In [3], an Agent Platform (AP) called SolAgentPlatform is described, that enables the communication between agents running in Android-based devices and agents running in SunSPOT sensors. This architecture is extensible and partially supports the FIPA approach, making possible the integration of new types of sensors and other kinds of AAL devices in an easy way. Besides, this AP acts as a middleware that provides a set of services to the agents running on it. In [13], a MAS is presented aimed at improving healthcare and assistance to older or dependent people at their homes. A service-oriented platform, called PANGEA, has been created, where all services are implemented as Web services emulating a client-server architecture since this platform includes both a service provider agent and a service consumer one. In [17], Jade is presented, which is a FIPA-compliant implementation of a mobile agent platform able to dynamically add and configure services, implemented as Web services, as in the previous case. It supports the communication between Web services and agents thanks to WSAI (Web Services Agent Integration). Another work [18] presents a Model-Driven Development approach oriented to develop pervasive systems based on agents that follow a Service-Oriented architecture.

Additionally, EDA is also considered a good option when developing MAS. For instance, in [19], it is presented an agent framework that offers policy control for the insertion of plug-ins on-the-fly in a coordinated and proper way. Namely, the components of an agent communicate through an asynchronous event bus with traditional publish/subscribe semantics. Another example is the work presented in [20], a proposal for processing information to detect Activities of Daily Living (ADLs). This proposal is based on EARS (Event-driven Activity Recognition System), a multi-agent activity recognition framework which is in charge of detecting ADLs, following an event-triggered process.

Finally, as Becker states in [12], EDA is considered one of the best approaches to be applied in the development of AAL systems. However, he claims that SOA along with EDA will converge to develop an optimal solution in order to meet the different quality demands derived from AAL systems, i.e. an Event-Driven Service-Oriented Architecture (ED-SOA). Some works that follow an ED-SOA architecture for developing AAL environments can be found in the literature, such as [21][22][23].

Consequently, after analyzing the different approaches and proposals, we have detected that most of them focus on the ADL recognition and support, without considering the special needs of people with ABL. Although these proposals exhibit important benefits, they also present lacks when they are used in isolation. In the following section,

1 Foundation for Intelligent Physical Agents
we describe clearly the main challenges that an architecture must support for the development of Vi-SMARt: a rehabilitation AmI system for people with Acquired Brain Injury. Then, in Section 4, it is described how we have faced such challenges considering the current advances in the area.

3 Challenges to Be Solved by Vi-SMARt

The development of Vi-SMARt has posed important questions that have to be properly addressed to provide people with ABI with a proper solution. Before describing the architecture of Vi-SMARt, we are introducing in the following such challenges in order to understand the rationale underpinning such specification:

- **Collaboration.** The Vi-SMARt project is being developed in cooperation with several associations and hospitals that have a long experience in the rehabilitation of people with ABI. They have highlighted that one of the main problems these people have is isolation as they find difficult to engage their friends and family in their new life. For this reason, therapists working with ABI claim they need new solutions that turn the rehabilitation therapies into a social activity that enables people with ABI to communicate, collaborate and cooperate with their peers. This demand is also justified by different studies, such as [4], that show how other therapists have also highlighted the importance of collaboration with others as a facilitator of the rehabilitation. For this aim, one of the main challenges to be faced is the definition of a solution that enables people with ABI to carry out their rehabilitation by means of virtual rooms where they can collaborate, cooperate and communicate for the achievement of both personal goals and group goals.

- **Adaptability.** One important issue that hampers the definition of a proper specification of the architecture of Vi-SMARt is the fact that the sensor configuration is specific for every deployment on the patients’ side, depending on the sensors and communication devices deployed, the type of sensor data provided to monitor the therapy being carried out such as raw, fused, or already aggregated data. In addition, the evaluation of the therapy at hand is complicated by inter-personal variances so frequent among people with ABI, but also intra-personal along its treatment process, resulting in a diversity of needs and demands. Consequently, the proposed architecture has to ensure a high level of adaptability towards the patients’ surrounding environment and the patients’ needs and constraints.

- **Extensibility.** We just need to have a look at the market to see more and more devices that offer new monitoring facilities, the trendy wearables. These devices can be used by therapists to monitor users’ heart rate, stress, etc., while people with ABI are performing their therapies in order to adapt them to people’s capabilities and status. Moreover, it must be highlighted that, as the brain is the damaged area, people with ABI can suffer different long-term deficits that directly affect their daily lives. These deficits can be classified into four categories: (1) physical impairments related to inability to control part of the body, such as paralysis of one side of the body, motor in coordination, or balance problems; (2) cognitive impairments that directly affect intellectual performance, including attention or memory...
problems; (3) emotional impairments such as depression or adjustment problems; and (4) behavioral impairments related to interaction with their environment, such as irritability and restlessness. Due to both this variability and the new advances in technology, the architecture to be developed must facilitate its extension by adding new devices and therapies at runtime, in order to offer its evolution to changing demands over time.

- **Scalability.** As aforementioned, ABI is known as “the Silent Epidemic” because the number of people with ABI is growing every year, being one of the most extended problems nowadays. Just for instance, according to the Brain Injury Centre [24] Traumatic Brain Injury, one of the causes of ABI, is more common than breast cancer, spinal cord injury, HIV/AIDS, and multiple sclerosis (MS) combined. Just in the United Kingdom it is estimated that at least 1 million people live with long-term effects of brain injury [25]. According to such figures, the number of potential users can become extraordinarily high. Therefore, the architecture must be designed to scale as the number of users requests, so that the rehabilitation process is not affected by the number of users connected or any shortage of resources. Additionally, it must be also considered that not only individual therapies must be offered but also collaborative ones. That is, different virtual rehabilitation rooms can run simultaneously where between two and four people with ABI can collaborate, cooperate or compete to carry out specific tasks of their treatment. Thus, the architecture must be designed to facilitate that connection problems with the server do not affect all this rehabilitation process.

As can be observed, the architecture of Vi-SMARt must be designed to address serious challenges that a traditional client/server approach cannot face. In the following section, a description of such design is presented justifying how it can help to face each one of the described dares.

### 4 Architectural Styles in Vi-SMARt

As aforementioned, important challenges had to be addressed in the development of Vi-SMARt. One of the first steps we have carried out has been the development of an architecture able to address them. As shown in Fig. 1, Vi-SMARt has been structured into three different systems:

- **Therapy Service.** This system has been defined as a data service where patients' data, preferences and results of their treatment are managed. Moreover, all the therapies designed by therapists, schedule, etc., are also managed by this service.
- **Therapy Design Environment.** This system has been identified to provide therapists with facilities to design new therapies, schedule the treatment of patients, monitor their evolution, etc.
- **Therapy Execution Environment.** This system has been defined to provide people with access to their treatment. It integrates different devices, such as, Kinect, Leap Motion, heart-rate sensors, etc., to facilitate the interaction and the monitoring during their treatment.
As can be observed in Fig. 1, the deployment of these systems is carried out by exploiting the following three approaches: MAS, SOA and P2P. In the following section, how the MAS approach has been used in the context of Vi-SMARt is presented.

The Therapy Service has been developed as a SOA service so that Therapy Execution Environments can use it to obtain the therapy to be done by the patient and then work in a disconnected way until he finishes his therapy. Then, the Therapy Execution Environment forwards the results of the execution for its analysis by the therapists. In this way, the scalability of the proposal improves as the connections are reduced.

Another important decision is related to the virtual rehabilitation rooms. As we have described in the previous section, one of the challenges to be addressed is to provide people with ABI with facilities for collaboration while they perform their treatment in order to improve, not only their cognitive and physical impairments, but also their behavioral ones. For this aim, we have used a P2P approach to provide Therapy Execution Environments with a serverless operation that enables people with ABI to communicate, collaborate and cooperate with their peers in order to achieve certain predefined goals while they can be supervised by a therapist. This alternative also improves the scalability of the architecture as the virtual rehabilitation rooms work in an isolated way one of each other, and only need to connect to the server just to download the therapy to be executed and the store the results of the execution.

Finally, the design of the Therapy Execution Environment follows an Event-Driven approach in order to address the expected extensibility. This architectural pattern has been selected in order to facilitate that not only new types of therapies can be easily integrated in Vi-SMARt, but also new devices or sensors.

Fig. 1. Deployment of Vi-SMARt.

In the following section, the different MAS that made up Vi-SMARt and how they have been deployed following the previous infrastructure are described.
4.1 A MAS-Based Architecture for Vi-SMARt

The three systems that comprise Vi-SMARt architecture have been designed as MAS. To illustrate these MAS a general overview created with Prometheus Tool is used. This graphical notation includes the agents involved (in light brown color), the protocols including the messages they exchange (in magenta color), the actions each agent is responsible for (in light green color) and lastly, the percepts that represent the external information arriving to the MAS (in light red color) from the ambient.

Fig. 2. Vi-SMARt: Therapy Design Environment MAS.

In Fig. 2, the multi-agent system architecture created for the Therapy Design Environment is depicted. The aim of this MAS is two-folded. On the one hand, it supports the specialist in creating the therapies, and on the other hand it supports the therapist in managing the patient’s information gathered during the execution of therapies. This MAS includes three agents. PatientManagementAgent is responsible for all the goals related to patient’s data. It is in charge of producing statistics and drawing conclusions to help the specialist in understanding how the patient is evolving. This is very valuable information when it comes to design new therapies or when testing the effectiveness of the therapies designed. This agent also provides relevant data for therapy design to TherapyDesignerAgent.

TherapyDesignerAgent is responsible for supporting the specialist in the creation of therapies. This agent provides help to the specialist in designing the planning of the therapy, that is, in describing what activities should be made, the workflow of these activities and how this planning adapts to the patient’s evolution [26]. This adaptation is specified in terms of a Fuzzy Inference System (FIS). The patient’s information provided by PatientManagementAgent is also used to suggest potential features that can be used in the FIS to customize for the patient the therapies designed.

Lastly, TherapyManagementAgent serves as the interface of the MAS with other systems and with the user. This agent receives the requests from the user and returns a user interface for a therapy based on the patient’s data, according to the data produced by both the TherapyDesignerAgent and the PatientManagementAgent. The user interface is generated in a similar way as in AB-HCI [27].
The Therapy Service MAS (Fig. 3) provides the services required by both the Therapy Design Environment and the Therapy Execution Environment. This design allows for a centralized repository of those services required to design or start a therapy. All the services are made available through TherapyServiceAgent. This agent handles all the requests and communications, and offers a secured interface for the services. A secure interface is required in this case, since there is sensitive information about the patients stored.

Three agents are available to manage the three services provided. TherapyServiceManagerAgent is responsible for storing, organizing and providing the therapies and the steps these therapies require to be performed. PatientDataServiceManagerAgent stores, organizes and provides the patient’s data required to perform a therapy or to design it. Some patient’s data are needed to adapt the therapy to each patient, that is, to address the adaptability claimed in the previous section. This adaptation is achieved by means of the FIS. All the FISs designed for each therapy and step are managed by FIS_ServiceAgent. This agent has two main goals. On the one hand, it stores, organizes and provides the fuzzy sets designed for each therapy or step. On the other hand, it also makes those fuzzy inferences which are global.

Finally, the Therapy Execution Environment (Fig. 4) system provides the infrastructure for a therapy to be executed either individually or in the virtual rehabilitation room as a collaborative therapy. When a patient starts a therapy in the virtual rehabilitation room, this system will be instantiated. The instance will retrieve the therapy to be executed from the Therapy Service. To execute the therapy retrieved, the system will also retrieve the patient’s data required together with the fuzzy sets defined for this therapy and its steps. All these communications are handled by the CommunicationManager-Agent. All the persons executing the collaborative therapy enter the P2P network for the therapy. The first client to enter the P2P network becomes the host for the P2P. All the others will be regular peers. A host is required because some collaborative therapies include adaptation relying on information provided by all the peers, the so called collaborative adaptations. All the communication between the peers is made on the basis
of an Event Driven approach. When the therapy arrives to CommunicationManagerAgent, it will forward this information to TherapyControllerAgent so the therapy can start. CommunicationManagerAgent sends also the Patient’s data required for the therapy to TherapyMonitoringAgent. This information is enriched with the incoming information from the sensors that report about the ambient. One agent is devoted to each sensor to make the sensor network easily extensible. Before starting a therapy, it has to be adapted to the patients’ characteristics. Therefore, the therapy is sent to AdaptationManagerAgent so it is adapted. This agent will produce an adapted therapy that will be started by TherapyControllerAgent. Whenever an adaptation is required, AdaptationManagerAgent will ask SMARTManagerAgent to infer what decisions to take.

Fig. 4. Vi-SMARt: Therapy Execution Environment MAS.

5 Conclusions and Future Work

In this paper, we have presented a novel AmI system for the rehabilitation of people with ABI, as part of the Vi-SMARt project (Virtual, Social, Multi-sensorial and Adaptive systems for the Rehabilitation of people with ABI). This system enables therapists to design therapies adapted to abilities and disabilities of each patient. These therapies can use different senses, such as visual, auditory or haptic ones, to rehabilitate some physical and/or cognitive capabilities of the patient. In addition, to manage the therapy execution process, the therapist can use different types of information. In particular, Vi-SMARt can control environmental information, physiological data of each patient, and other information gathered from the therapy execution. To manage all these features in our AmI system, the use of Multi-Agent Systems has emerged as an adequate solution. However, as we have described in section 3, there are many challenges still to be solved before its application in a final and real system. Therefore, our proposal tries to address all these challenges offering an appropriate architecture that makes Vi-SMARt extensible, adaptable, scalable and collaborative. To achieve these goals, we include in our
solution the MAS, SOA and P2P approaches to manage and establish the interrelationships between our three systems: Therapy Service, Therapy Design Environment and Therapy Execution Environment. The inclusion of SOA allows for working in a disconnected way during the therapy execution. In addition, P2P allows providing the communication between several patients that collaborate in the execution of a specific therapy. Finally, MAS supports the specialist in creating the therapies and in managing the patient’s therapy execution and the information gathered during this process. Our next step is the evaluation of Vi-SMARt in a real environment. To do so, we count on the support of an association that assists people affected by Acquired Brain Injury.

Acknowledgements

This work was partially supported by the Spanish Ministry of Economy and Competitiveness and by the FEDER funds of the EU under the project grant insPire (TIN2012-34003). It has also been funded by the Spanish Ministry of Education, Culture and Sport thanks to the FPU scholarship (FPU12/04962).

References

Chapter 9

An Interactive Fuzzy Inference System for Teletherapy of Older People


Type of venue: International Journal (IF: 1.993, Q2)

This publication endorses this dissertation by compendium of publications.
An Interactive Fuzzy Inference System for Teletherapy of Older People

Arturo C. Rodríguez · Cristina Roda · Francisco Montero · Pascual González · Elena Navarro

Abstract The progressive aging of the population in developed countries is becoming a problem for healthcare systems, which must invest ever higher sums in caring for their older citizens. One of the most important issues in this area involves the physical and cognitive problems associated with growing old. In order to reduce the effect of these problems, gerontechnology has emerged as one of the most promising alternatives, especially in the field of the telerehabilitation systems developed to date. However, most of these systems do not offer therapists the facilities to design therapies adapted to individual patients. This paper proposes a novel system that supplies this need and enables therapists to create bespoke motor therapies as state diagrams and manage them efficiently in a collaborative setting. The proposed system is equipped with a fuzzy-based decision-making component that therapists can use to control transitioning between states according to variables such as fatigue and performance. Therefore, the system makes it feasible to provide older patients with the treatment they need in their own homes while its effectiveness is controlled by a Fuzzy Inference System.

Keywords Gerontechnology · Telerehabilitation · Teletherapy · Fuzzy Inference System · Bespoke therapy

Introduction

One of the greatest achievements of the developed countries has been the increase in average life expectancy in the last 100 years; however, it is also becoming one of their greatest problems, as they must invest ever higher sums in caring for their older citizens [1]. This problem has been highlighted by the European Commission Information Society and Media [2], which has stated that “The situation is becoming unsustainable and will only worsen in the future as chronic diseases and demographic change place additional strains on healthcare systems around Europe.” The European Commission claims that the only available alternative is to deeply reform the way health care is delivered by means of both the exploitation of Information and Telecommunication Technologies (ICT) and the necessary organizational changes. This combined solution will allow preventive and person-centered healthcare systems to be introduced.

Gerontechnology [3] has emerged as an interdisciplinary field of research that can provide both patients and healthcare organizations with the required solution. This field combines gerontology and technology to develop products, environments and services that can (1) prevent, delay or compensate for those physical and cognitive problems related to aging and (2) provide older people with new opportunities in their personal life in terms of leisure,
learning, etc.; i.e., the main goal of gerontechnology is to improve the quality of life of older people.

Bronswijk et al. [4] have suggested that all the research, development and design carried out in the gerontechnology field should be oriented to its end users from its inception, and they have proposed the use of the gerontechnology matrix to focus on the advances in this field and to make the most of them in terms of both their usefulness to the end user and their technological impact. This matrix has two dimensions. The first identifies the application domains (health and self-esteem, housing and daily activities, communication and governance, mobility and transportation, and work and leisure) that represent a challenge in the everyday life of older people. The second dimension is the technology impact (enhancement and satisfaction, prevention and engagement, compensation and assistance, and care and organization) for which a product, environment or service should be designed to satisfy the goals of the applications domains. Telerehabilitation systems can be classified in the health and self-esteem application domain, one of the most critical for improving the quality of life of older people. Telerehabilitation can be defined as “the application of telecommunication, remote sensing and operation technologies, and computing technologies to assist with the provision of medical rehabilitation services at a distance” [5].

In the last decades, different telerehabilitation systems have been developed with different technological impacts from their inception. Initially, as Brennan et al. [6] have stated, the first systems were created as a proof of concept to demonstrate that patients, physically located in remote locations, could be provided with rehabilitation assessment and treatment techniques and monitored by telephone [7]. Since then, as ICT has advanced, new services have been developed. Winters [8] has identified four different categories of remote rehabilitation services:

- **Teleconsultation** is a standard “face-to-face” telemedicine model that uses interactive videoconference between a local provider (patient) and a remote rehabilitation expert.
- **Telehomecare** is defined as service delivery that allows a clinician (usually a nurse or technician) to coordinate services with a low-to-moderate bandwidth interactive connection.
- **Telemonitoring** is the clinical application wherein the rehabilitation provider sets up unobtrusive monitoring or assessment technology for the client (patient).
- **Teletherapy** is defined as a model of telerehabilitation service delivery in which the client conducts therapeutic activities (such as exercise) at home using a therapy remotely managed by a therapist.

*Teletherapy services* can have a high technological impact in terms of care and organization, as they can help to control the costs of therapeutic activities and improve the patients’ quality of life. It is important to highlight that teletherapy does not mean unattended therapy, but on the contrary, it must be designed and monitored by therapists to avoid any risk that can negatively affect the patients. This is especially important when working with older adults as they may be prone to different physical and/or cognitive problems that could constrain the therapy to be applied. This need to personalize therapies not only applies to their design but also when they are being administered. For this reason, any system of teletherapy should control factors such as the patient’s performance and fatigue level, which can have an effect on the treatment’s final effectiveness.

New therapy systems include groundbreaking devices such as Wii, Microsoft Kinect [9–12] and OptiTrack [13], which can be easily managed even by users with age-related cognitive problems. The next section describes several works that use MS Kinect or Wii devices to manage telerehabilitation tasks for physical rehabilitation. However, as pointed out in “Related Work” section, most of these proposals are lacking certain properties that would allow therapists to take full advantage of them. These include a lack of support for designing and monitoring bespoke therapies; i.e., therapies adapted to the user’s real needs, such as his/her performance or muscle fatigue. In order to overcome this deficiency, we propose to exploit Fuzzy Inference Systems (FISs) [14]. Specifically, we propose a therapist-centered system [15] with facilities for designing and executing FIS to make therapies more customizable. This system enables therapists to create bespoke therapies by defining the specific tasks that each patient should carry out for his/her treatment. Therapies are defined as a composition of several interrelated tasks in a specific order, while the user’s postures or actions are managed by MS Kinect. This system is equipped with a fuzzy-based decision-making component that can be used by therapists to control transitions between states according to variables such as fatigue or performance, so that bespoke therapies can be carried out by older people at home while its effectiveness is controlled by the Fuzzy Inference System.

We consider that our proposal has two special advantages: The first is that therapists have been involved in the process of designing the system from the very beginning. The second is that both therapists and patients are usually favorably impressed by the technology, especially when applied in the rehabilitation field [16–19]. For this reason, we designed a natural user interface developed on MS PixelSense [20] to enable therapists to define therapies easily and intuitively and to give them a full feedback on the patients’ rehabilitation by MS Kinect.
This paper is structured as follows. “Related Work” section reviews related work on teletherapy and fuzzy-model-based systems. “Specification of Therapies” section describes our method of specifying therapies. “Fuzzy Inference Systems for Controlling Transitions and Adapting Therapies” section describes the implemented support for Fuzzy Inference Systems. “Case Study: Designing a Bespoke Treatment” section presents a case study of an application of the proposal, and finally “Discussion and Conclusions” section contains a discussion and the main conclusions drawn from this work.

Related Work

As already mentioned, the development of new devices, such as Wii or MS Kinect, has revolutionized teletherapy services [9–12] as they now make it possible to control complex user actions and give similar results to other more expensive devices (OptiTrack [13]). They can also be easily installed and managed by users with age-related cognitive handicaps. Several studies have already been published on using MS Kinect to manage teletherapy tasks for physical rehabilitation. For instance, Oliver et al. [11] propose a system that allows the therapist to define specific physical exercises that can be controlled by MS Kinect. A similar study [21] proposes a virtual Web environment to design rehabilitation postures, also via MS Kinect. However, neither allows elementary tasks to be easily linked with others to create more complex therapies. Another proposal takes the form of a game [22] developed to rehabilitate patients with chronic pain of the lower back and neck. This system uses MS Kinect as input and several biosignal acquisition devices to capture full-body motion while the game is in progress. Finally, a recently published literature review [23] analyzes 69 of the most important proposals for health care that use Wii for motor rehabilitation. One of the conclusions of this study is that one of the potentials that should be explored is the development of new games customized to individual needs. As can be seen, these systems do not usually provide therapists with the possibility of creating their own therapies, even though many motor impairment problems associated with aging can improve with a rehabilitation program specifically designed and monitored by therapists.

Even though these systems do offer certain advantages when performing motor rehabilitation treatment, once a therapy has been designed and implemented, most of the analyzed systems are not able to make decisions about how it should be adapted to the performance and/or physical conditions of the user during the exercise in order to achieve maximum effectiveness. In other words, the analyzed systems cannot adapt the therapy to the real needs of the user.

As aforementioned, an alternative to facilitate this need for bespoke therapies is to exploit the power provided by FISs. These systems have a reasoning mechanism that performs inference operations using fuzzy rules to make decisions under uncertain conditions. They have been widely used to control uncertainty in many domains, such as tourism forecasting [24], applied mathematics [25] and engineering [26]. The physical rehabilitation domain is not unaware of its advantages in dealing with “fuzzy” variables. For instance, when a therapist is working with a patient, he must take muscle fatigue into account. Tseng et al. [27] define muscle fatigue as “a decreased ability to maintain the expected force or power output commonly experienced after exertion of physical power.” One of the main difficulties when dealing with this variable is that there are no clear boundaries between high, medium and low fatigue levels, and they are also affected by other factors such as age or physical fitness. FIS therefore seems to be the natural solution for physical rehabilitation.

FISs have already been used for physical rehabilitation. For instance, Huq et al. [28] developed a system based on fuzzy logic for autonomous post-stroke upper-limb rehabilitation exercise. This system uses its decision-making mechanism to estimate the patient’s muscle fatigue in order to vary the resistive and assistive forces a robotic device gives to the patient as well as the duration of the exercise. Other authors, such as Yang [29], evaluated patient fatigue by means of a FIS with haptic inputs. Other authors have focused only on the patient’s performance. For example, Barghout et al. [30, 31] developed a FIS that quantifies the patient’s performance using haptic data. Zhang et al. [32] also incorporated a FIS to estimate the performance of post-stroke patients. A rehabilitation station in [33] exploits games and a fuzzy system to adapt the play according to the players’ performance. This rehabilitation station has a degree of configurability, as it allows therapists to adapt success and failure criteria to the patient. The system designed by Gopalai et al. [34] uses FIS to estimate the biofeedback given to patients by vibrotactile actuators to improve their postural control during wobble board training. This system determines the quality of postural control using as input the inertial measurement units of both the patient’s trunk and the wobble board. Other authors, instead of designing a rehabilitation system, have defined a methodology [35] for the design of a real-time fuzzy system to control the rehabilitation of patients with upper-limb dysfunction using robotic devices. This system is able to tune both the human–robot interaction and the human–robot interaction dynamics. As can be observed, all these systems focus on just one or two fuzzy variables, such as fatigue or performance. However, they do not allow therapists to select the variables to be used by the FIS and how they should be described in order to design bespoke.
Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation

therapies. As far as we know, the only system that offers some degree of flexibility in defining FIS is the one proposed in Segundo et al. [36]. However, this system was specifically designed for differential diagnosis and so cannot be applied to teletherapy.

From the foregoing therefore three main conclusions can be drawn. Firstly, there are many systems available for controlling patients during motor rehabilitation teletherapy. However, these systems do not offer the therapist with a flexible support to design new therapies or to adapt them to specific patients. Secondly, even though some existing systems provide facilities to control patients during the exercises, they cannot control the carrying out of therapies as complex as those applied by therapists in their daily work [37]. Thirdly, the existing systems that also exploit fuzzy rules for rehabilitation do not offer therapists with facilities to describe FIS necessary for the design of bespoke therapies. In order to address these shortcomings, we have developed a therapist-centered system for them to create their own therapies from scratch by defining each task in individual treatments. Hence, our system is aimed at:

1. Offering therapists a useful tool for designing and applying therapies for older people rehabilitation and monitoring therapies and patients in order to control the rehabilitation process. For this aim, our systems enable therapists to define therapies as a composition of several interrelated tasks in a given order, while the elementary steps associated with a posture or movement are managed by MS Kinect. Moreover, thanks to its natural user interface, therapists can easily define treatments in collaboration with the patient (“Collaborative Specification of Therapies” section) by using tagged objects. These treatments are created by instantiating a proper meta-model (see “A Meta-model for Designing Therapies and Treatments” section, Fig. 1), and a model-to-text engine allows the automatic generation of code for its performance.

2. Offering therapists facilities to design FIS that allows the system to adapt the therapy to the real needs of the user. By using our system, specialists can also define customized FIS, instantiating another proper meta-model (see “Fuzzy Inference Systems for Controlling Transitions and Adapting Therapies” section, Fig. 5) to create bespoke therapies that can automatically self-adapt with regard to variables such as fatigue, performance. These variables control the final execution of the therapy, adapting the flow of activities according to the specific performance and physical conditions of the user while he is doing the exercise.

Specification of Therapies

In a previous work [38], we created a preliminary version of a system designed to address some of the shortcomings mentioned in the previous section. In this paper, we propose an improved and more complete version that allows therapists to define individual therapies for older people, including the rules to be observed when carrying them out. These rules are meant to dynamically adapt the therapy to the patient at runtime, either by counting successes and failures or by updating the difficulty level of the task. Information can also be obtained on the patient’s fatigue level. During the design of this system, two strict constraints were imposed: Firstly, it had to enable a natural interaction, and secondly, the therapies had to be developed by a team of therapists. Microsoft PixelSense [20] was chosen as the environment for the system, since it facilitates collaboration and natural interactions. MS Kinect was also included to facilitate the definition of the physical rehabilitation tasks. The main design features are described in “A Meta-model for Designing Therapies and Treatments” section, and details of the collaborative design environment are given in “Collaborative Specification of Therapies” section.

A Meta-model For Designing Therapies and Treatments

One of the main goals of our proposal is to enable therapists to design bespoke therapies with the aim of providing older people with individualized treatments. For this purpose, we created a meta-model which is flexible enough to let therapists define both the activities themselves and their complex interrelationships (Fig. 1). This system offers therapists a user-friendly interface (see Fig. 4) for creating therapies by instantiating the meta-model plus a model-to-text engine to automatically generate the code necessary for their execution. The same code is also used to adapt therapies to individuals at runtime.

Some of the most important concepts included in the meta-model are Therapy, Activity, Task and Step. A Therapy is defined as a set of activities to rehabilitate a specific physical impairment. Each Activity can be divided into elementary tasks. Each Task is a sequence of Steps, which are user gestures or postures that may be controlled by MS Kinect or may only be informative. Each element
9. An Interactive Fuzzy Inference System for Teletherapy of Older People
can be related to another element of the same type. This relationship, modeled by the Transition meta-class, represents the sequential order of the elements. In order to control transitions between different Steps, Tasks or Activities, there are conditions or rules that must be defined by therapists. These rules represent the knowledge base of the system to decide, at runtime, which is the best transition between the available ones, i.e., which is the most desirable next action a user should make while performing a Step, a Task or an Activity. The system is thus able to adapt the performance of the exercise to the user’s needs by means of an expressiveness factor introduced to define compound states composed by state-machine diagrams and FISs.

As can be observed in Fig. 1, Activities are composed of Tasks and Tasks by Steps; e.g., an activity can be represented as a compound state made up of an ordered sequence of tasks defined by a therapist. The therapist can order tasks by establishing relations (transitions) between tasks (states) and between tasks and steps. The FIS element defined in Fig. 1 is used by the therapists to design the control regarding what will be the next transition between states, i.e., the next task or step the user should perform.

As each StateElement (Activity, Task or Step) can have more than one output transition, its related FIS can be defined to decide the next state according to different parameters, such as user fatigue or performance. The system is also able to adapt other parameters such as the difficulty level of the next task to be performed by the user (patient). This adaptation is controlled by fuzzy rules, previously created by the therapist, thanks to the meta-model presented in “Fuzzy Inference Systems for Controlling Transitions and Adapting Therapies” section. The Variable element was defined for this, as it connects this meta-model with the meta-model used to design the FIS rules. Variable is used to describe parameters that can be monitored by the system in order to control the application of treatments and therapies, e.g., fatigue or stress. The relationship between Variable and Person is established by means of the Value element; i.e., a specific person associates a specific value with a specific variable, e.g., fatigue. It may also be necessary to define other types of variables, such as number of repetitions or level of difficulty, related to a state instead of a person. This relationship between Variable and StateElement is established by means of the Value element, so that a specific StateElement can be associated with a specific value of a specific variable. This makes it possible, for instance, that an activity owns a specific value for number of repetitions variable.

Finally, other important elements of the meta-model are Treatment, Therapist and Person, which allow a therapist to assign a treatment to a concrete person. TemplateTherapy is included in the meta-model so that a therapist can define therapy templates, which can later be customized to the specific needs of a patient by instantiating ConcreteTherapy. As can be observed, TemplateTherapy can be related to several ConcreteTherapy, according to the number of patients doing the therapy. Thus, every treatment assigned to a person will have at least one specific therapy.

In order to illustrate how the meta-model described in Fig. 1 can be used by a therapist to instantiate a given therapy, an example of a therapy model with a single activity was created, as shown in Fig. 2. This model supports the activity depicted in Fig. 3.

Collaborative Specification of Therapies

A user-friendly interface was developed using MS PixelSense [20] to enable therapists to design treatments instantiating the meta-model in both a transparent and collaborative way. StateElements are represented by rectangles, and transitions are lines between these with an arrow pointing in the appropriate direction, indicating the sequential order of the states. All these elements are shown in Figs. 4, 7 and 8. Using the MS PixelSense environment, therapists can create a treatment from scratch using tagged objects; that is, physical objects marked with a dot pattern called tag. Currently, three different types of tagged objects can be used to design a treatment just by dropping them on MS PixelSense: a tagged object to create new states (Activity, Task or Step), another to zoom in on states (i.e., to access elements contained inside a specific state) and the last one to delete states. For example, Fig. 4 shows how a tagged object to create states is placed on MS PixelSense. Therapists can also define transitions between two states simply by tapping and holding the source state and dragging it to the target state.

Treatment is defined as a composite state made up of therapies. These therapies consist of activities, and these activities of tasks; finally, tasks are sequences of steps. Note that, for the sake of readability, the interface only shows one level at a time, i.e., either therapies, activities, tasks or steps. For instance, Fig. 4 shows the tasks that make up the Butterfly Wings Activity. If a therapist needs to know the steps that make up a specific task, she only has to drop the zoom tagged object (with a magnifying glass icon, as shown in Fig. 7) on the task. This zoom can also be done pinching, as in most touch devices. Finally, thanks to the bar on the right of the user interface (see Fig. 4), therapists can navigate throughout the different levels, i.e., treatments, therapies, activities and tasks.

Therapists can also use the interface to define the fuzzy rules that will control transitions between states and consequently the execution of the therapies. These fuzzy rules are edited by filling in tables, related to each state, called fuzzy associative matrices. Each cell of these tables
contains the linguistic value of a variable used in the consequent of a rule. For instance, Fig. 4 illustrates the fuzzy rules of a task.

Finally, the system also uses MS Kinect to define specific physical rehabilitation steps, such as the ones associated with gestures and postures. During the definition of a task belonging to a therapy, the therapists can create its steps by capturing their postures and/or gestures using MS Kinect. In this way, they are able to specify, control and assess how these steps should be performed by patients and their precision level.

Fuzzy Inference Systems for Controlling Transitions and Adapting Therapies

A Fuzzy Inference System (FIS) is a method that uses fuzzy rules, membership functions and fuzzy logic to map an input space into an output space. It also contains a reasoning mechanism (decision-making unit) that performs inference operations on the rules. In this way, a FIS helps the system in which it is incorporated to make decisions in spite of a certain degree of uncertainty. Unlike classical logic, fuzzy logic does not establish rigid boundaries when dealing with a variable (e.g., Number of repetitions is Low when it is 3 or less), but it defines regions in the domain of the variable that can be overlapped and determines the degree of truth of a fact according to the real value. These regions are known as fuzzy sets, and the degree of truth of a fact is the degree of membership with regard to a crisp value of the variable in the fuzzy set referred to in the fact. The degree of membership is calculated by using the Membership Functions that define the fuzzy sets. The fuzzy sets are tagged with linguistic values that correspond to a particular linguistic variable, i.e., a variable that has non-numeric values (linguistic values). Thus, expressions (facts) like Number of repetitions is Low (where Number of repetitions is the linguistic variable and Low is the linguistic value) will not be simply true or false, as in classical logic, but they will have a degree of truth or a degree of
membership of the real value of the variable with regard to the fuzzy set *Low*.

Fuzzy rules are an important concept when defining a FIS. They have an IF–THEN structure and represent the knowledge base used by the inference process. Figure 6 portrays an example of a fuzzy rule and the elements that make it up. These rules are composed of expressions, such as the aforementioned *facts*, and must be built by experts in the application domain. Every rule has an *Antecedent* (the IF-part of the rule) and a *Consequent* (the THEN-part of the rule). The reasoning mechanism of the FIS evaluates these rules for inferring a result, i.e., a final decision of the system. There are several methods of implementing the reasoning mechanism of a FIS. One of the most widely used is Mamdani’s method, which is highly intuitive and similar to human reasoning. According to this method, the outputs of the membership functions of the consequents are fuzzy sets. After evaluating all the rules of the FIS, this method obtains a fuzzy output function that must be defuzzified to obtain a specific value. This process is called *defuzzification*, and there are different algorithms for its implementation [14], such as the max membership principle or the centroid method, both of them widely used. Other inference methods, such as the Takagi–Sugeno method, do not require a defuzzification process, but are less intuitive and more complex. In order to make our
system more intuitive for therapists, we decided to use the Mamdani’s method in the inference process.

Since our system is meant to be used by therapists, it provides support for building FISs; that is, therapists can use it to describe both fuzzy rules and the linguistic variables used by the system to adapt the therapies to individual users. Therapists can decide the sequence of therapies, activities, tasks and steps that subjects have to perform. For this purpose, a meta-model was created to provide therapists with a language that they can use to define variables and fuzzy sets as well as to build fuzzy rules. The diagram of this meta-model is shown in Fig. 5. Note that the concepts of this meta-model are also used throughout the case study described later on in “Defining the FIS: Support for Uncertainty” section.

The main concept in this meta-model is the FIS entity. This element is composed of Rules to represent the fuzzy rules used in Mamdani’s method. A Rule is composed of two expressions: one that represents the antecedent of the fuzzy rule (IF-part of the rule) and another expression for representing the consequent (THEN-part of the rule). The expression of an antecedent can be a simple or complex expression. SimpleExpression represents an atomic fact (e.g., Number of repetitions is Low), and ComplexExpression represents a set of Expressions, which can be either complex or simple, connected by a logic operator (e.g., Number of repetitions is Low AND Fatigue is Medium). A SimpleExpression has two parts: a Variable to be monitored by the system and a Linguistic Value, which corresponds to a FuzzySet defined by means of four values. Note

![Meta-model for defining Fuzzy Inference Systems which provides a language for expressing fuzzy rules](image)

Fig. 5
that the expression of a consequent must always be a SimpleExpression as it represents a single fuzzy output function. Since FISs can be defined in our system to decide the transition the system must choose in a particular state at a given time, a Linguistic Value can be related to a Transition. The Transition element is also part of the treatment meta-model described above (see Fig. 1).

Figure 6 shows an instance of a fuzzy rule. This rule has a ComplexExpression as antecedent that comprises two SimpleExpressions and the LogicOperator AND. The first SimpleExpression is composed of the Variable Number of repetitions and the LinguisticValue Low. The second is composed of the Variable Fatigue and the LinguisticValue Medium. The consequent contains a SimpleExpression composed of the Variable Transition and the LinguisticValue T1.

The therapist can build one or several FISs associated with each StateElement by defining the fuzzy rules of the FIS, the variables that must be monitored, their corresponding linguistic values and the fuzzy sets. These FISs are defined to fulfill two main goals: (1) to control how therapies, activities or tasks are adapted to a specific patient, taking certain features into account, such as the fatigue level; and (2) to decide the next transition in the course of a therapy. For instance, the system could modify the difficulty level of a task taking two characteristics into account: fatigue level while performing the steps of the task and the current difficulty level. The therapist would thus define a FIS composed of rules such as:

IF Difficulty is Medium AND Fatigue is High THEN Difficulty Level is Low

As can be observed, the antecedent of this rule has two variables, Difficulty and Fatigue. Our system already provides support to these variables as follows:

- The Difficulty of a task is calculated as the size of the area in which the patient has to place part of his body to perform the task successfully. The therapist defines these areas by means of the four points of each step that makes up a particular task. The difficulty is established at Task level by means of percentages of the resulting area. The lower the percentage of the area, the greater the difficulty of the task. The height of the area will be modified if the movement of the step is vertical. On the other hand, if the movement is done across the horizontal axis, the width of this area will be modified. Both height and width of the area will be adapted for oblique movements. Figure 14 shows an example of a vertical movement of a Task just after the system had modified the target area.
- Our aim was not designed to provide a way of measuring patient Fatigue, but to use the already existing approaches to provide therapists with the facilities to model these approaches. Initially, we considered using Huq et al.’s methodology [28] to measure Fatigue and giving our system its own methodology in future work. Huq et al. stated that Fatigue is a variable that can be inferred by means of another FIS that uses as input the precision and the average speed of the subject while performing an exercise. They consider muscle fatigue, i.e., the decreased ability to maintain the expected force or power output. Huq et al.'s system assumes that muscle fatigue reduces muscular ability to generate force and twitch fibers. This in turn reduces precision and speed while doing an exercise. In our system, precision is assessed by measuring the average distance of the joints of the virtual skeleton from the target area (see Fig. 13), and average speed is calculated by the time taken by the subject to do a Task, i.e., while performing the different steps of a Task. Huq et al. used previous experiments to define the fuzzy sets of speed and precision. In our proposal, the therapists are in charge of defining these sets, so they can establish an individual FIS to assess fatigue. Finally, it may be desirable to consider a statistical model for each type of patient and thus automatize fatigue assessment, but this is not within the scope of the present study, as the current aim is to offer a system that allows specialists to define customized FISs to create self-adapting bespoke therapies. Variables, such as fatigue, control the final execution of the therapy, adapting the flow of activities while the patient is carrying out an exercise. Using this system, experts in rehabilitation should be able to model fatigue, or any other variable, according to their previous experience.

![Fig. 6 Example of a fuzzy rule, using the FIS meta-model](image-url)
As stated above, transitions between Therapies, Activities and Tasks can also be controlled by FIS systems defined by therapists. For this, the consequents of the rules will be composed of a Variable and a Linguistic Value that will be evaluated for the StateElement under execution. Therapists define rules such as:

IF Fatigue is High AND Repetitions is Medium
THEN Action is T1

The consequent establishes that the T1 transition connected to the StateElement under evaluation will be selected if the antecedent is positively evaluated. The defuzzification process is a FIS whose consequents involve transitions carried out by the middle-of-maxima method, so that the fuzzy set with the highest degree of membership will determine the transition.

Since therapists are in charge of defining a FIS and its corresponding rules, the quality and efficacy of such FIS will depend on their judgment. Although this facility can introduce a bias, it will also make experts more prone to accepting our rehabilitation system, which they could then adapt to their preferences.

**Case Study: Designing a Bespoke Treatment**

In this section, a case study is described to show how our system provides therapists with support for creating treatments from scratch. The treatment used as an example is composed of an exercise aimed at improving flexibility, musculoskeletal function, balance and agility in older adults, adapted from [37]. This particular therapy focuses on the performance of Range-of-Motion (ROM) activities for gently preparing frail older people with special exercise needs. These ROM activities are aimed at exercising all the major joints of the body and can be performed either seated or standing. Table 1 shows these activities, classified into two clearly distinct types: Lower-body and Upper-body ROM activities. For the sake of brevity, the remainder of this section will focus on one ROM activity, Seated Butterfly Wings, which is the seated version of the Upper-body ROM Activity number 8, as shown in Table 1.

**Creating a State Diagram for the Treatment: The Therapist’s Role**

The first step in creating a treatment is to set up a state diagram using the touchable interface developed for MS PixelSense (see “Collaborative Specification of Therapies” section). This activity will be represented as a StateElement that will be part of a therapy which accordingly will contain other ROM activities.

The therapist creates each task of the activity by dropping the tagged object for the creation of states on the MS PixelSense. Figure 7 shows the user interface for creating a Task and its associated tagged object. Once the tasks of the activity are created using this tagged object, it is time to define the different transitions between each task in order to establish their execution flow. These relationships are established by tapping and holding the source state and dragging it to the target state. Every transition and task can be labeled using its corresponding text box. Similarly, in the upper part of the screen, another text box shows the name of the activity, in this case Butterfly Wings.

The application provides a menu on the right side of the screen (see Fig. 8) that displays the parent diagrams of the diagram being edited. This menu has been included to enable therapists to navigate throughout the nested structure that is being created. In this case, the top diagram corresponds to Treatment 1, which contains a single therapy (ROM Therapy). The second diagram corresponds to a therapy with three activities, where the yellow-colored activity is Butterfly Wings, whose elements are displayed in the main working area.

Finally, the application provides a simple editor for establishing the properties of a particular task. This editor is available not only for a Task, but also for the rest of the StateElements and can be accessed by pressing the (+) icon and then expanding the rectangle. For example, as can be seen in Fig. 9, a therapist can fix a Difficulty level of 80 % for the Upward and Downward Movement task involved in Butterfly Wings.

**Defining the FIS: Support for Uncertainty**

Once the states diagram for the Seated Butterfly Wings activity has been created, i.e., all tasks and transitions have been defined, a set of FISs can be defined for each state. For this, the therapist can open the FIS editor by pressing the “+” symbol embedded in each node or state representation, as shown in Fig. 8. This user interface enables therapists (1) to create and manage the FIS of a particular state, (2) to define the fuzzy sets of each variable and (3) to create the set of fuzzy rules of each FIS. This user interface incorporates a wizard that helps therapists to carry out these actions step by step.

When a FIS is created, new fuzzy rules can be defined. In order to facilitate this process, the system lists the variables that can be monitored in two separate groups (Input and Output variables), and the therapists will choose a maximum of two input variables and one output variable. The rules are represented by means of bidimensional matrices or tables.

The therapists can also define the fuzzy sets of each variable. Firstly, the number of fuzzy sets must be
specified. By default, the system creates these fuzzy sets proportionally, and afterwards the therapists can modify them by dragging and dropping the points of the membership functions. For instance, Fig. 10 portrays the user interface for setting the FIS of a given task. In this case, five fuzzy sets must be specified to characterize the fatigue variable.

Once the fuzzy sets of the two input variables and one output variable have been defined, the system displays the corresponding matrix or table. As Fig. 11 shows, each gray cell represents a fuzzy rule that relates two linguistic values of two input variables to a linguistic value of one output variable. That is, the headers of the row and the column of a specific gray cell determine the input linguistic values of the variables (the antecedent of the rule), and the text shown within the gray cell represents the output linguistic value of the variable (the consequent of the rule).
There are also two kinds of matrices: one for representing the AND logical operator and another for the OR logical operator. If the therapist selects only one input variable, a matrix will be shown that will have only the main diagonal enabled. For instance, an AND matrix for the output variable Action and the input variables Fatigue and Repetitions is shown Fig. 11. In the running example, five fuzzy sets were defined for Fatigue (Very Low, Low, Medium, High and Very High), and three for Repetitions (Low, Medium and High). For instance, the gray cell [0, 0] represents the rule IF Fatigue is Very Low AND Repetitions is Low THEN Action is Trans2.

Note that therapists can change the linguistic values of the output variable (in this case, Action) by touching the gray cells. Therefore, the Action variable is used by the system to decide which task must be executed after the current task. For this reason, the linguistic values that can be chosen for the Action variable are the different output transitions of the current task Upward and Downward Movement. As Fig. 8 illustrates, these transitions are Trans2 (the subject has to repeat the current task) and Trans3 (the subject will carry out the next task of the activity: Final Position).

Creating Steps for a Particular Task

The steps that must be specified for each task are the atomic gestures or postures that an aged person has to perform during a rehabilitation task, and in order to define them, the MS Kinect device is employed. Therapists select the relevant joints for the step from the skeleton model and then choose the type of step, and record the associated gesture or posture. The system creates a target area according to (1) the difficulty specified at Task level and (2) the selected joints. As the meta-model in Fig. 1 establishes, Joint and Area elements must be specified for each Step. Figure 12 shows the step-recording process. The transparent rectangle represents the successful process. The skeleton in the bottom left corner can be used by the therapist to select the relevant joints for a given area, and the slider above allows him to set the size of the target area.
Performance of a Treatment by a Particular Subject

The result of performing the process explained above is to obtain a rehabilitation model. The system is able to use this model to automatically generate the software components for previously defined therapies, activities, tasks and steps. While the patient is performing the different steps that compose each task, the system monitors the process, using MS Kinect and other sensors.

According to the values of the monitored parameters and the FIS defined by the therapists for each state, the system will make decisions about (1) the execution sequence of therapies, activities, tasks or steps (i.e., controlling the transitions of each state) and (2) adaptation to a specific older person (e.g., changing the difficulty level of a task if the subject is fatigued). Figure 13 shows the user interface when the subject is performing a step belonging to the task Upward and Downward Movement of the activity Seated Butterfly Wings. The size of the target area, represented by the rectangle in the middle of the image, is too small due to the difficulty defined at the Task level.

Adapting a Treatment to a Particular Subject

An important feature of our proposal is that, when a patient is doing an exercise, the system is able to adapt it to his real needs. For example, if the system detects that the subject is tired (Fatigue is High) during the task Seated Butterfly Wings (see Fig. 13) and the difficulty of the task was previously defined as high (Difficulty is High), the FIS associated with this task can infer a new difficulty level for this kind of situation (Difficulty Level is Medium) to alleviate the patient’s fatigue. In this case, the fuzzy rule that is activated, previously defined by the therapist, is the following:

\[ \text{IF Difficulty is High AND Fatigue is High THEN Difficulty Level is Medium} \]

Therefore, a new bigger target area associated with this step is defined (see Fig. 14), to help the patient to successfully perform the step.

Discussion and Conclusions

The main aim of our proposal is to provide therapists with a new interactive system for designing bespoke therapies. As some authors have stated [39, 40], the analysis of complex interactive systems involving new devices and interaction techniques needs different evaluation methods that allow their utility and effectiveness to be studied. Given the innovative nature of our proposal, we carried out a study on its utility and effectiveness following the method proposed by Olsen [39].

First of all, a definition is needed of Olsen’s STU (Situations, Tasks and Users) context, for which our proposal is relevant. In this case, the Users are the therapists that want to define physical treatments for a specific senior person, the Task corresponds to the design of treatments
composed of different therapies, and the Situation is the design of bespoke therapies to be carried out at home without the direct assistance of a therapist.

The following paragraphs analyze the claims of innovative systems identified by Olsen and use them to evaluate our interactive tool. These claims are: Importance; Problem not previously solved; Generality; Reduce solution viscosity; Empowering new design participants; Power in combination; and Can it scale up?

It is important to highlight that each claim was evaluated not only from the point of view of the application designers, but also taking into account the comments of therapists of the Castilla-La Mancha Acquired Brain Injury Association (ADACE), which offers support not only to
Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation

- **Importance.** As Olsen has stated, before any other claim, we need to evaluate and discuss importance. To analyze this claim, the focus is on studying the importance of the population of the system’s potential users, in our case therapists that design complete or partial treatments for older people with physical problems and who need to carry out rehabilitation activities at home without direct support from the therapist. The ability to design new therapies from scratch increases the potential audience and so the importance of our tool, as the therapists have freedom to create new physical activities by introducing all the variables needed to monitor the final execution, as well as specific positions or gestures that the system must check. This was analyzed and discussed with ADACE therapists, who welcomed the possibility of designing their own therapies.

- **Problem not previously solved.** Although the rehabilitation of older people has long been analyzed and there are some proposals that facilitate the assignment of a specific physical therapy to a person, based on a collection of previously designed therapies which can be carried out at home, our proposal goes beyond that. In our case, the therapists not only have the possibility of adapting some parameters of the previously designed therapies, but can also design all the elements of a specific therapy as well as the state of the person at rehabilitation time. This state can be established by means of cardiovascular and biological parameters using fuzzy levels. This means therapists can define rehabilitation activities included in a therapy and the conditions to control the flow between such activities by defining fuzzy variables that control the transitions. They also have the capability to establish specific physical exercises from scratch by simply recording the image of the new posture or gesture. The ability to build therapies and control the performance and physical conditions of the user was one of the main requirements with which the ADACE therapists agreed. They suggested a set of cardiovascular and biological parameters in order to define therapy templates.

- **Generality.** The STU of this rehabilitation tool is wider than previous ones, as it is possible to provide a more complete solution than others previously proposed. In particular, it is difficult to find solutions that allow the therapist to adapt some activities to a specific person, and those that do include this feature only offer the possibility of modifying some previously defined parameters. In our case, the solution could be applied to any particular situation, as the therapists can define from scratch any physical therapy for a specific patient. Moreover, by adding new types of Steps, the designed meta-model can also be useful in the definition of other types of cognitive therapies. Thus, we can talk about a general tool, quite a strong feature according to Olsen. Initial user stories, focus group sessions and questionnaires involving ADACE therapists identified shortcomings in the currently available rehabilitation tools. Generality was a recurrent deficiency. Flexibility in the consideration of physical parameters was a requirement during the design and later development of our tool.

- **Reduce solution viscosity.** This feature analyzes the effort required to iterate on many possible solutions. Olsen [39] defined the three following ways of reducing solution viscosity:

  - **Flexibility** analyzes the possibility of making rapid design changes that can be evaluated by users. Therefore, to make the design as simple as possible, it is important that the system can facilitate the design of a solution taking into account other therapies previously defined, named TherapyTemplate in our meta-model. Therapists can easily modify or adapt this template to a specific situation and person. This issue demands additional user experience studies, but our proposal has a positive attitude and is being accepted and used by the ADACE therapists.

  - **Expressive leverage** is achieved when a designer can accomplish more by expressing less and, as Olsen stated in [39], it is best achieved by reusing some previous design choices. In our case, the use of the TherapyTemplate element allows the therapist to reuse some previous designs that can be adapted to a specific context and person. This issue was much valued by the specialists, who generally use a collection of therapies that finally adapt to a specific person.

  - The last characteristic associated with viscosity is expressive match, defined as an estimation of how close the means for expressing design choices are to the problem being solved. In our case, the ability to see the final physical activity that the therapist is designing as well as other features such as the definition of the body joints that must be controlled in each activity is associated with this claim. All these features were suggested by the ADACE therapists, who need a clear view of what the users should do to rehabilitate a specific part of the body.

- **Empowering new design participants.** This issue addresses whether the tool makes it possible for other participants in the design process to benefit from its use by facilitating their involvement in the design tasks.
Although ADACE therapists were involved in the design and development of the proposed rehabilitation tool, other therapists could benefit from this tool and our participatory design process. The interaction and user interface of the system was designed considering specific constraints from practicing therapists, which is a point in favor of its acceptance by therapists not involved in its development.

- **Power in combination.** The effectiveness of a tool can be shown by supporting combinations of rehabilitation activities, or in other words, combining parts of existing treatments (activities, tasks and steps) into more complex ones. The tool can manage rehabilitation treatments, therapies, activities and tasks. New treatments can also be added by reusing available activities, tasks and even steps. Physical and cognitive rehabilitation tasks can be combined, and additional elements can be considered in the rehabilitation process using haptic and virtual reality elements.

- **Can it scale up?** Olsen’s last criterion is scalability and the ability to apply the solution to large complex problems. In collaboration with ADACE, we intend to validate the tool in large-scale problems in order to test its scalability in future work. To date, although the tool has been used to design complex treatments by combining available rehabilitation activities, we want to evaluate its usefulness in different situations.

We consider our proposal to have several advantages over existing proposals. Firstly, it can solve previously insoluble issues; secondly, it is a general solution; thirdly, it offers flexibility and expressive facilities to therapists and involves them in the design of new developments. Finally, our rehabilitation tool can be combined with different haptic and virtual reality techniques.

Our new system for creating therapies using a collaborative and natural user interface enables therapists to visualize, specify, document and create bespoke therapies from scratch, by defining activities, tasks and steps for individual patients. By using this system, motor therapies are defined as a composition of several interrelated scenarios with a specific flow, controlled by defining fuzzy logic rules. The rehabilitation tasks are defined by using a MS Kinect device. Finally, thanks to the natural user interface of our system, therapists can define treatments in an efficient way by using tagged objects, multimodal (voice and tactile) interaction and direct manipulation of specific rehabilitation elements.

Although in a first evaluation we have noticed that this system meets the Olsen claims to analyze the effectiveness of an innovative system, we now plan to carry out new evaluations, involving more therapists and patients, to test the system in a clinical environment. In this sense, we have to highlight that although the system design has been user-centered, we are currently designing new evaluations that involve both therapists and patients during a clinical study. One of these evaluations will focus on the usability of the systems and will provide us with useful insights into aspects such as to what extent therapists need to be trained to use the system, or to detect any other potential issues that might arise while using the system. It would also be interesting to perform evaluations of the system with different types of user to establish a statistical model for the automatic assessment of certain variables. This could lead to better fuzzy rules, for instance, relating fatigue and time-out between Steps, Tasks or Activities. Moreover, these evaluations will enable us to draw more in-depth conclusions about the usefulness of this technology.

We are also planning to add a module able to modify fuzzy rules at runtime in order to learn from patient reactions. For this aim, we are currently developing a Multi-Agent System (MAS) [41, 42] for supervising and applying therapies to older people. We expect this MAS to improve the decision-making process necessary to define therapies adapted to both the environment and the patient.

**Acknowledgments** This work was partially supported by Spanish Ministerio de Economía y Competitividad/FEDER under TIN2012-34003 grant and through an FPU scholarship (FPU12/04962) from the Spanish Government. We are also indebted to the specialists of the ADACE association for their invaluable cooperation.

**References**

9. Da Gama A, Chaves T, Figueiredo L, Teichrieb V. Improving motor rehabilitation process through a natural interaction based
Chapter 10

A Multi-Agent System for Acquired Brain Injury Rehabilitation in Ambient Intelligence Environments


Type of venue: International Journal (IF: 2.392, Q1)

This publication endorses this dissertation by compendium of publications.
A Multi-Agent System for Acquired Brain Injury rehabilitation in Ambient Intelligence environments

Cristina Roda\textsuperscript{a,b}, Arturo C. Rodríguez\textsuperscript{b}, Víctor López-Jaquero\textsuperscript{b}, Elena Navarro\textsuperscript{b}, Pascual González\textsuperscript{b}

\textsuperscript{a} Albacete Research Institute of Informatics (I3A), Albacete 02071, Spain
\textsuperscript{b} Computing Systems Department, University of Castilla-La Mancha, Albacete 02071, Spain

\textbf{A R T I C L E I N F O}

\textbf{Keywords:}
Ambient Intelligence
Multi-Agent System
Acquired Brain Injury
Motor impairment
Tele-rehabilitation
Fuzzy logic

\textbf{A B S T R A C T}

Acquired Brain Injury (ABI) is becoming an epidemic problem in our society, especially among older adults, being known as “the Silent Epidemic”. People with ABI demand solutions in research that offer them with a relearning process such that they can recover not only their physical skills but also their cognitive abilities. In this context, the inherent characteristics of Ambient Intelligence (AmI), transparency and intelligence, have turned it into one of the best approaches to square up to the impairment that ABI can cause. As AmI proposes the development of context-aware systems that integrate different devices to recognize the context and act accordingly, these systems can react promptly to the needs of people with ABI while they carry out their rehabilitation process. Moreover, the exploitation of a Multi-Agent architecture emerges as a natural solution to develop AmI systems, since agents are reactive, proactive and exhibit an intelligent and autonomous behavior. Therefore, in this paper, a Multi-Agent architecture (MAS) for healthcare AmI systems is presented. It contributes to treat people with ABI by using specific devices to control the patient’s movements and some physiological responses, such as the variation of the heart rate, during her rehabilitation process. In this way, the natural relationship between AmI and MAS is exploited. Finally, how this system is used to both design and execute therapies for people with ABI is presented.

1. Introduction

To the knowledge of the ABI network states [1] that people with ABI have suffered “damage to the brain that occurs after birth and which is not related to congenital disorders, developmental disabilities, or processes that progressively damage the brain”, but to different causes such as traumatic brain injury (TBI), brain tumours, degeneration of the blood vessels, etc. Unfortunately, this problem is affecting more and more people every year. Just for instance, according to the Brain Injury Centre [2] TBI is more common than breast cancer, spinal cord injury, HIV/AIDS, and multiple sclerosis (MS) combined. This explains why it is being known as “the Silent Epidemic”.

All of us are exposed to suffer this problem. However, older adults are among the most affected ones as they are more prone to suffer accidents [3]. For this reason, the development of any solution in this area must pay special attention to the needs and constraints imposed by this collective, because they are usually reluctant to the use of technology. For instance, a household survey [4] about ICT use carried out among 1001 people from England and Wales in 2003 showed that the use of computer was a minority activity amongst older people, because they considered it had low relevance to their daily-life. Ambient Intelligence (AmI) has become a meaningful advance in this sense, as it represents “the future vision of intelligent computing where environments support the people inhabiting them” [5], that is, it sets the focus on people’s real needs.

As Ducatel et al. [6] state, AmI promotes the development of innovative and intelligent user interfaces “embedded in an environment that is capable of recognizing and responding to the presence of different individuals in a seamless, unobtrusive and often invisible way”. This means that AmI systems become transparent as people do not perceive their complexity neither their presence, and are intelligent to react in a proactive and sensitive way [7] at the same time. These two characteristics have had a great impact because it has allowed technology to be used by people who, otherwise, would have been probably computer illiterate.

AmI systems have enabled [8] bringing health and social care to the
patient instead of bringing the patient to the health system. They have many different purposes, ranging from training for cognitive rehabilitation [9] till physical rehabilitation [10]. This work is part of ViSMART project (Virtual, Social, Multi-sensornal and Adaptive systems for the Rehabilitation of people with ABI), a project aimed at designing an Aml system for rehabilitation of persons with ABI. This paper focuses on those aspects more directly related to rehabilitation of motor impairment problems. One of the main features of our system is that it provides therapists with support to design new therapies, to adapt them to each specific person and to control their execution instead of using a fixed set of exercises. In this paper, we present the Multi-Agent [11] architecture that supports the execution of the therapies. The rest of the paper is structured as follows. After this introduction, Section 2 presents the related work. Then, in Section 3, the architecture of the rehabilitation system is detailed. Section 4 describes how to design, execute and adapt therapies by using ViSMART. Finally, the conclusions and future work are described in Section 5.

2. Related work

ABI, as defined in the previous section, may result from a number of different causes, either internal or external depending on the origin of the injury. Internal causes are the most frequent among older adults, usually due to vascular disorders, such as strokes or haemorrhages. External causes, generally known as traumatic brain injury (TBI), are usually due to traffic accidents, falls, etc. As observed from the causes described, all of us are exposed to this problem to some extent in our lives. As people with ABI have suffered brain damage, this result in different long-term deficits depending on the area injured and the level of damage, turning into physical deficits, cognitive deficits, emotional problems and behavioural deficits. People with ABI must be provided with a proper treatment as soon as possible because there is increasing evidence of its effectiveness during the first stages after injury [12]. Although their treatment is carried out in a healthcare centre, this alternative has several drawbacks, especially in terms of the time available for the treatment process. Moreover, Christiansen et al. [13] have confirmed that the use of computers in the treatment process encourages and stimulates cognitive behavior and helps the patient to recover damaged functions and overcome disabilities. However, two of the main challenges that any system for the treatment of ABI must face is that it must be both adaptable, taking into account different parameters, such as age or damage level, and adaptive so it reacts promptly to the state of the patient as she progresses during a rehabilitation session.

In this sense, as Aml technologies are expected to be sensitive, responsive, adaptive, transparent, ubiquitous, and intelligent [14], they emerge as a natural alternative to develop these systems. It is worth noting that intelligence is one of the most critical characteristics, as it makes Aml systems more sensitive, responsive, adaptive, transparent and ubiquitous. The main reason is that intelligence helps in understanding user environments and, consequently, in providing adaptive assistance [15]. This explains why Aml entails contributions from different AI areas [16], specially, Multi-Agent System (MAS) [11]. There is a natural relationship between Aml and MAS [17]. Aml proposes the development of context-aware systems that integrate different devices to recognize the context and act accordingly. Agents provide an effective way to develop such systems since agents are reactive, proactive and exhibit an intelligent and autonomous behavior [18]. Agents react to humans based on information obtained from sensors and their knowledge about human behaviors within agent-based Aml applications [19]. Rashidi and Mihalidis stated in [20] that Aml can be exploited in different application areas, being rehabilitation one of them.

In the rehabilitation field, we can find some works that propose intelligent robotic systems to assist the physical rehabilitation process of the patient, e.g., for lower limb rehabilitation [9] that uses a MAS to detect bioelectric and physical signals through a sensor network located in the patient’s body in order to determine his movement intention and assist him in doing such a movement. Another different proposal, called OntoRis [21], offers an ontology-based rehabilitation service that the patient can use to acquire comprehensive information about his prescribed rehabilitation treatment, or it can simply serve as an interactive learning platform for people interested in this particular medical field. Abreu et al. [22] focus their attention on cognitive rehabilitation, namely on using 3D games for neuropsychiatric disorders rehabilitation. These authors propose a MAS for automatic control while the patient is playing a 3D game in order to reduce the human intervention needed to manage the execution of software processes. Another interesting system has been proposed by Tian et al. [23] that uses the fusion of low-cost inertial measurement unit (IMU) and Kinect techniques to monitor upper limb motion. Other system [24] promotes the use of neural networks and interaction changes of the people to classify mental fatigue so that it can be properly managed in the context of rehabilitation. Another similar approach is IAServ (Intelligent Aging-in-place Home care Web Services platform) [25], which produces a personalized healthcare plan to meet the desire of patients of still living in their own house. This is done by first submitting the patient’s profile to IAServ by the healthcare professional, and then this profile is converted into an ontology specification to enable the generation of a personalized care plan for the patient, provided by an inference engine.

Neural Networks (NNs) have been also exploited in different aspects of healthcare. In [29], NNs are defined as “trainable systems that can learn to solve complex problems from a set of exemplars and generalize the acquired knowledge to solve unforeseen problems”. There are some works [28,29] that reflect how Artificial Intelligence (AI) mechanisms can be embedded in Aml environments to make them more intelligent, adaptable, energy efficient and suitable to the user’s needs. In this sense, the main AI approaches that could be applied in Aml are NNs and Fuzzy systems. They can be used alone or jointly to address a great variety of problems. There are some works that apply NNs within an Aml system. For instance, some authors [20] use them for recognizing complex activities. Other authors [5] use NNs as part of a vision-based fall detection system which extracts video features (3D motion, shape or inactivity) to detect falls. Another work [30] presents how to train a NN that takes as input those characteristics that describe the user’s interaction patterns and provides as output an estimation of the user’s mental fatigue. Moreover, there are other works that illustrate the perfect integration of such approaches, i.e. Aml and NNs, with agent-based systems. For instance, in [31] a multi-agent system is presented for the purpose of controlling a building environment in an smart way in order to achieve effective energy and comfort management. In this sense, it includes a NN to predict the indoor temperature preferred by the occupants, using data of the outdoor temperature and the associated time. Namely, in the healthcare area, other works have been published, such as the one in [26] that presents the SALSA framework for developing Aml hospital systems. Its architecture incorporates a reasoning component which uses a NN for controlling the agent’s actions. Other work [27] presents an agent-based architecture that combines simple sensors with an intelligent algorithm based on NNs able to recognize different activities occurred in an Aml environment, such as sleeping or eating, as well as abnormal behaviors. Initially, our proposal includes fuzzy reasoning mechanisms, but, as described in Section 4, our inference agent is ready to integrate other engines, such as NN, for the purpose of learning.

Vi-SMART has some similarities with respect to the one presented by Abreu et al. [22] as we also propose a MAS that is able to control the performance of all the tasks that a patient is doing during a rehabilitation therapy. However, unlike the work mentioned before [9] about physical rehabilitation, where an intelligent controller is used to manage the robot behavior, our proposal is focused on the performance...
of the tasks prescribed in a particular rehabilitation therapy. Furthermore, our system takes the advantage of using a MAS combined with AmI. The use of the AmI paradigm makes sense here when talking about older people, one of the collectives more affected with ABI, given that AmI provides transparent and intelligent mechanisms to interact with any type of software in a transparent, simple and easy way. This kind of intelligent system may avoid possible conflicts that arise when older people interact with software systems in a classical way as most of them are not familiarized with the use of technology at all.

Thus, this paper presents an extension and enrichment of the system described in [32-33]. In such papers, it is presented how the design of therapies is tackled. Furthermore, the set of editors for both therapies and fuzzy rules to control de rehabilitation process are described. To execute the therapies designed, in this work, a run-time environment for the execution of therapies is proposed relying on the multi-agent approach. By doing so, a complete system to create, perform and monitor therapies for physical rehabilitation of people with ABI is described. Hence, our proposal provides a tool that can be used by experts in the field of rehabilitation to define a set of customized therapies and the rules that determine the behavior of the system at runtime. In this way, activities can be adapted to people with ABI, and especially older people, while performing a particular therapy. The creation of therapies is driven by a meta-model. In the rehabilitation domain, a Therapy is composed of Activities, an Activity is composed of Tasks, and a Task is a set of Steps, which can be either Gestures or Postures that the older person has to perform. Relationships between elements of the same hierarchical level can be established in order to define a sequence of Therapies, Activities or Tasks, respectively. Therefore, the therapy model can be described as a composite state diagram.

The therapist can easily instantiate the meta-model by using a user interface developed for Microsoft PixelSense (see Fig. 3) and specify the movements that the older person has to perform by using Microsoft Kinect. The resulting model can be interpreted by the system, which automatically generates therapies that are performed by older people with motor impairment problems. In this sense, Vi-SMART can be adapted according to the patients' needs. These therapies are also supervised and monitored by the system using MS Kinect [34] and other kind of sensors that provide information about the physical conditions of the people with ABI in a transparent way. Furthermore, in the context of rehabilitation, a patient only needs to make the movements according to the specified therapy without direct manipulation with any device, taking advantage of the AmI paradigm. Moreover, the therapist defines a set of Fuzzy Inference Systems (FISs) (see Section 4.1 for further details) that enables our system to adapt the therapies and to decide the performance order of the tasks, activities and other parameters of therapies at runtime. After this overview of the Vi-SMART system, in the next sections, the MAS that supports it will be explained.

3. A MAS architecture for Vi-SMART

In this section, a general overview of our MAS designed to improve the supervision and execution of therapies carried out by people with ABI is discussed. It makes use of AmI to both enrich the environment provided to people with ABI and improve the decision making activities, so that Vi-SMART offers therapies adapted to the ambient situation and the person using the system. The MAS architecture of Vi-SMART has been designed to satisfy the goals and requirements that were systematically gathered by means of the Prometheus methodology (see Section 3.1). One of such goals was that Vi-SMART should be extensible so that new sensors can be easily integrated to improve the rehabilitation process. This is the main reason for using one agent for each type of ambient information or percept. Furthermore, it has been designed to achieve a better separation of concerns. In this sense, it was decided to specify a particular agent to decouple the inference engine from the rest of the system. In this way, Vi-SMART is prepared to add seamlessly other engines, such as those for Neural Networks, in the next future. It was also decided to decouple the adaptation from the therapy execution so that both can be changed independently.

Our MAS focuses on the physical rehabilitation of people with ABI. In this rehabilitation process, a therapist prescribes a treatment comprising several therapies designed for the rehabilitation of specific complain. Therapies are organized according to activities, and those activities consist of some tasks. Finally, these tasks are carried out by executing some steps, that can be either postures (e.g. arms raised) or gestures (a set of postures which implies a movement, e.g. throwing up and down the arms). Thus, this MAS is designed to achieve several goals: (1) it supports the execution of the therapies designed by the therapist for people with ABI, (2) it carries out sensing activities to monitor how well an activity is being done. Furthermore, this MAS also provides monitoring for some ambient features, namely the oxygen level, the stress, the heart rate and the emotional state of the person. All these features are captured by means of dedicated hardware devices that provide the data to the MAS. Such ambient information can be specified by using our context meta-model that allows the designer to reflect dynamic and static data from the context of use. This meta-model will be further explained in Section 3.1. Next, an overview of the design of our MAS is included.

3.1. MAS development methodology

The design of our MAS was made by using Prometheus Design Tool [35], the tool for supporting Prometheus methodology for intelligent agent development. Prometheus comprises three main stages: System Design, Architectural Design and Detailed Design. System Design is a stage to help in organizing the requirements of the MAS by expressing them into an understandable graphical notation. This phase is made based on the principles of the goal-oriented approach. In Architectural Design stage the composition of the MAS is designed. This stage includes deciding what agents the MAS will have, the messages they will exchange or the capabilities they will exhibit. Finally, Detailed Design stage is related to the internal design of agents. As these stages are made, the System Overview diagram is composed. This diagram is a great tool for representing the general architecture of a MAS system (see Fig. 1). This graphical notation includes the agents involved (in light brown color), the protocols that include the messages they exchange (in magenta color), the actions each agent is responsible for (in light green color) and lastly, the percepts that represent the external information arriving to the MAS (in light red color) from the ambient.

3.2. Managing ambient information

With the aim of managing the external information arriving from the ambient to the MAS, dedicated agents are used. We decided to use one agent for each type of external ambient information (percept), because it is easier to extend its capabilities. This is an important feature, since the availability of sensors for assorted sources of information advances very quickly. Thus, our MAS must be ready to easily consider extra valuable information from the ambient to improve the available information used in the decision making process. The ambient information we are already considering is related to the oxygen level, posture, gesture, stress, heart rate and emotional state. In this manner, AgentOxygenLevelCapture, AgentPostureCapture, AgentGestureCapture, AgentStressCapture, AgentHeartRateCapture and AgentEmotionalStateCapture are the agents responsible for capturing the information from the ambient, respectively. Fig. 1 depicts an overview of the design of our MAS.

Oxygen level represents the saturation of oxygen in the blood, and it is a good indicator to help to find out when the person making the therapy is becoming tired, i.e. when the patient's fatigue increases.
3.3. Specifying ambient information

As previously mentioned, the ambient information can be specified during the design process by using our context meta-model (see Fig. 2) that allows the designer to reflect dynamic and static data from the context of use. Some relevant concepts related to the context surrounding our system are the Environment, the User and the Devices. They can be considered as Context Elements in a context model. Moreover, in many cases, these elements are modeled by describing their Capabilities. In this sense, Vi-SMAR, as most of the Tele-rehabilitation systems, must be aware of such context (ambient) information that changes at runtime, such as the acoustics of the room (Environment) where the patient is located, the brightness of the display (Device) the patient is watching while performing the therapy, and especially those related to the patient (User), such as the oxygen level, posture/gesture, heart rate or stress level. This makes necessary to take into account, first, i) the Dynamic Context Data, that represent features of a context element that change dynamically, and, second, ii) the Alert Conditions associated to dynamic context data that represent the current value, condition or range of values the system must be aware of. Similarly, it is also useful to specify the ambient information that takes part of user's context of use and which will not change while the user is doing her therapy, i.e. the Static Context Data. For example, the age of the patient (User) or her physical disabilities, such as visual impairment; the maximum resolution of the display (Device); or even the dimensions of the room (Environment). Finally, in the following, the main elements of this context meta-model are explained:

- **ContextModel.** It represents an instance of the context model.
- **ContextElement.** It is used to specify an element whose description is relevant for the system. It can be a User, a Device or any aspect related to the Environment.
- **Capability.** It is used to specify a set of features related to a context element. For instance, the GPS capability of a mobile device, or the vital signs of a user. The features that define a Capability are called ContextData and they can be either dynamic, DynamicContextData, or static, StaticContextData.
- **StaticContextData.** It represents characteristics of the context that are not going to change while the user is carrying out her therapy. For instance, it could be important to describe the visual conditions of a user as part of the Capability senses since users with visual impairments will not interact with the system in the same way as users without such kind of problems. Additionally, a StaticContextData element can be related to one or more Literals since some features only need one Literal to be described, e.g. Frequency: 1.2 GHz, while others need two or more, e.g. MaxResolution: 1900, 1080.
- **DynamicContextData.** It represents features of the context that change at runtime having an impact on the system. For instance, the feature Heart Rate of a user changes over time given that the user's heartbeat does. This dynamic information is not directly related to Literals, but to AlertConditions.
- **AlertCondition.** It is employed to set a value, range of values or condition to fire an alert. When the value measured at runtime of a dynamic feature matches an AlertCondition (i.e. it matches a concrete value, range of values or condition), the system will react...
Accordingly,

- **AtomicAlertCondition**. It is used to denote a condition that cannot be further decomposed. An atomic condition is always related to a concrete **DynamicContextData** element, e.g., the **DynamicContextData** Heart Rate can be related to a condition relevant to the system, such as GREATER_THAN 180 bpm.

- **CompositeAlertCondition**. It is used to specify complex conditions that are composed of two or more **AlertConditions** related by means of logical operators. For instance, LOWER_THAN 180 bpm AND GREATER_THAN 120 bpm for the **DynamicContextData** Heart Rate.

- **Sensoring**. It is used to specify relationships between dynamic context information and the **Device** (acting as a sensor) in charge of measuring it. The **Sensoring RefreshTime** attribute is used to specify the frequency of measurement.

- **Literal**. Expression used to specify types, values and measurement systems for specific context data, e.g. IntLiteral: 180 bpm. Enumerations can also be used by means of EnumLiteral.

### 4. Using Vi-SMARt

In this section, a description of how Vi-SMARt is used for designing as well as executing and adapting therapies is presented. Therapists design the process of rehabilitation by creating a statechart diagram model that entails the activities, tasks and steps the patients should perform during a therapy. These concepts are represented by nodes (states) within the diagram. In addition, therapists can include one or several FISs in each state for controlling the adaptation of the related activity, task or step, as well as the transition to the next state. For this reason, once the therapy has been designed, it can be done by patients in an autonomous way, while the system monitors the execution and adapts it at runtime, taking into account the patient’s state.

#### 4.1. Designing therapies

Despite the fact that the design of a therapy is not controlled by Agents of our MAS, it is important to explain this process to facilitate the explanation of the therapy execution and adaptation later on. Therapies are represented by a multi-level statechart diagram model, so that every therapy is composed of activities, activities are made up by tasks, and every task has a set of steps associated. There are transitions between nodes at the same level. These transitions are controlled by a FIS created in each node at design time. Vi-SMARt enables therapists to create these diagrams and the related FISs through a touchable user interface for MS PixelSense (see Fig. 3).

#### 4.2. Executing therapies

Once the therapies have been designed, the MAS of Vi-SMARt can exploit them. One of these agents, **AgentTherapy**, is responsible for running and controlling the activities that compose the therapy, the tasks of every activity, and the steps in every task. A therapy behaves like a four-level state diagram. However, as aforementioned, transitions between states are controlled by different FISs that the therapist
Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation

C. Roda et al.

Neurocomputing 231 (2017) 11–18

defines for each state. A FIS is a method that uses fuzzy rules, membership functions and fuzzy logic to map an input space into an output space. It also contains a reasoning mechanism that performs inference operations relying on the fuzzy rules. Biometrical signals can be used as input fuzzy variables and the transitions are represented as output variables of the FIS. In this sense, the therapist defines the fuzzy sets related to these variables and builds the rules that will later guide the inference process. Fuzzy sets are those that can contain elements with a degree of membership. For an input variable, many fuzzy sets can be defined by means of functions (one function for each fuzzy set). These functions calculate the degree of membership of a variable according to the value of that variable. Each set is tagged with a linguistic value (e.g. Very Low, Low or Medium). These tags are used for representing facts in the rules, e.g., “Number of repetitions is Low”. “Number of repetitions” represents an input variable, and the fact will be evaluated by the FIS according to the membership function of the Fuzzy set tagged with “Low”. Rules provide a natural way to express how transitions should be made by using linguistic values rather than numerical ones. For example, if the person who is performing the therapy is becoming tired (i.e. her fatigue is increasing), it might be better to do a less demanding task (i.e. a transition to another state). Unlike classical logic, fuzzy logic does not establish rigid boundaries when dealing with a variable (e.g., Number of repetitions is low when it is 3 or less), but it defines regions in the domain of the variable that can overlap and determines the degree of truth of a fact according to the real value. The FIS evaluates all facts of the set of rules and uses the obtained values to make decisions.

In Vi-SMARt, all fuzzy inference is handled by AgentInference, keeping the inference process isolated from the rest of agents of the system. In this way, just this agent accesses the profile of the user in order to reduce the development effort for privacy issues. Privacy has been proven an issue for any kind of medical related software, since most people find their medical information especially sensitive [38]. As aforementioned, this AgentInference is prepared to add other inference engines apart from the fuzzy one, such as NN, for the purpose of learning. The AgentInference takes the values of the input variables from the AgentAdaptation and obtains a specific value by means of the rules defined by the therapist and its reasoning mechanism. The reasoning mechanism is implemented by using Mamdani’s method [39], which is usually preferred rather than others since it is highly intuitive and similar to human reasoning. According to Mamdani’s method, a FIS is defined by a set of rules, where both antecedents and consequents are fuzzy sets. This method obtains a fuzzy output that is defuzzified to obtain the final concrete value. For this aim, Mamdani’s method obtains the degree of membership for the input values by using the fuzzy sets of the antecedents of the fuzzy rules. These degrees of membership are used to obtain a single value that is applied to the fuzzy set of the consequent of each rule obtaining a new output fuzzy set. All the consequents obtained are aggregated to obtain an area. This area is employed to compute, usually by applying the defuzzification centroid method, a value used to make the decision.

4.3. Logging therapy statistics

Another action carried out by the AgentTherapy is generating statistics. To facilitate the monitoring of the patients’ progress, specialist need to have feedback from the execution of the therapies they design [40]. In order to provide such feedback to the therapist, AgentTherapy logs every action the patient does during the execution of the therapy, logging every activity, task and step the patient took during the therapy. These statistics include completion time, repetitions, postures and gestures identified, tries before a posture or a gesture is correctly done, as well as fatigue. With these statistics, the therapist can follow the patient’s progress and can receive valuable feedback to design other therapies or adapt the current ones.

4.4. Adapting therapies

Although therapists have feedback from the log to design the therapy, it is impossible to foresee all the ambient factors that may have an impact on the execution of the therapy. Therefore, the transitions from one element of the therapy to another is made in terms of fuzzy rules. These fuzzy rules support the therapist in designing under what circumstances the transition should be to one element or another, thus supporting an adaptive workflow of the therapy execution, i.e. the system adapts automatically and dynamically the workflow of the therapy execution, taking into account the fuzzy rules previously defined. As described in Section 4.2, these fuzzy rules are evaluated by AgentInference.

Moreover, therapy execution can also be adapted by means of AgentAdaptation. It makes use of the information captured through sensors and stored by AgentPersonDataManager, and the logs of the therapy execution to adapt the fuzzy sets to the context. For instance, if AgentTherapy reports the user has made too many repetitions of a gesture or a posture to do it correct, AgentAdaptation decides to reduce the precision required to consider the posture or gesture as correct by adapting the corresponding fuzzy sets. As this loss of precision of the user can be due to fatigue, AgentAdaptation, by checking the evolution of the data captured by the sensors, may recommend a break in the therapy execution. As observed, Vi-SMARt provides a therapy execution environment that manages a wide number of ambient situations that can change the therapy execution as defined by the therapist.

4.5. Designing therapy adaptation

In order to specify the variables, the rules and the fuzzy sets that control the therapy adaptation, our system provides the therapists with an interface that can be easily used for this purpose (see Fig. 4). Several...
FISs can be created in each state and configured for controlling the transitions from one state to another and adapting other variables at runtime, such as the difficulty of tasks or the number of repetitions.

To create a FIS, therapists select input and output variables, define the number of fuzzy sets for each variable and the values that determine the membership to the set. Once the variables have been chosen, the rules are presented in a matrix form. Despite of the potential complexity that a rule system with many variables can reach, the interface only accepts two input variables and one output variable. The matrix form represents the inputs in rows and columns. One input variable is represented in rows. The number of rows is determined by the number of fuzzy sets of the variable and it is tagged with its linguistic values. The other input variable is represented in columns by using the same procedure. Cells represent the value for the output variable when the input variables have the values of the corresponding row and column. For example, suppose therapist chooses Heart Rate and Difficulty as input variables, and Action (Transition) as output variable. Then, he defines five fuzzy sets for Heart Rate (Very Low, Low, Medium, High and Very High) and three fuzzy sets for Difficulty (Low, Medium and High). Therefore, the matrix will have five rows (for Heart Rate) and three columns (for Difficulty). If the cell (2, 3) shows the value “Transition3”, the rule represented will be: “If Heart Rate is Low AND Difficulty is High THEN Action is Transition3”.

5. Conclusions and future work

In this paper, a system for monitoring the execution of therapies in an Ambient Intelligence environment by using a Multi Agent System is presented. This system has been developed to provide people with ABI with physical therapies for rehabilitation. These therapies are designed by therapists from scratch by defining, not only the elementary activities, but also the rules that determine the behavior of the system. Their design is driven by a meta-model to define the rehabilitation environment as well as facilities to define Fuzzy Inference Systems that enable the system to adapt the therapies and to decide the execution order of activities, tasks and steps at runtime.

In order to support these features, a MAS has been designed. It makes use of Aml to enrich the environment and improve the decision making activities to provide therapies adapted to the ambient situation and the person using the system. Therefore, the MAS monitors ambient features, namely the oxygen level, the stress, the heart rate and the current emotional state of the person. All these features are captured by means of dedicated hardware that provide the necessary data to the MAS. In addition, our system provides the therapists with feedback from the execution statistics to enable them to monitor the execution of the therapies and the patients’ evolution.

The main benefits of our proposal are derived directly from the design of the MAS architecture itself. As aforementioned, it has been designed to provide an extensible system so that new sensors can be easily integrated to make available new capabilities beyond the existing ones. Moreover, Vi-SMART also pursues a better separation of concerns regarding the inference process to facilitate the integration of other inference engines, such as those supporting Neural Networks. In addition, our system also separates the therapy adaptation of the therapy from its execution, so that the adaptation mechanisms can be modified without changing the execution of the therapy at all, and the other way around. Therefore, the novelty of our approach is that it provides flexibility when used in the context of rehabilitation, since it enables therapists to create and adapt every rehabilitation process to the specific needs of their patients.

However, it is obvious that our proposal, as any other, presents some limitations. Currently, no learning mechanism is supported in Vi-SMART and it has not been validated in a real environment. Thus, our ongoing work is focused on the evaluation of Vi-SMART in a real set up. For this aim, we rely on the collaboration of some associations that assist older people and people with ABI. Besides, another challenge to be addressed in the near future is the integration of other techniques, such as Neural Networks, to support learning from therapy executions in order to improve the adaptation process.

Acknowledgements

This work was partially supported by the Spanish Ministry of Economy and Competitiveness and by the FEDER funds of the EU under the project grant InsaPre (TIN2012-34003). It has also been funded by the Spanish Ministry of Education, Culture and Sport thanks to the FPU scholarship (FPU12/04962).

References


[23] Y. Tian, X. Meng, D. Tao, D. Liu, C. Feng, Upper limb motion tracking with the
Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation

Arturo C. Rodríguez got his Bachelor’s degree in Computer Science in 2010 at the University of Castilla-La Mancha (UCLM). He is a software engineer and research assistant at the LoUISE group of the Albacete Research Institute of Informatics and Ph.D. student in Computer Science at the UCLM. He is currently working on Model-Driven Development of Post-WIMP systems and their application in physical and cognitive rehabilitation of people with Acquired Brain Injury.

Victor López-Jaquero got his Ph.D. in Computer Science from the University of Castilla-La Mancha (UCLM) in 2005. He is a lecturer and a researcher at the UCLM, teaching courses related to Data Bases and HCI. His research activities are focused on HCI and more specifically on adaptable and adaptive user interface design, following a model-driven approach and multi-agent systems to support the adaptation facilities. He is author of more than 40 papers in relevant conferences and journals indexed in JCR regarding User Interfaces design, including, but not limited to, XML-based User Interface Description Languages and User Interfaces Generation for different target platforms, following a Model-Driven approach, Adaptive and Adaptable User Interfaces and Multi-Agent Systems.

Elena Navarro, Ph.D. (2007, University of Castilla-La Mancha, UCLM), MSc (2010, Rey Juan Carlos University), MSc (2000, University of Murcia), BCS (1993, UCLM), is an associate professor at the Computing Systems Department of the UCLM (Spain) where she lectures in Software Engineering. Prior to this position, she did research at the Computer Science Laboratory of the Agricultural University of Athens (Greece), funded by the EU’s Training and Mobility of Researchers Program. She had previously served on the staff of the Regional Government of Murcia. Her current research interests are Requirements Engineering, Software Architecture, Model-Driven Development and HCI.

Pascual González received his Ph.D. in Computer Science from Technical University of Madrid (Spain) in 1999. Since 1991, he has been lecturer and researcher at the University of Castilla-La Mancha, where he lectures in Software Engineering and Human-Computer Interaction. He is the head of the Laboratory on User Interfaces and Software Engineering (LoUISE) research group and he is leader of several research projects. He is the author of different papers in relevant international conferences and journals indexed in JCR regarding User Interfaces design, including, but not limited to, Human Computer Interaction, Augmented and Virtual Reality, and Software Engineering applied to interactive systems.

C. Roda et al.

Neurocomputing 231 (2017) 11–18


Cristina Roda, MEd (2013, University of Castilla-La Mancha, UCLM), MSc (2011, UCLM), BCS (2010, UCLM), is a software engineer researcher at the LoUISE group of the Albacete Research Institute of Informatics and Ph.D. student in Computer Science at the UCLM (Spain), holding a scholarship from the National Government. Prior to this position, she worked as a software engineer at Vector Corp. Her research interests are in Software Engineering, Model-Driven Development, Software Architecture, HCI, Adaptation, Context-Aware Computing and Ambient Intelligence.
Chapter 11

Past and Future of Software Architectures for Context-Aware Systems: A Systematic Mapping Study


Type of venue: International Journal (IF: 1.424, Q1)
Past and Future of Software Architectures for Context-Aware Systems: A Systematic Mapping Study

Cristina Roda¹, Uwe Zdun², Elena Navarro¹, Víctor López-Jaquero¹

¹LoUISE Research Group, University of Castilla-La Mancha, Spain
{Cristina.Roda, Elena.Navarro}@uclm.es, victor@dsi.uclm.es

²Software Architecture Group, University of Vienna, Austria
Uwe.Zdun@univie.ac.at

Abstract. There is a clear growing interest on context-aware systems currently. Context-aware systems are evolving towards interacting with the user in a transparent and ubiquitous manner, especially by means of different types of sensors that can gather a wide range of data from the user, the platform he/she is interacting with, and the environment where such interaction is taking place. Therefore, those systems are aware of the user, the platform and the environment to adapt their behaviour to changing conditions. It is worth noting that the Software Architecture of a system is a key artefact during its development and its adaptation process. Hence, the definition of the software architecture becomes essential while developing context-aware systems since it should reflect how the context is tackled for adaptation purposes. However, we did not find much evidence in the literature about software architecture proposals that deal with context-awareness in an explicit way during the adaptation process. On the other hand, there are Human Computer Interaction works that focus on context-aware adaptations, but neglect partially or completely any possible change in the system architecture during the adaptation process. Due to this, we perceived a need to analyse which research works highlight the use of context and its relationship with the software architecture in existing context-aware systems. We have designed a systematic mapping study to provide an overview about the different approaches used in context-aware systems in order to align the adaptation at architectural level (changes in the configuration of architectural components) and at HCI level (changes in the interaction modality or the user interface in general).

Keywords: systematic mapping study, software architecture, context-aware system, user, platform, environment, quality of adaptation, evaluation, quasi-gold standard

1 Introduction

Recently, a trend has emerged that aims at transforming the environment where the user interacts with the system into a smart environment so that he can interact with the system in a transparent and seamless manner. In this sense, context-awareness emerges as a suitable approach for developing such systems conferring flexibility, adaptability and capability of acting autonomously on behalf of users [26]. Context-aware systems pay attention to the context surrounding the user to adapt their behaviour to the new conditions. Such context usually refers to the user, the platform he/she is interacting with, and the environment where such interaction is being done. As stated in [5], these three dimensions (User, Platform and Environment) represent the main cornerstones for defining the context.

As Perry and Wolf [37] already realized in 1992, the Software Architecture (SA) of every software system and its specification should be carefully considered. They defined the SA as “the selection of architectural elements, their interactions and the constraints on those elements and their interactions necessary to provide a framework in which to satisfy the requirements and serve as a basis for the design”. Therefore, SA becomes a key artefact whenever it is necessary to develop a system that offers complex capabilities.

Given the importance of context and its different dimensions in context-aware systems for adapting its behaviour, as well as the importance of the SA while designing a system, it could be thought that most of the proposals in the area have been defined exploiting SA mechanisms to provide such context-aware adaptations. Nevertheless, we did not find much evidence in the literature about software architecture proposals that deal with adaptation based on the context of use. Most of the architectural proposals focus on some specific kind of adaptation, but mostly from the point of view of software architecture and the possible configurations of its components. Consequently, such architectural approaches do not deal with context-awareness in an explicit way during the adaptation process, usually not considering the user at all and hence neglecting the adaptation from the point of view of Human-Computer Interaction (HCI). On the other hand, there are also HCI works that focus on context-aware adaptations, but do not pay many attention to the likely software architecture support.
In this way, we perceived a need to analyse which research works highlight the use of context and its relationship with the software architecture in existing context-aware systems. To the best of our knowledge, a systematic mapping study is the best approach to carry out such analysis. As defined by Kitchenham and Charters [32], a systematic mapping study consists of a broad review of primary studies in a specific topic area that aims at identifying what evidence is available on the topic. Thus, by making a mapping study, we will be able to provide an overview about the different approaches used in context-aware systems in order to align the adaptation at architectural level (changes in the configuration of architectural components) and at HCI level (changes in the interaction modality or the user interface). Moreover, we will also identify the gaps existing up to now with regard to the deficiencies and weaknesses of systems when dealing with context and its different dimensions. These findings will help the research community by providing guidelines for modelling, retrieving and managing context and its different dimensions, as well as by describing how to consider such context by the software architecture in order to provide the user with a proper adaptation experience.

This paper is structured as follows. Section 2 presents the research methodology applied to perform this systematic mapping study. Next, Section 3 describes how the data collection for this mapping study was carried out. Section 4 shows the results obtained from the search process of the mapping study by answering the research questions previously established. Section 5 presents the discussion derived from the analysis of the different results. Our main conclusions are presented in Section 7. Finally, the selected papers of this mapping study and their publication venues are listed in Appendix A, as well as a glossary in Appendix B which includes many terms considered relevant from the results.

2 Research Methodology

As aforementioned, this article presents a systematic mapping study that focuses on providing an overview about the state of the art on software architecture concepts when used for developing context-aware systems. A systematic mapping study [38] is an evidence-based form of secondary study that provides an overview of a research area, identifying common publication venue types (e.g. conference, journal or workshop), quantitative analyses (e.g. number of published studies per year), and research findings in the investigated research field. Kitchenham and Charters [32] also provide a definition of a systematic mapping study, considering it as a broad review of primary studies in a specific topic area that aims to identify what evidence is available on the topic.

This mapping study has followed several guidelines [54][10][32] for conducting systematic studies, especially in the field of software engineering. Moreover, we have also taken into consideration the work by Zhang et al. [59][60] while performing our search strategy, since it was driven by a concept introduced by these authors: the Quasi-Gold Standard. In this sense, our mapping study is composed of the following six stages, being each one of them described separately in the following sections:

1. The rationale for conducting the mapping study (Section 2.1).
2. The establishment of the Research Questions (RQs) that this mapping study answers (Section 2.2).
3. The search process we have followed relying on the Quasi-Gold Standard approach (Section 2.3).
4. The categorization we have followed to classify the results obtained (Section 2.4).
5. How the data collection has been done applying the selection criteria (Section 3).
6. The results obtained along with the answers to the RQs (Section 4).

2.1 Mapping Study Rationale

The first step while performing a systematic study is to define the main reason that has led to carry it out. In this case, we did not find much evidence in the literature about software architecture proposals that provide support for adaptation based on the context of use. Most of the architectural frameworks are centred on supporting some kind of adaptation, but mostly from the point of view of the software architecture and the possible configurations of its components. Such proposals do not manage the context-awareness explicitly during the adaptation process, so that they neglect the user completely and hence also the adaptation from the point of view of HCI. On the other hand, there are HCI studies focused on context-aware adaptations which neglect partially or completely any possible change in the system architecture during the adaptation process to support such adaptation. In this sense, we have detected a need to analyse which research works highlight the use of context and its relationship with the software architecture in existing context-aware systems.

By doing so, we will be able to provide an overview about the different approaches used in context-aware systems in order to align the adaptation at both architectural level (changes in the configuration of architectural components) and HCI level (changes in the interaction modality or the user interface in general). Moreover, we will also identify the existing gaps up to now, i.e. what are the deficiencies and weaknesses of systems when dealing with context and its different dimensions. These findings will help the research community to define guidelines for modelling, retrieving and managing
context and its different dimensions, as well as to establish how the software architecture may support the context in order to provide the final user with a good adaptation experience. Therefore, the main objectives of our study are the following:

- **Objective 1.** Review software architecture concepts, such as designs, frameworks, components, styles, patterns, infrastructures and specifications, proposed for designing and building context-aware systems with the aim of understanding the relationship between them.
- **Objective 2.** Review how context-based adaptations are considered by those software architecture concepts proposed for context-aware systems.
- **Objective 3.** Review the maturity of the different software architecture concepts proposed for context-aware systems, as well as the maturity of the methods for assessing the context-based adaptations.

### 2.2 Research Questions

Based on those objectives previously mentioned, we defined the following Research Questions (RQs), which will be answered throughout this paper:

- **RQ 1.** Which architectural designs, frameworks, components, styles, patterns, infrastructures and specifications are used in the existing context-aware systems? (Related to Objective 1)

  **Rationale.** Architectural designs, frameworks, components, styles, patterns, infrastructures and specifications (i.e., software architecture concepts) have been used as artefacts to guide the development of software products and establish a certain level of claimed quality. However, there is no overview of primary studies on defining and/or exploiting those software architecture concepts in relation to the existing context-aware systems. In this article, we aim at offering such an overview in order to help researchers and practitioners to analyse which of those architectural topics are the most relevant and the most used.

- **RQ 2.** How do the software architecture concepts proposed for context-aware systems consider the three main dimensions of context (User, Platform and Environment) to support adaptation? (Related to Objective 2)

  **Rationale.** In the field of context-aware computing, there are many different proposals for modelling the context. However, as stated in [5], there is a consensus that the three main cornerstones for defining the context are: User, Environment and Platform. Therefore, we want to discover how each dimension of context is addressed and taken into account within the different architectural proposals to carry out the adaptations based on these three dimensions of context. In this sense, we propose three different sub-research questions where each one of them refers to a different dimension of context for the sake of clarity.

- **RQ 2.1.** How do the software architecture concepts proposed for context-aware systems consider the user dimension of context to support adaptation?

  **Rationale.** The user dimension here refers to all characteristics related to the user of the system: preferences regarding the user interface (colour, font size, presentation, content, etc.), profile features (name, age, job, etc.), physical impairments (blind, single-handed, etc.), health condition, physiological data (heart rate, blood pressure, etc.), skills, expertise, contacts, etc. Thus, we are interested in studying the user dimension of context as a separate concern to identify how the different software architecture concepts proposed for context-aware systems are addressing it to carry out adaptation.

- **RQ 2.2.** How do the software architecture concepts proposed for context-aware systems tackle the platform dimension of context to support adaptation?

  **Rationale.** As stated in [7], this particular dimension of context is related to the software side of a system, such as the operating system, the technology infrastructure represented by a well-defined programming language, applications supported, etc. However, the platform is not only limited to the software, but also it refers to hardware elements, such as the screen size of the monitor, a temperature sensor, or the network characteristics (bandwidth, speed, etc.). Therefore, we want to analyse, separately from the other context dimensions, how the different software architecture concepts proposed for context-aware systems tackle the platform dimension of context to support adaptation.

- **RQ 2.3.** How do the software architecture concepts proposed for context-aware systems take into account the environment dimension of context to support adaptation?

  **Rationale.** Along with the user and platform, the environment is also considered as one of the main dimensions of context. This dimension includes the ambient noise, light level, temperature, weather, date, nearby people, etc. in the location where the interaction between the user and the system is taking place. Thus, similarly to the other dimensions of context, we are interested in studying this context dimension too with the aim of identifying
how the different software architecture concepts proposed for context-aware systems are addressing it to carry out adaptation.

- **RQ 3.** What evaluations have been performed so far for validating the different context-aware adaptations proposed for context-aware systems? (Related to Objective 3)

  **Rationale.** Every software artefact, such as a user interface or a software architecture, should guarantee a certain degree of quality when delivered to prevent the final user from rejecting it. Therefore, context-aware adaptation should also ensure a certain degree of quality whenever they are applied. In this sense, proposals supporting context-aware adaptation should provide some mechanisms to estimate the degree of quality of these adaptations, i.e. to evaluate whether context-aware adaptations are perceived positively by the final user or the entity the adaptation was designed for. With this purpose, we aim to discover to what extent the evaluation of context-aware adaptation in context-aware systems has been tackled in the literature. By doing so, we will be able to estimate the maturity of the proposals for adaptation.

- **RQ 4.** What evaluations have been performed so far for validating the different software architecture concepts proposed for context-aware systems focused on adapting the system using the surrounding context (user, platform and environment)? (Related to Objective 3)

  **Rationale.** We want to find out what evaluations have been carried out until now to validate the software architecture concepts proposed for context-aware systems. These evaluations may range from informal evaluations, just showing some examples, to more formal empirical evaluations. In this sense, we want to categorize the different software architecture concepts, depending on the type of evaluation presented. This will enable us to determine the maturity of such software architecture concepts for developing context-aware systems.

### 2.3 Search Process Based on the Quasi-Gold Standard

Two have been the main reasons that have led us to use the systematic search strategy presented by Zhang et al. [59][60]: first, i) to identify the most relevant works in the literature with regard to our topic in an objective and systematic way, and second ii) to properly assess the search performance. These authors stated that most of the systematic studies in Software Engineering apply a subjective search process based just on the expertise and knowledge of the team on the research topic and tested on supposedly “well-known” studies previously gathered by such team to assess the search performance. Zhang et al. claimed that this set of “well-known” samples depends on the knowledge of the searchers on the particular topic and, thus, it cannot replace the Gold Standard (i.e. the known set of identified primary studies in a collection according to the definition of the Research Questions) for evaluation, since it is not possible to have a complete set of primary studies before the execution of the systematic study. Therefore, these authors introduced the concept Quasi-Gold Standard (QGS) [59] that refers to a set of known studies extracted from related venues, such as domain-specific conferences and journals perfectly recognized by the research community in the field, for a given period of time. Thereby, the QGS acts as a Gold Standard, but with the only restriction of where (venues) and when (during a period of time).

In this sense, with the aim of providing objectivity to the search process, Zhang et al. integrate the QGS method into the systematic search process that relies on the analysis of information from the works collected, i.e. from the QGS, instead of using a subjective set of studies which depends on the perceptions of the people who carry out the search. Besides the support for objectivity, another benefit of applying the QGS method is that it can be used to evaluate the search strategy performance, that is, to evaluate the search results in order to guarantee a certain quality level of the automated search. Zhang et al. propose to carry out this assessment by means of the Sensitivity which is defined as the proportion of relevant studies retrieved for a given topic:

\[
\text{Sensitivity} = \frac{\text{Number of relevant studies retrieved}}{\text{Total number of relevant studies}} \times 100\%
\]

However, as the gold-standard is unknown, the sensitivity cannot be calculated so that the QGS is used to measure the Quasi-sensitivity of the search universe.

Fig. 1 depicts the different steps that integrate this systematic search strategy. The first step consists in identifying the most relevant venues (both conferences and journals) regarding our research topic in order to discover the QGS (Step 2), and also identifying the engines (libraries and databases) that will be used for our automated search (Step 4). In step 3, the search strings are built to use them as input for the automated search. These strings can be defined either by researchers, thanks to their knowledge on the matter or, more objectively, they can be elicited from the studies in the QGS by means of the analysis of word frequency or content. In step 5, the search results from the automatic search are evaluated regarding the quasi-sensitivity and compared with a threshold previously established. If the search performance is found acceptable (usually when quasi-sensitivity ≥ 80%), then the results obtained by the automated search are combined with the QGS, finishing the search process. Otherwise, the flow must go back to step 3 to refine the search...
strings until the search performance reaches an acceptable level. In Section 3.2, each one of these steps are explained in more detail regarding our systematic study on software architectures for context-aware systems.

![Workflow of the systematic search strategy](adapted from [60])

### 2.4 Categorization of Results

Several well-defined and widespread taxonomies have been used in this work to classify the selected papers obtained from our search process. One of them is the categorization originally proposed by Wieringa et al. [55] for classifying papers regarding the Research type. Table 1 shows these research types along with its description that we have adapted from Kuhrmann et al. [34]. Note that one result can be categorized as more than one research type.

<table>
<thead>
<tr>
<th>Research type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluation research</td>
<td>It is shown how the technique is implemented in practice (solution implementation) and which are the consequences of the implementation in terms of benefits and drawbacks (implementation evaluation). This also includes identifying problems in industry. An example for evaluation research is reporting of an industrially hosted case study.</td>
</tr>
<tr>
<td>Solution proposal</td>
<td>A solution for a problem is proposed. The solution can be either novel or a significant extension of an existing technique. The potential benefits and the applicability of the solution are shown by a small example or a good line of argumentation.</td>
</tr>
<tr>
<td>Validation research</td>
<td>Techniques investigated are novel and have not yet been implemented in practice. Techniques used are, for example, experiments, that is, work done in the lab.</td>
</tr>
<tr>
<td>Philosophical papers</td>
<td>These papers present a new way of looking at existing things by structuring the field in the form of a taxonomy or conceptual framework.</td>
</tr>
<tr>
<td>Opinion papers</td>
<td>These papers express the personal opinion of somebody whether a certain technique is good or bad or how things should be done. They do not rely on related work or research methodologies.</td>
</tr>
<tr>
<td>Experience papers</td>
<td>These papers explain on what and how something has been done in practice. It should be the personal (informal) experience of the author.</td>
</tr>
</tbody>
</table>
Another categorization, which has been applied to the selected papers when analysing the RQ 4, is the one proposed by Wohlin et al. [57] for classifying the Empirical strategies or evaluations presented in the papers (see Table 2).

Table 2. Empirical strategies categorization

<table>
<thead>
<tr>
<th>Empirical strategies</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Survey</td>
<td>A survey is a system for collecting information from or about people to describe, compare or explain their knowledge, attitudes and behaviour by means of interviews or questionnaires. Surveys are conducted when the use of a technique or tool already has taken place or before it is introduced.</td>
</tr>
<tr>
<td>Case study</td>
<td>A case study is an empirical enquiry that draws on multiple sources of evidence to investigate one instance (or a small number of instances) of a contemporary phenomenon within its real-life context, especially when the boundary between phenomenon and context cannot be clearly specified.</td>
</tr>
<tr>
<td>Experiment</td>
<td>An experiment is a formal, rigorous and controlled investigation. Based on randomization, different treatments are applied to or by different subjects, while keeping other variables constant, and measuring the effects on outcome variables. During an experiment, quantitative data is collected and then statistical methods are applied.</td>
</tr>
</tbody>
</table>

In addition to these empirical strategies, we have also categorized the evaluation as a mere Example when no formal empirical strategy is applied. In those cases, the paper has not been classified neither as a Validation or Evaluation research because of the lack of formalism and empirical evidence those papers presented.

Finally, we have also classified the results with regard to the following criteria in order to facilitate the later analysis and discussion. Note that it is not possible to predefine the different categories that belong to each one of these criteria before analysing the papers.

- **By Country**, denotes the country of the institution of the first author of the paper. It is a single value per paper.
- **By Venue**, indicates whether the paper is a Journal, a Conference or a Workshop. It is a single value per paper.
- **By Research topic**, represents the main research topic throughout the paper, e.g. context modelling, context inference, service delivery, context-aware applications development, UI adaptation, etc. A paper can be classified according to more than one topic.
- **By Application domain**, represents the main application domain/s of the proposal, e.g. healthcare, e-learning, smart home, etc. Not every paper is oriented to an application domain, but it is possible to have more than one domain for some papers.
- **By What is adapted**, indicates which element of the context-aware system is the subject of the adaptation, e.g. services, UI, architecture configuration, content, presentation, navigation, etc. Not every paper expresses what is adapted. In contrast, it is possible to have more than one element that is adapted during the adaptation process in the same work.
- **By Software architecture concept** (RQ 1), indicates the type of software architecture concept employed, e.g. service-oriented-architecture (SOA), multi-agent system, multi-layered architecture, etc. Every paper selected should describe some software architecture concept in the proposal, or even more than one.
- **By User dimension of context** (RQ 2.1), determines the different characteristics of the user considered in the proposal for performing the adaptation of the system, e.g. preferences, profile, physiological data, social profile, abilities, disabilities, etc. Not every paper deals with this dimension of context. In contrast, one work can present more than one user characteristic.
- **By Platform dimension of context** (RQ 2.2), determines the different characteristics of the platform considered in the proposal for performing the adaptation of the system, e.g. network, bandwidth, battery level, screen size, etc. Not every paper deals with this dimension of context. In contrast, one work can present more than one platform characteristic.
- **By Environment dimension of context** (RQ 2.3), determines the different characteristics of the environment considered in the proposal for performing the adaptation of the system, e.g. location, noise, light, temperature, weather, date, nearby people, etc. Not every paper deals with this dimension of context. In contrast, one work can be classified according to different characteristics.
• By Other dimensions of context (RQ 2), determines if the proposal makes use of other dimensions of context different from User, Platform or Environment. Not every paper deals with other dimensions of context. In contrast, one work can be classified according to more than one category for this RQ. For example, some other dimensions of context are the following:
  o Historical data, such as the interaction history of the user. It is usually used to know the evolution of the user and hence making changes and adaptations in the system that improve the user experience
  o User activity, refers to the activity the user is performing at a specific moment of time, e.g. sleeping, cooking, brushing the teeth, etc.
  o Interaction task, refers to the task of interaction the user is making with the system, e.g. opening a new window, clicking a link, typing in the keyboard, etc.

3 Data Collection

In this section, the procedure for collecting all data of this mapping study is presented. Section 3.1 presents the different criteria used to select only the relevant results from the search process. The related venues and engines identified for this study are described in Section 3.2.1. Section 3.2.2 presents the most relevant articles that define the QGS. Section 3.2.3 depicts the definition of the search string for the automated search. Section 3.2.4 presents the outcomes obtained from the automated search. Finally, Section 3.2.5 exhibits the evaluation of the search performance by means of the quasi-sensitivity calculation.

3.1 Selection Criteria

Generally, the search process in a systematic study returns a great deal of results, but not all of them are usually considered relevant. For this aim, we have defined the following criteria to filter out the papers returned and select only the relevant ones.

The inclusion criteria are:

• I1. The study presents any kind of software architecture concept that supports the development, design, etc., of context-aware applications.
• I2. The study reports an example or similar that shows how a particular software architecture concept supports context-aware adaptations.
• I3. The study presents some type of evaluation for validating a specific software architecture concept regarding a particular context-aware system.
• I4. The study presents some type of evaluation for assessing the adaptation in context-aware systems considering the supporting architecture.

On the other hand, the exclusion criteria are:

• E1. The study is not written in English.
• E2. The study is not related to Computer Science research field at all.
• E3. The study is related to Computer Science research field, but not to Software Engineering or Human-Computer Interaction fields, or to a software architecture concept applied on context-aware systems.
• E4. The study is an older publication from the same authors about the same approach.
• E5. The study is a short one, less than three pages.
• E6. The study is a large one, e.g. a PhD thesis or a book.
• E7. The study is a secondary or tertiary study (e.g. a systematic mapping study or literature review), not a primary one.
• E8. The study is a technical report, editorial note, preface, index, introduction, presents a special/theme issue, it is unreferenced, illegible or inaccessible (i.e. paper is not available through the search engines used).
• E9. The study has been published after 2015.
3.2 Quasi-Gold Standard-Based Search Strategy

This section details the steps of the search strategy we have followed to find the primary studies to be used in this mapping study. We would like to remind that we have based our search strategy in the Quasi-Gold Standard concept presented by Zhang et al. [60].

3.2.1 Identifying related venues and engines

The first stage of the search process is related to the identification of the most relevant related venues used in the next step for establishing the QGS (see Section 3.2.2), as well as the identification of the engines for the automated search in step 4 (see Section 3.2.4).

Regarding the related venues, they are usually selected without difficulty considering both the research questions previously established and the expertise and knowledge of the researchers on the subject under study. However, for this mapping study, this selection was challenging because our study is focused not only on one research field, but on two research fields clearly differentiated in Computer Science: Software Engineering (SE) and Human-Computer Interaction (HCI). On the one hand, we are interested in architecture concepts, such as designs, frameworks, components, styles, patterns, infrastructures and specifications (derived from the research questions described in Section 2.2), which are obviously connected to the area of Software Architecture (SA). On the other hand, we are only interested in those architectures designed for context-aware systems, which are directly connected to the Pervasive and Ubiquitous Computing area in the HCI field. Although, our first thought was to include conferences and journals related to both research fields, after a quick review of the literature and a thorough discussion, we realized that most of the relevant works related to context-aware systems were published in specialized venues of HCI and Pervasive and Ubiquitous Computing for establishing the QGS. Thereupon, we finally decided to set as related venues those connected to HCI and Pervasive and Ubiquitous Computing. Moreover, those works published in venues related to SA and SE fields will be discovered during the automatic search stage given that they are also indexed in the selected search engines. Table 3 shows the final set of relevant venues selected for establishing the QGS, along with their associated publisher, digital library or engine. Note that five different venues focused on HCI and Pervasive Computing have been identified, namely the journals Human-Computer Interaction, IEEE Pervasive Computing, Personal and Ubiquitous Computing, and User Modeling and User-Adapted Interaction; as well as the ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp). Note that this last conference, UbiComp, is the result of merging the most renowned conferences in the field during the last few decades, such as PERVASIVE, UbiComp and HUC. These five venues have been selected since they are considered the most relevant and referenced ones with regard to HCI and Pervasive Computing.

<table>
<thead>
<tr>
<th>Type of venue</th>
<th>Venue</th>
<th>Publisher</th>
<th>Digital libraries/Engines</th>
</tr>
</thead>
<tbody>
<tr>
<td>Journal</td>
<td>Human-Computer Interaction</td>
<td>Taylor &amp; Francis</td>
<td>ACM Digital Library, Scopus</td>
</tr>
<tr>
<td></td>
<td>IEEE Pervasive Computing</td>
<td>IEEE Computer Society</td>
<td>ACM Digital Library, IEEE Xplore, Scopus</td>
</tr>
<tr>
<td></td>
<td>Personal and Ubiquitous Computing</td>
<td>Springer London</td>
<td>ACM Digital Library, Springer Link, Scopus</td>
</tr>
<tr>
<td></td>
<td>User Modeling and User-Adapted Interaction</td>
<td>Springer Netherlands</td>
<td>ACM Digital Library, Springer Link, Scopus</td>
</tr>
<tr>
<td>Conference</td>
<td>ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)</td>
<td>ACM</td>
<td>ACM Digital Library, Springer Link, Scopus</td>
</tr>
</tbody>
</table>

1 Institute of Electrical and Electronics Engineers
2 Association for Computing Machinery
3 International Conference on Pervasive Computing
4 International Conference on Ubiquitous Computing
5 International Symposium on Handheld and Ubiquitous Computing
Regarding the selection of engines for the automated search, Zhang et al. [60] stated that it depends on several factors, such as whether they publish the selected venues or not, the degree of coverage among them, and also the availability and accessibility of the different articles through such engines. Furthermore, they observed that ACM Digital Library and IEEE Xplore constitute a must-have when talking about the engines to be selected for any automatic search for systematic studies in SE. Although our initial intention was choosing the minimum set of search engines for avoiding as much overlapping as possible, we noticed that many works were only indexed by one of these search engines, especially those indexed by Scopus and Springer Link. Therefore, for the sake of completeness, our final set of selected search engines include ACM Digital Library and IEEE Xplore, as recommended by Zhang et al., as well as Scopus and Springer Link. Note that when searching with ACM, we have chosen the option Guide to Computing Literature instead of ACM Full-Text Collection as the former includes a wider set of articles. Table 4 shows the search engines finally chosen for conducting the automated search (see Section 3.2.4).

<table>
<thead>
<tr>
<th>Selected search engines</th>
<th>Web reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACM Digital Library</td>
<td>2</td>
</tr>
<tr>
<td>IEEE Xplore</td>
<td>31</td>
</tr>
<tr>
<td>Scopus</td>
<td>49</td>
</tr>
<tr>
<td>Springer Link</td>
<td>51</td>
</tr>
</tbody>
</table>

Zhang et al. stated [59, 60] that authors can also employ other search engines not directly derived from the related venues while performing a systematic study, but, in such a case, the QGS is only valid for the assessment of the search performance through those engines inferred from the nominated venues.

### 3.2.2 Establishing the Quasi-Gold Standard

The second stage of the search process is the establishment of the QGS. Zhang et al. [59, 60] recommend a manual search through the related sources identified in the previous step for finding the most relevant and most referenced works regarding our research topic. In this manner, all these relevant papers will be part of our QGS. However, this manual process is very time-consuming since all articles published in such venues have to be analysed one by one. To facilitate such work, we propose a semi-manual search which consists in using a generic search engine, such as Google Scholar [23], to identify a good initial set of studies related to our topic, and then applying the snowballing technique, screening one by one the resulting articles, to find the most relevant and most cited works.

The semi-manual search followed here is based on the guidelines for snowballing in systematic studies provided by Wohlin [56]. This author proposes among others the following tips that we have used to identify a good initial set, i.e. a set of papers to use for the snowballing procedure:

1. A good initial set of studies can be identified by using, for example, Google Scholar, with the aim of avoiding bias in favour of a specific publisher.
2. If too many papers are found, then the most relevant and highly referenced ones should be identified.
3. The initial set should cover different publishers, different years and different authors.
4. The initial set should be formulated taking into account the keywords used in the RQs.

Therefore, the semi-manual search was conducted by using Google Scholar with an initial search string composed of the keywords architecture and context-aware extracted from the RQs (see Section 2.2). Seventeen papers were obtained from this first search, ordered by relevance according to Google algorithms, including various PhD dissertations related to relevant authors on the subject, such as Dey’s PhD dissertation [17], as well as some technical reports. However, these types of articles, such as PhD theses and technical reports, were not included because of our exclusion criteria (see Section 3.1). Nevertheless, some specific vocabulary and synonyms related to the terms context and context-aware were gathered. For instance, in Dey’s PhD dissertation [17], some context-aware synonyms were found, such as adaptive [8], reactive [16], responsive [20], situated [30], context-sensitive [43] and environment-directed [21]. Some of these terms have been used in the construction of the search string for the automated search (see Section 0).

Afterwards, the snowballing strategy was applied, looking for articles that cite those included in the initial set (forward snowballing [56]), and also looking for articles cited by the works in the initial set, i.e. those which appear in the references list (backward snowballing [56]). Thereby, all works found were filtered by their number of references, and also by their publication venue given that we are only interested in those ones published in the venues previously selected (see Section 3.2.1). Finally, we obtained a set of 17 relevant papers which are indeed focused on architectures for context-aware systems and hence they constitute our QGS. Table 5 presents the distribution of the relevant papers selected from the
related venues. Notice that the QGS covers the publication period between 1999 and 2011 so that 1999 was selected as
the starting date for the automated search (see Section 3.2.4).

Table 5. Quasi-Gold Standard distribution

<table>
<thead>
<tr>
<th>Venues</th>
<th>Papers of the QGS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human-Computer Interaction</td>
<td>[18][27]</td>
</tr>
<tr>
<td>IEEE Pervasive Computing</td>
<td>[24][33][42][46]</td>
</tr>
<tr>
<td>Personal and Ubiquitous Computing</td>
<td>[29][44]</td>
</tr>
<tr>
<td>User Modeling and User-Adapted Interaction</td>
<td>[13][14][15][25][28][39][52]</td>
</tr>
<tr>
<td>ACM Int. Joint Conf. on Pervasive and Ubiquitous Computing (UbiComp)</td>
<td>[1][47]</td>
</tr>
</tbody>
</table>

3.2.3 Defining the search string for the automated search

As represented in Fig. 1, the search process at this point can be carried out in two different ways in order to obtain the
final search string for the automated search. On the one hand, as Zhang et al. stated [3][4], most systematic studies found
in the literature in SE conduct automated search in a subjective manner, i.e. researchers build search strings for automated
search based on their knowledge and experience about the research topic. On the other hand, these search strings can also
be elicited in a more objective way based on the QGS (dashed lines in Fig. 1). In this sense, text mining techniques are
used to discover the most frequently occurring words or sentences within the QGS. By doing so, the search strings can be
elicited using such candidate search terms, producing a higher search performance rather than using a subjective search
string definition.

In our case, we started with the definition of the search string in a subjective manner, only depending on our knowledge,
expertise and manual findings. Subsequently, different keywords and terms repeatedly found during the establishment of
the QGS were incorporated to our search string to offer a more objective perspective to the search process. Obviously,
the definition of the search string should be as useful and accurate as possible with the aim of extracting the most relevant
papers related to the scope of the study. Nevertheless, the process of defining an adequate search string is neither an easy
task nor a quick one because this study is focused on analysing the different architectures proposed for context-aware
systems. Moreover, as mentioned before in the RQs (Section 2.2), we are especially interested in studying the three
dimensions of context: User, Environment and Platform. Therefore, our final search string should take into account all
these keywords, i.e. architecture, context-aware, and also the dimensions of context: user, environment and platform, as
well as the most common synonyms for such terms.

With regard to the term architecture, some synonyms were retrieved from the RQs, such as: design, framework, component, style, pattern, infrastructure and specification. We included all these words in the search string to find as many related works as possible, but focusing on any type of architecture concept applied in context-aware systems. As
aforementioned, this study deals with two different research areas, Software Architecture and context-awareness, that
address the term of architecture in a different way, using different terminology. For this reason, we have used the following
keywords to address the concept architecture in our search string:

architecture OR design OR framework OR component OR style OR pattern OR infrastructure OR specification

Regarding the concept context-aware, some synonyms have also been included in the search string as we noticed while
establishing the QGS that this term appears in different forms. Some identified synonyms of context-aware were: adaptive [8], reactive [16], responsive [20], situated [30], context-sensitive [43] and environment-directed [21]. However, some of
them, such as adaptive, reactive, responsive and situated, are too generic to use them as a single word. For solving this
problem, we explored the literature in order to find out the most common terms used along with these generic concepts.
The term adaptive usually is used along with user interface, agent, application and wearable; the term reactive is often
used with environment; responsive is frequently used with the terms environment and application; and finally, the word
situated is broadly used with communication and application, as well as the composed term situation aware. This led us
to define the search string related to context-aware as follows:

“context-aware” OR “adaptive user interface” OR “adaptive agent” OR “adaptive application”
OR “adaptive wearable” OR “reactive environment” OR “responsive environment” OR
“responsive application” OR “situated communication” OR “situated application” OR
"situation aware” OR “context-sensitive” OR “environment-directed”

With regard to the dimensions of context, user, environment and platform, some synonyms have also been used to
build the search string, including several relevant words commonly used when referring to such dimensions of context:
Finally, the final search string used for the automated search was defined as follows by gathering the previous ones connected by an AND operator:

Table 6. Search string (similar terms are included in brackets)

<table>
<thead>
<tr>
<th>Concept</th>
<th>Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture</td>
<td>(architecture OR design OR framework OR component OR style OR pattern OR infrastructure OR specification)</td>
</tr>
<tr>
<td>Context-aware</td>
<td>AND (“context-aware” OR “adaptive user interface” OR “adaptive agent” OR “adaptive application” OR “adaptive wearable” OR “reactive environment” OR “responsive environment” OR “responsive application” OR “situated communication” OR “situated application” OR “situation aware” OR “context-sensitive” OR “environment-directed”)</td>
</tr>
<tr>
<td>Dimensions of context</td>
<td>AND (user OR human OR device OR platform OR environment OR task)</td>
</tr>
</tbody>
</table>

Note that we searched for papers published between 1st of January 1999 and 31st of December 2015. On the one hand, the selection of 1999 as our starting date was set during the establishment of the QGS (see Section 3.2.2), because we observed that the first relevant studies related to architectural approaches focused on context-aware systems dated from 1999 [1][47]. On the other hand, the selection of 2015 as end date was chosen because this systematic study started in 2016. Therefore, papers gathered from early 1999 till the end of 2015 may bring a full coverage of existing work on architecture concepts for context-aware systems.

3.2.4 Conducting the automated search

This section presents the steps followed during the search process, and depicted in Fig. 2, to obtain the final set of relevant papers for this mapping study. The automated search using all the four search engines previously selected (see Section 3.2.1), i.e. ACM Digital Library, IEEE Xplore, Scopus and Springer Link, was firstly conducted using the final search string (see Table 6) and the limitation of the publication years from 1999 to 2015, both included. Note that the search string was differently coded depending on the search engine used, in order to fulfil the syntax requirements and capabilities of each search engine. Furthermore, the search process over each engine was conducted looking up in the Title and Keywords sections in order to keep the number of results obtained manageable. Table 7 shows the number of articles retrieved from each search engine, as well as the total of papers, 6701, after merging all the results obtained.

Table 7. Articles retrieved with the automated search

<table>
<thead>
<tr>
<th>Search engine</th>
<th>Articles retrieved</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACM Digital Library</td>
<td>540</td>
</tr>
<tr>
<td>IEEE Xplore</td>
<td>122</td>
</tr>
<tr>
<td>Scopus</td>
<td>2400</td>
</tr>
<tr>
<td>Springer Link</td>
<td>3639</td>
</tr>
<tr>
<td>Total</td>
<td>6701</td>
</tr>
</tbody>
</table>
Next step was to remove all duplicates as some papers appeared as a result of more than one search engine. We found 421 papers duplicated, so we continued the search process with 6280 articles. Then, we checked how many articles of the QGS (see Section 3.2.2) appeared in the set of papers provided by the automated search, identifying 14 articles out of 17 papers of the QGS. 3 articles were missing in our search results with respect to QGS. Therefore they were also included in the set of papers to be used in the following steps.

Then, we applied several filters to the 6283 articles selected previously taking into account the exclusion and inclusion criteria defined in Section 3.1. First, we filtered out those papers whose venue and title revealed clearly that they were not related at all with the topic being addressed by this study, keeping as result 1807 articles. Second, we filtered out by abstract, introduction and conclusions, i.e. we kept those papers that, after reading their abstract, introduction and/or conclusions, we found enough evidence that they were relevant to our study, keeping 476 references. Finally, a filter by content was applied so that some more articles were excluded once they were read carefully. Obviously, all papers included in the QGS were never excluded when applying these type of filters as they must be part of the final selected papers. Note that the most common exclusion criteria at this point were: E3, as some papers were definitely not useful for...
this work; E4, since some papers were older publications from the same authors about the same approach; E8, because some articles were completely illegible, especially figures detailing software concepts; and E9, as some papers were published after 2015. Table 8 presents the number of papers excluded by exclusion criterion. Note that some of the papers excluded satisfied more than one exclusion criterion. At the end of this search process, we selected 355 references for being analysed in this mapping study (see appendix A. 1 for further details) as will be presented in Section 4.

Table 8. Number of papers excluded by exclusion criterion

<table>
<thead>
<tr>
<th>Exclusion criterion</th>
<th>No. of papers excluded</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
<td>10</td>
</tr>
<tr>
<td>E2</td>
<td>273</td>
</tr>
<tr>
<td>E3</td>
<td>5326</td>
</tr>
<tr>
<td>E4</td>
<td>20</td>
</tr>
<tr>
<td>E5</td>
<td>30</td>
</tr>
<tr>
<td>E6</td>
<td>8</td>
</tr>
<tr>
<td>E7</td>
<td>105</td>
</tr>
<tr>
<td>E8</td>
<td>171</td>
</tr>
<tr>
<td>E9</td>
<td>5</td>
</tr>
</tbody>
</table>

3.2.5 Evaluating search performance

The completeness and performance of the automated search has been assessed by means of the aforementioned quasi-sensitivity concept, whose formula is presented below:

\[
\text{Quasi-sensitivity} = \frac{\text{Number of QGS studies retrieved from the automated search}}{\text{Total number of QGS studies}} \cdot 100\%
\]

A sensitivity threshold greater than 80% can be seen as an acceptable reference for sensitivity evaluation of search performance, as recommended by Zhang et al. [60]. The quasi-sensitivity has been calculated as follows, taking into account that the total number of studies in the QGS is 17 (see Section 3.2.2 for more details) and the number of QGS studies retrieved when performing the automated search was 14:

\[
\text{Quasi-sensitivity} = \frac{14}{17} \cdot 100 \approx 82.4\%
\]

Therefore, with a quasi-sensitivity of 82.4%, we can state that the performance of the automated search proposed for this study is quite acceptable.

4 Results

A data extraction form was prepared that includes several fields to answer each of the RQs. We also collected other information of each paper, namely, keywords, country, venue type, research topic, application domain, research type, main contribution of the paper, and, finally, what elements are adapted during the adaptation process. All these data were collected from all the 355 selected papers (see appendix A. 1). In the next subsections, first, we present an overview of the selected papers regarding their publication venue, year and country (Section 4.1.1), research topic, application domain and research type (Section 4.1.3), and what is adapted (Section 4.1.4). Afterwards, in Sections 4.2, 4.3, 4.4 and 4.5, some descriptive statistics and frequency analysis are presented to answer each research question introduced in Section 2.2.

4.1 Overview of Selected Papers

This section shows an overview of the selected papers in terms of their publication venue, year, country, research topic, application domain, research type as well as what is adapted by the system. Furthermore, at the end of this section, there is a summary about the different context dimensions covered by the SA concepts found in the selected papers, and the type of evaluation performed regarding those SA concepts.

4.1.1 By publication venue

Table 9 shows the top ten most popular publication venues for papers focused on context-aware systems regarding the number of papers published. As shown in this table, around 20% of the 355 selected papers are published in these
publication venues: Personal and Ubiquitous Computing, User Modeling and User-Adapted Interaction, Wireless Personal Communications, World Wide Web journals, as well as the International Conference on Embedded and Ubiquitous Computing. In appendix A.2, it is shown the distribution of the 355 selected papers over all the publication venues, having identified 235 different publication venues: 133 conferences, 77 journals and 25 workshops.

Table 9. Number of papers published in the top ten most popular publication venues

<table>
<thead>
<tr>
<th>Ranking</th>
<th>Venue</th>
<th>Type</th>
<th>No. of papers</th>
<th>% of papers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Personal and Ubiquitous Computing</td>
<td>Journal</td>
<td>14</td>
<td>3.94</td>
</tr>
<tr>
<td>2</td>
<td>User Modeling and User-Adapted Interaction</td>
<td>Journal</td>
<td>11</td>
<td>3.10</td>
</tr>
<tr>
<td>3</td>
<td>Wireless Personal Communications</td>
<td>Journal</td>
<td>9</td>
<td>2.54</td>
</tr>
<tr>
<td>4</td>
<td>World Wide Web</td>
<td>Journal</td>
<td>7</td>
<td>1.97</td>
</tr>
<tr>
<td>5</td>
<td>International Conference on Embedded and Ubiquitous Computing</td>
<td>Conference</td>
<td>6</td>
<td>1.69</td>
</tr>
<tr>
<td>6</td>
<td>Procedia Computer Science</td>
<td>Journal</td>
<td>5</td>
<td>1.41</td>
</tr>
<tr>
<td>7</td>
<td>Journal of Systems and Software</td>
<td>Journal</td>
<td>5</td>
<td>1.41</td>
</tr>
<tr>
<td>8</td>
<td>International Conference on Universal Access in Human-Computer Interaction</td>
<td>Conference</td>
<td>5</td>
<td>1.41</td>
</tr>
<tr>
<td>9</td>
<td>Multimedia Tools and Applications</td>
<td>Journal</td>
<td>5</td>
<td>1.41</td>
</tr>
<tr>
<td>10</td>
<td>Expert Systems with Applications</td>
<td>Journal</td>
<td>4</td>
<td>1.13</td>
</tr>
<tr>
<td></td>
<td><strong>Total</strong></td>
<td></td>
<td><strong>71</strong></td>
<td><strong>20%</strong></td>
</tr>
</tbody>
</table>

4.1.2 By year and country

Fig. 3 illustrates the distribution of the number of published papers from 1999 to 2015. It can be noticed that there is a rising interest of the research community on context-aware systems and their architecture artefacts especially since 2007.

Moreover, Fig. 4 depicts the top ten countries that have devoted more effort on research in context-aware systems. Researchers from these countries have published in total 224 papers of the 355 selected ones, i.e. the 63.1% of all selected papers. As observed Asian countries, South Korea and China, lead this ranking significantly, followed by Spain, Italy and USA.
4.1.3 **By research topic, application domain and research type**

By analysing the content of the selected papers more thoroughly, they can be classified by research topic, application domain and research type. Fig. 5 presents the research topics of the selected papers as a tag cloud so that the more frequent a research topic is, the bigger its font size is. The research topics most frequently used are: *context management, context inference, context modelling, context gathering* and *context-aware applications development*. These terms have been described in a glossary in Appendix B.

Furthermore, the most frequent application domains of context-aware systems were also identified. Fig. 6 shows these *application domains* and groups of interest of context-aware systems, being the most recurrent ones are: *healthcare, smart environment, smart home, mobile environment, older people, Ambient Assisted Living, e-learning, guide-assistant* and *recommendation systems*.
Finally, Fig. 7 shows how the selected papers were classified according their research types using Wieringa et al. [55] taxonomy (see Section 2.4 for more details). Most of the selected papers are categorized as Solution proposals (335 papers, i.e. 94.37%), followed by Validation research (107 papers, i.e. 30.14%), Experience paper (6 papers, i.e. 1.69%), Evaluation research (1 paper, i.e. 0.28%), and finally Philosophical paper (1 paper, i.e. 0.28%). Some papers are categorized as both Solution proposal and Validation research, e.g. P3, P8, P10 or P329, and there are also some papers, e.g. P7 or P25, which are both Solution proposals and Experience papers. Other papers, such as P48 or P311 are only Validation researches and the paper P274 is the only one classified as Validation research and Experience paper. The paper P87 is the only one categorized as Solution proposal and Philosophical paper. Finally, P27 is the only one classified as Solution proposal and Evaluation research.

335 papers: Solution proposal

107 papers: Validation research

1 paper: Evaluation research
P27

1 paper: Philosophical paper
P87

6 papers: Experience paper
P7, P25, P37, P90, P243

P48, P82, P109, P119, P118, P127, P151, P179, P192, P196, P214, P229, P349, P350, P255, P272, P281, P289, P311

Fig. 7. Research types of selected papers

4.1.4 By what is adapted

Most of the papers analysed in this mapping study are focused on the adaptation process and, generally, they clearly identify what is adapted by the system. We considered this information very useful and a cornerstone when designing and developing context-aware applications. Therefore, Fig. 8 depicts an overview of the most frequently adapted elements in context-aware systems. As shown, services is the element of context-aware systems most frequently adapted, followed by the content presented in the user interface, as well as the presentation of the information.

Fig. 8. What is adapted in context-aware systems?
4.2 RQ 1 – Software Architecture Concepts

In order to provide an overview of software architecture concepts used for building context-aware systems, after reading all the 355 selected papers, we have identified the following 16 different software architecture concepts and patterns:

- Aspect-Oriented Architecture (AOA)
- Blackboard Architecture
- Client-Server Architecture
- Component-Based Architecture
- Distributed Architecture
- Event-triggered Architecture
- Micro-Architecture
- Middleware Architecture
- Model-Driven Architecture (MDA)
- Multi-Agent System (MAS)
- Multi-layered Architecture
- Peer-to-Peer (P2P) Architecture
- Pipe-and-filter Architecture
- Plug-and-Play (PnP) Architecture
- Publish/subscribe Architecture
- Service-Oriented Architecture (SOA)

Fig. 9 provides an overview of the SA concepts and the number of selected papers that apply them. Notice that the same work can use more than one architecture concept. It is interesting to highlight that the multi-layered approach is the most commonly used one for building context-aware systems, followed by component-based, SOA and MAS.

In the following lines, we offer some examples of papers regarding these SA categories in more detail. For example, there are plenty of works that use SOA, among other approaches, such as the papers P138 which proposes the use of SOA along with a multi-layered and middleware approach. P253 also follows a SOA paradigm as it proposes the CAMPH system to provide a number of services for context data acquisition, context storage, context reasoning, service organization and discovery used to facilitate the development of pervasive homecare applications. This work also makes use of a multi-layered architecture, since the middleware infrastructure separates context data and context-aware services.
in different layers so that they can be used by the applications. P314 also uses SOA along with a P2P approach to provide an optimized context management.

Other works are only focused on applying a SOA approach. For instance, P48 makes use of Web services for designing context-aware applications mainly due to the loosely coupling benefit. P2 and P144 consider SOA too, as well as P82 that integrates services into widgets used for collecting data from sensors and acting like GUI interactors to make changes to the environment. For example, a Light widget can identify and change the light conditions of a specific location.

There are other proposals built using a MAS approach, such as P295 or P168. P139 proposes an agent-based framework along with a multi-layered approach for providing customized services based on context history and users’ preferences. The proposed framework consists of the following layers: data gathering layer, context management layer, preference management layer and application layer. Some other papers present a component-based architecture for developing context-aware systems, e.g. P246 and P173. The architecture presented in P264 also follows the structure of a middleware and it is distributed.

With regard to multi-layered architectures, some examples are P275 and P315 which identify four layers: the application layer that consists of context-aware, proactive and mobile applications as well as sensors to provide application-specific context information to such applications; the Event-Condition-Action (ECA) engine that is responsible for the application behaviour; the privacy enforcement layer that ensures that privacy-sensitive information can only be accessed according to privacy policies; finally, the infrastructure layer entails functionality for context management and reasoning. P276 follows a multi-layered approach having several layers, such as the ontology editor for modelling ontologies, the authoring tool for creating learning objects from existing content and contextualize them with references to the ontology, and the delivery platform for context-aware delivery of learning objects and managing the learning progress.

P166 proposes a client-server architecture, called ALICE (Architecture for Location-Independent Computing Environments) to support mobility by providing a range of application level client/server protocols. P44 exploits the publish/subscribe paradigm to receive and redirect messages from sensors to sensor and device models.

4.3 RQ 2 – Dimensions of Context

As aforementioned, three main dimensions of context are widely extended in the research community: User, Environment and Platform. The purpose of this research question is to discover how each dimension of context is addressed and taken into account in the different proposals.

Fig. 10 presents an overview of the different dimensions of context considered by each selected paper. Most of the selected papers (341 papers, i.e. 96.06%) consider the Platform in some way, followed by the Environment (325 papers, i.e. 91.55%), the User (306 papers, i.e. 86.20%), and, finally, Other dimensions of context (186 papers, i.e. 52.39%).
There are few papers that only consider one dimension of context, such as P53 and P137 (User), P83 (Environment), or P322 (Platform). Others only treat two of them, such as P40 (User and Environment), P81 (Environment and Other dimensions), P284 (User and Platform), or P346 (Platform and Environment). On the other hand, other papers deal with all three main dimensions of context (User, Platform and Environment), such as P109 or P295, while others also consider other dimensions of context in addition to the three usual dimensions, such as P283 or P171. Next, we offer some examples of papers regarding these context dimensions.

There are many selected papers that deal with the three principal dimensions of context, i.e. User, Environment and Platform. For example, P314 takes into account the user needs and preferences, the situation or the available resources, and also the presence and GPS location information for adapting services and applications. P166 revolves around mobile devices and their resources, such as memory, processing power, battery level or network features to adapt the system. It also takes into account the environment, the application characteristics and the user preferences at runtime for dynamic adaptation of services. P139 proposes a data gathering layer in its architecture for collecting and processing the profiles of the users (gender, age, job, hobby, etc.), or for managing data, such as time, location and temperature. By doing so, this proposal is able to predict the user preferences and provide personalized services or products based on those preferences.

Many papers address other dimensions of context, in addition to the main ones. For instance, some papers, such as P116 and P264, take into account the user activity he/she is doing for guiding the adaptation. P82 considers location of an entity within the environment, current temperature, time, ambient light or noise level, as well as physiological factors of the user, and the activity the person is involved in, such as reading or talking, to carry out the adaptation. It also considers software components attributes, e.g. load of the CPU, state of the files in the file system, or state of the application. P168 in addition to location, time, network characteristics, user preferences, user profile, or system services, also considers the ongoing activities of the users. P315 treats as context for the adaptation the user activity, geo-location, speed and direction, battery level, time, available networks, favourite places, and means of transport. P253 considers many context information, e.g. location of a device, occurrence of an event, room temperature, number of people in a room, medical profile, blood glucose, blood pressure, heart rate or body temperature of a user. It also manages both daily activities of the user being monitored, e.g. eating, and abnormal activities, e.g. falling down, that should be detected in a reasonable time. P48 proposes the following context dimensions: user, device, network, activity, service, location, and resource. P43 presents a framework for improving recommender systems based on the user activity, his/her preferences and profile, information regarding the different devices used, and the location of the user, light, noise, temperature of the environment or time of the day, among others.

Other approaches pay attention to the historical data as another dimension of context, that is, past information related to the context of use, e.g. the interaction history of the user. For example, P276 exploits the learning history of the user for adapting his learning activities. P81 employs words used recently, user’s previous conversations, current time, date, and user’s location to select conversation topics, among others. P54 presents a system that takes into account location, temperature, light, noise or humidity level of the environment, behaviour and preferences of the user, battery level of the devices, among others, as well as the user activity and his historical information, represented as a data set generated using a variety of sensors. By analysing all this information, the system is able to learn user behaviours in an office environment in order to use the inferred rules to populate a user model and support appropriate proactive behaviour, e.g. turning on the fan under appropriate conditions.

Some papers exploit the user’s interaction task for the adaptation process. For example, P254 presents a software that supports different sensor types, including location and user interaction sensors, to monitor the active application the user is involved in, among others. P81, in addition to historical data, uses the user’s interaction task to support a word predictor to foresee which word a user is typing using the letters that have been typed so far. P131 proposes a context-aware system for museums that has the ability of monitoring the continuous changes of the user’s interests during his interaction with the system. Finally, P143 presents an Affect and Belief Adaptive Interface System (ABAIS) that exploits a variety of user assessment methods (diagnostic tasks, physiological sensing, etc.) to determine which adaptation approach is the most appropriate.

4.3.1 RQ 2.1 – User dimension of context

We discovered a great variety of terms used for characterizing the user of a context-aware system. These terms are represented in Fig. 11, being the top ten most frequently used: preferences, regarding the language, colour, type of content, interaction modality, etc.; profile of the user including name, age, gender, email address, etc.; health state refers to any information related to the health conditions or medical data of the user; physiological data, e.g. heart rate, blood pressure, skin conductance, etc.; role of the user played in the system/community; identity of the user; social profile, e.g. family, friends, doctors or interests; name of the user, as part of his/her profile; behaviour of the user when using the system, e.g. motivated, bored, stressed, etc.; and finally, heart rate of the user related to his/her physiological data. Note that we have
included the terms name and heart rate in this ranking despite being part of the profile and physiological data of the user, respectively, because these terms are widely used in context-aware systems.

Table 10 shows the number of papers that make use of these top ten words about the user. The terms most frequently used are preferences, which appears in 58.59% of the selected papers, and profile, with a 41.41% of use.

Table 10. Top ten terms in User dimension of context

<table>
<thead>
<tr>
<th>User term</th>
<th>No. of papers</th>
<th>% of papers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preferences</td>
<td>208</td>
<td>58.59</td>
</tr>
<tr>
<td>Profile</td>
<td>147</td>
<td>41.41</td>
</tr>
<tr>
<td>Health state</td>
<td>60</td>
<td>16.90</td>
</tr>
<tr>
<td>Physiological data</td>
<td>36</td>
<td>10.14</td>
</tr>
<tr>
<td>Role</td>
<td>35</td>
<td>9.86</td>
</tr>
<tr>
<td>Identity</td>
<td>27</td>
<td>7.61</td>
</tr>
<tr>
<td>Social profile</td>
<td>26</td>
<td>7.32</td>
</tr>
<tr>
<td>Name</td>
<td>26</td>
<td>7.32</td>
</tr>
<tr>
<td>Behaviour</td>
<td>25</td>
<td>7.04</td>
</tr>
<tr>
<td>Heart rate</td>
<td>20</td>
<td>5.63</td>
</tr>
</tbody>
</table>

4.3.2 RQ 2.2 – Platform dimension of context

While characterizing the platform of context-aware systems, we have also detected a great heterogeneity among the terms used in the selected papers. However, there is some information related to the platform more widely used, as Fig. 12 shows: devices used and all their related information, e.g. type, size, capabilities, etc.; sensors in charge of collecting data from the context; services offered by the system, e.g. weather services or traffic information services; network and its relevant information, e.g. speed, bandwidth or connectivity; screen size of the device being used; memory, e.g. capacity, level or availability; battery level of the device; applications running or available in the system; interaction modality/ies provided by the system; and finally, bandwidth of the network in use. Note that we have included the terms sensors, screen size, memory, battery level or bandwidth in this ranking despite being part of the devices and network information of the platform because they are widely used in context-aware systems.
Table 11 presents the number of papers that make use of these top ten words related to the platform. As can be seen, the terms most frequently used are devices, being used by 56.06% of the selected papers, sensors, with an appearance rate of 48.73%, services used in 36.34% of the selected papers, and network, detected in 31.27% of the papers.

<table>
<thead>
<tr>
<th>Platform term</th>
<th>No. of papers</th>
<th>% of papers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Devices</td>
<td>199</td>
<td>56.06%</td>
</tr>
<tr>
<td>Sensors</td>
<td>173</td>
<td>48.73%</td>
</tr>
<tr>
<td>Services</td>
<td>129</td>
<td>36.34%</td>
</tr>
<tr>
<td>Network</td>
<td>111</td>
<td>31.27%</td>
</tr>
<tr>
<td>Screen size</td>
<td>48</td>
<td>13.52%</td>
</tr>
<tr>
<td>Memory</td>
<td>48</td>
<td>13.52%</td>
</tr>
<tr>
<td>Battery level</td>
<td>46</td>
<td>12.96%</td>
</tr>
<tr>
<td>Applications</td>
<td>43</td>
<td>12.11%</td>
</tr>
<tr>
<td>Interaction modality</td>
<td>41</td>
<td>11.55%</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>39</td>
<td>10.99%</td>
</tr>
</tbody>
</table>

4.3.3 RQ 2.3 – Environment dimension of context

Similarly to the previous dimensions, we have encountered many terms related to the environment. Nevertheless, there are also some of them that are very recurring (see Fig. 13): location of the user/entity; time at which the interaction takes place; temperature of the place; light of the ambient; noise of the ambient; weather information of the current day; date; humidity level; gesture the user is doing; and orientation of the user/entity. Note that we have separated temperature and humidity from the weather category as they are highly employed in an isolated way for characterizing the environment of context-aware systems.
Table 12 presents the number of papers that use these top ten words for characterizing the environment. As can be observed, the term most frequently used is by far the location, being used by 83.38% of the selected papers, followed by the time, 38.87%, temperature with a 22.25%, and light with a 20.56%.

Table 12. Top ten terms in Environment dimension of context

<table>
<thead>
<tr>
<th>Environment term</th>
<th>No. of papers</th>
<th>% of papers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>296</td>
<td>83.38%</td>
</tr>
<tr>
<td>Time</td>
<td>138</td>
<td>38.87%</td>
</tr>
<tr>
<td>Temperature</td>
<td>79</td>
<td>22.25%</td>
</tr>
<tr>
<td>Light</td>
<td>73</td>
<td>20.56%</td>
</tr>
<tr>
<td>Noise</td>
<td>51</td>
<td>14.37%</td>
</tr>
<tr>
<td>Weather</td>
<td>36</td>
<td>10.14%</td>
</tr>
<tr>
<td>Date</td>
<td>28</td>
<td>7.89%</td>
</tr>
<tr>
<td>Humidity</td>
<td>24</td>
<td>6.76%</td>
</tr>
<tr>
<td>Gesture</td>
<td>21</td>
<td>5.92%</td>
</tr>
<tr>
<td>Orientation</td>
<td>18</td>
<td>5.07%</td>
</tr>
</tbody>
</table>

4.3.4 Other dimensions of context

As aforementioned, we also wanted to determine if some proposals make use of other dimensions of context, in addition to User, Platform or Environment. In this sense, we have identified the following three dimensions of context: Historical data, User activity or Interaction task (see Section 2.4 for more details). Table 13 presents the number of papers that use these additional dimensions. Namely, the interaction task the user is doing with the system is used in 29.01% of the selected papers, the user activity the user is performing (e.g. walking, sleeping, shopping, cooking, etc.) in 18.31%, and the historical data in 13.52%.

Table 13. Other dimensions of context

<table>
<thead>
<tr>
<th>Dimension</th>
<th>No. of papers</th>
<th>% of papers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interaction task</td>
<td>103</td>
<td>29.01%</td>
</tr>
<tr>
<td>User activity</td>
<td>65</td>
<td>18.31%</td>
</tr>
<tr>
<td>Historical data</td>
<td>48</td>
<td>13.52%</td>
</tr>
</tbody>
</table>
4.4 RQ 3 – Evaluations of Context-Aware Adaptations

Results presented in this section focus on proposals that apply context-aware adaptations and that provide some mechanism to evaluate these context-aware adaptations, that is, to determine their quality. Therefore, we want to find out to what extent the evaluation of context-aware adaptations in context-aware systems is covered in the literature to estimate its maturity. As Fig. 14 shows, there are only 9 papers with some kind of proposal for assessing the adaptation, i.e. only the 2.54% of the selected papers, namely P10, P107, P141, P146, P200, P214, P295, P297 and P304.

Fig. 14. Selected papers with adaptation evaluation

P214 presents a framework that consists of, among others, a design space for evaluating the adaptation coverage in a reliable way. This design space is used to assess and compare the adaptation levels of different applications based on common criteria. However, other papers focus on evaluating the adaptation process by means of its usefulness, such as P107. This paper suggests a user-centred evaluation to verify the usefulness of the adaptations applied.

We have found other types of evaluation that assess the adaptation by means of a Graphical User Interface (GUI) or wizard that shows the user how the adaptation will change the system so he/she can accept or reject it in a visual manner. We have called this type of evaluation visual validation. For example, P146 presents a tool for the generation of context-aware adaptive software system implementations that supports a visual validation of the context-aware adaptive behaviour of the system through a GUI. The user can press the button Adapt to the Context Information Changes to see how the context changes impact the system. This evaluation feature enables the detection of missing and incorrect adaptation behaviours in a visual manner. P200 proposes a visual validation of the adaptation too. Its proposal is able to recognize the actual policy rule that causes an incorrect behaviour of the system in order to assist the user when he/she corrects the adaptation behaviour by means of a wizard. Finally, P295 proposes a widget to support the user while expressing his/her opinion about the current presentation topic. This is carried out by means of two buttons: the Wow button that reflects the user feelings whenever he/she is positively impressed by a specific presentation, while the Basta! button is used when the user is not interested in the current topic, stopping the current presentation.

We have also detected that other articles evaluate the adaptation by using the satisfaction of the user, usually through some type of survey or questionnaire. For instance, P10 evaluates the user’s satisfaction regarding the offered context-based recommendations. P141 measures the user’s satisfaction exploiting the perceived ease of use or perceived control, among others. P297 estimates the satisfaction degree between the adapted media selected by its proposed Content Adaptation Process and the media requested by the learner. Finally, P304 proposes the evaluation of the user’s satisfaction regarding functionality and adaptability by means of a survey.

Fig. 15 shows the different types of evaluation identified to assess the adaptations in context-aware systems: design space, usefulness evaluation, visual validation, and satisfaction evaluation. Note that there are quite a few papers, only 9, that deal with some kind of adaptation evaluation out of the selected papers. This lack will be further discussed in Section 5.
4.5 RQ 4 – Evaluations of Software Architecture Concepts

Finally, we wanted to find out what evaluations have been carried out up to now to validate the software architecture concepts proposed for context-aware systems. By doing so, we will be able to discover the degree of maturity of the different software architecture concepts. Therefore, we have categorized the different software architecture concepts found in the selected papers according to the type of evaluation presented by using the taxonomy of Wohlin et al. [57] for classifying Empirical strategies. We have also added the category Example for papers that do not present a formal empirical evaluation (see Section 2.4 for more details about this categorization).

Fig. 16 shows the results from the point of view of the evaluation type presented in each selected paper. The majority of the papers only include an Example (116 papers, i.e. 32.68%), followed by Case Studies (91 papers, i.e. 25.63%), Surveys (19 papers, i.e. 5.35%), and finally Experiments (17 papers, i.e. 4.79%). Thus, there are 112 papers among the selected ones (31.55%) that do not have any type of evaluation, validation or example, neither formal nor informal one, for assessing the SA concept proposed.

With regard to the Example category, most of the software architecture concepts use examples to illustrate and validate the proposals in an informal way regarding the following SA concepts: client-server architecture (P52 or P100), component-based architecture (P13, P37 or P283), middleware architecture (P290 or P298), distributed architecture (P264 and P312), model-driven architecture (P66 or P93), multi-layered architecture (P226 or P300), event-triggered architecture (P334), micro-architecture (P101), service-oriented architecture (P6 or P133), multi-agent system (P85 or P204), publish/subscribe architecture (P89), and peer-to-peer architecture (P4 or P217).

We have identified some Surveys for assessing the following software architecture concepts: client-server architecture (P206), component-based architecture (P121 or P125), distributed architecture (P194), model-driven architecture (P127), multi-layered architecture (P306 or P355), event-triggered architecture (P27), service-oriented architecture (P38 or P112), and multi-agent system (P19 and P207).

Some Case studies have been also carried out for evaluating the following software architecture concepts: client-server architecture (P118 or P346), blackboard architecture (P165), component-based architecture (P8, P126 or P134), middleware architecture (P31 or P78), distributed architecture (P194), model-driven architecture (P47 or P49), multi-layered architecture (P57, P192 or P339), event-triggered architecture (P209), service-oriented architecture (P48, P119 or P253), multi-agent system (P45, P61 or P230), publish/subscribe architecture (P44 or P95), peer-to-peer architecture (P253), pipe-and-filter architecture (P268), and plug-and-play architecture (P99).

Finally, we have found a few Experiments for the validation of the following software architecture concepts: component-based architecture (P86 and P255), model-driven architecture (P127), multi-layered architecture (P10, P23, P171, P196, P228, P260, P286), service-oriented architecture (P281), and multi-agent system (P23, P113, P265, P304).
Note that we have found neither formal nor informal evaluation for assessing aspect-oriented architectures. Next, some of the most referenced selected papers are highlighted to provide some insights about the evaluation strategy applied to validate their software architecture proposals.

P315 uses just an example to demonstrate and validate its multi-layered architecture. It presents a typical use of the infrastructure in a mobile healthcare application scenario. On the other hand, P27 presents a survey for assessing the maturity of the event-triggered and multi-layered architecture proposed. It is the only paper that has applied and evaluated the proposal in a real industrial scenario. Therefore, it has been categorized as Evaluation research. First, the system was deployed in a hospital and it went into pilot use. Afterwards, a survey focusing on users’ experience while using the system was handed out to the doctors, and over 40 responses were collected.

Other papers have validated their proposals by means of case studies. P48 proposes a service-oriented architecture which has been evaluated by a case study applied in a medical system scenario and using concrete adaptation rules. P82 also provides a service-oriented architecture assessed by a case study. Authors show how the toolkit proposed can be used to design and build context-aware applications and how it supports reusability of components, evolution of applications, and acquisition of complex context. P139 suggests a framework based on the multi-agent and multi-layered approaches. This framework has been validated by using a case study to show its feasibility. P168 presents a multi-agent system. Authors designed and implemented a prototype as a case study to test the applicability, scalability and context effectiveness in everyday activities. Finally, P253 makes use of several paradigms, such as peer-to-peer, multi-layered and service-oriented architecture. To validate the proposal, authors used a case study by deploying a prototype in their research laboratory and using different real scenarios, such as a smart home, an office or a shop.

To conclude, P260 proposes a context-aware system supported by a multi-layered infrastructure. Its effectiveness has been proved by conducting experiments with a complete implementation of the system with some available sensors and an Android-based handheld device that acted as the host for the main activity recognition module.

### Discussion

In this section, an analysis of the results described in the previous section regarding the Research Questions is presented, in order to foresee future directions in this research field (Section 5.1). Afterwards, some implications for researchers and practitioners are discussed (Section 5.2).

#### 5.1 Analysis and Synthesis of Results

This section presents an analysis and synthesis of the results described in the previous section.
5.1.1 Publication venues, years and countries

As shown in Table 9, only around 20% of the selected papers were published in the top ten publication venues. The rest of the papers were published in a great variety of venues (see appendix A.2). One of the reasons for this issue is the heterogeneity and number of terms we have considered while performing the automated search. This great amount and variety of words implies a wide number of results, from venues focused on ubiquitous and pervasive computing, to others related to mobile and wireless communications. We detected a high number of venues related to ubiquitous and pervasive computing; context modelling, especially user modelling; smart environments and ambient intelligence; adaptive systems; and HCI in general. These topics are directly related to context-aware so it is obvious that many papers were published in this type of venues. On the other hand, there were also other venues that revolved around other topics not related to context-aware, but to software architectures. Evidently, the results depended on basically the words used for defining the search string and, as aforementioned, this search string included many terms associated to software architectures, context-aware and context dimensions (see Section 3.2.3).

As Fig. 3 illustrates, a clear evidence has been found regarding the growing interest of the research community in context-aware systems and their supporting architecture. It can be seen that, although this topic emerged many years ago (seminal papers were published in 1999), it has been since 2007, approximately, when the interest has increased significantly until nowadays. One of the main reasons for that interest on context-aware systems is the enormous benefit they provide and the wide availability of sensors at a reduced cost. Context-aware systems are usually implemented in smart environments where users interact through an interface transparent for them. This ubiquitous and transparent way of interaction is becoming a very attractive paradigm, especially when dealing with older or impaired people. Furthermore, there is an increasing attention on healthcare since the worldwide population is aging progressively, among other issues. In this way, context-aware systems have emerged as an appropriate solution for assisting, especially, older people in their daily life.

On the other hand, it can be observed in Fig. 4 that Asian countries, such as South Korea and China, lead the ranking of countries more interested in context-aware systems, followed by Spain, Italy and USA. However, as aforementioned, the advantages of context-aware systems are recognized worldwide so papers about this topic are published all over the world.

In conclusion, observing the tendency, context-aware systems will remain one of the main research fields in future years, involving more and more countries worldwide and possibly more and more publication venues.

5.1.2 Research topics, applications domains and research types

The research topics most frequently used, as Fig. 5 depicts, are: context management, context inference, context modelling, context gathering and context-aware applications development. These results highlight that most of the selected papers focus on handling the context of use in some way, as well as on developing context-aware applications. This has positive implications. For example, most architectural proposals take care of context in some way, one of the main aims of this study.

Furthermore, we also aimed at identifying the most frequent application domains of context-aware systems (see Fig. 6) that were: healthcare, smart environment, smart home, mobile environment, older people, Ambient Assisted Living, e-learning, guide-assistant and recommendation systems. This supports our previous finding, that is, that healthcare is the most important application domain for context-aware systems. Moreover, the vast majority of these systems focus on building smart spaces for older people to assist them in their daily life (ambient assisted living). In this line, we detected that many papers whose proposals were developed to assist people while learning, using a computing system (e-learning), or to guide people over through any type of smart environment, such as a museum, or to provide users with some type of recommendation.

Regarding the research types of the selected papers, they were classified according to Wieringa et al. [55] taxonomy (see Fig. 7). Most of the selected papers were categorized as Solution proposals (94,37%), followed by Validation research (30,14%), Experience paper (1,69%), Evaluation research (0,28%), and finally Philosophical paper (0,28%). These results are quite disturbing as many selected papers were only solution proposals, without any type of validation or evaluation, neither formal nor informal. Only 30,14% of the selected papers offered some kind of laboratory validation which means that the context-aware systems were developed, but not deployed nor evaluated in a more real and industrial setting. P27 was the only one classified as Solution proposal and Evaluation research, since its proposal was developed and evaluated in a real and industrial scenario. First, the system was deployed in a hospital and it went into pilot use. Afterwards, a survey focusing on users’ experience while using the system was handed out to the doctors, and over 40 responses were collected.

Thus, we have found that there are many proposals attending to develop context-aware systems for solving a concrete problem. However, we have detected a lack of maturity as many of these works are not validated at all. In this sense, future directions in context-aware computing should go on performing more evaluations on such systems in order to
implement them in a real environment and provide additional feedback about the effectiveness, usefulness, etc. these systems can offer.

5.1.3 Context dimensions covered by software architecture concepts

As Fig. 17 illustrates, the three main dimensions of context (User, Platform and Environment) are covered by different software architecture concepts. With respect to other dimensions of context, such as historical data, user activity or interaction task, all the analysed architectural approaches cover some of them, except for the aspect-oriented architecture. Therefore, it can be claimed that architectural concepts used for developing context-aware systems take into account the context of use in some way. Hence, context is not addressed as an isolated issue, but it is included as a key part of the software architecture. It has been observed that the main purpose was to reflect that some actions would be executed in the system, depending on the current context. Furthermore, it was also detected that most of the systems analysed react to context changes using rules defined following an Event-Condition-Action (ECA) format. Hence, systems perceive a
context change by means of an event, then systems check whether the condition related to such event was satisfied in order to react using the corresponding action. For example, in a smart home scenario, an ECA rule could be defined so that, if the system detects that a person is falling down (event), and such subject is an old person (condition), an alarm call is made to the emergency centre, the caregivers and the family of the person (action). Usually, the necessary infrastructure to exploit the ECA approach, and the process to obtain valuable and high-level context information of data collected from different sensors (e.g. that the user is laying on the floor because of a falling) are usually considered during the design of the software architecture of the context-aware system. This is done by adding one or more architectural components that are specifically in charge of this treatment of the surrounding context.

Regarding the software architecture concepts used to develop context-aware systems, most of the proposals employ, as Fig. 17 depicts, multi-layered architectures, component-based architectures, SOAs and MASs since they appeared as the most suitable ones for context-aware systems. The main reason for the popularity of these software architecture concepts is their simplicity, ease of use, and that they provide useful and effective mechanisms to facilitate the adaptation of the system to context conditions. For instance, as seen before in Section 4.1.4, many context-aware systems are focused on adapting the services offered to the user so that most of them employ a SOA approach since its main foundation is the support of services. On the other hand, the less popular software architecture concepts are aspect-oriented, blackboard, distributed, micro-architecture, pipe-and-filter, and plug-and-play.

It is interesting to highlight that, generally, all these architectural approaches deal with all the dimensions of context, but with some particularities. In general, the platform dimension is the most widely dealt with for adapting the system, followed by the environment, the user and other dimensions. Even though the environment dimension was identified as the second one, there is no significant difference regarding the number of papers between this and the user dimension. Therefore, despite these little differences, we confirm that all context dimensions, including other ones, are more or less considered in a similar way by every software architecture concept.

In conclusion, most of the selected papers, independently of the architecture schema adopted, when they say they take into consideration the current context for adapting the system, such context is referred most of the time to the user, the platform and the environment, and least often to other dimensions of context.

5.1.4 What is adapted?

After analysing all the selected papers, we noticed that they were focused on adapting specific elements in the system so we looked for those elements that are adapted since we considered this information very useful and a cornerstone when designing and developing context-aware applications. As seen in Fig. 8, services are overwhelmingly the component of context-aware systems most frequently adapted, followed by the content presented in the user interface, as well as the presentation of these contents.

The main reason for this is that most of the context-aware systems analysed were designed following a service-oriented approach. It is interesting to highlight that they provide services as assorted as calendar, weather, location, time, news or traffic information ones. Moreover, since many selected works were focused on the healthcare domain, alarm calls appeared here too as a relevant element to be adapted in context-aware systems. An alarm call referred to any type of message delivered to caregivers, doctors or patient’s family when an emergency situation was detected. Patient was usually an old person and/or an impaired one who still wants to live alone, but needs to be monitored to prevent them from severe injuries.

From these results, we can claim that not many context-aware systems were interested in adapting or reconfiguring the software architecture, despite changes in the environment were required. As mentioned before, there is no doubt that context-aware systems should pay attention to its architecture and the way it is designed. In this sense, they could take advantage of the support provided for architectural changes to offer adaptation to the context of use. Therefore, this significant gap in context-aware computing should be covered in future research, requiring more effort in order to align the adaptation at architectural level (changes in the configuration of architectural components) and at HCI level (changes in the interaction modality or the user interface).

5.1.5 RQ 1 – Software architecture concepts

RQ1 focuses on identifying which architectural concepts were used in existing context-aware systems in order to provide an overview of primary studies aiming at defining and/or exploiting those software architecture concepts. Fig. 9 summarizes the results that answer this research question. One of the problems faced to answer this RQ was that not all the selected papers describe clearly the architecture patterns applied. Sometimes, such software architecture concepts had to be identified from the context of the paper, while sometimes, there was no way to detect a well-defined architecture approach. In this last case, 55 papers, which represents 15.5% of the total of selected ones, did not employ any software architecture concept at all. Most of this 15.5% of papers were published in HCI and ubiquitous computing venues, and others were related to mobile networks and wireless communications, but not in venues related to Software Engineering
or Software Architecture. These findings demonstrate our first intuition about the existing gap between HCI and SA. Most of the papers focused on context-aware adaptations, but do not consider the software architecture at all since they neglect partially or completely any possible change in the system architecture during the adaptation process.

As illustrated in Fig. 9, multi-layered architectures, component-based architectures, SOAs and MASs were the most widely used for context-aware systems. On the contrary, the software architecture concepts less popular were micro-architecture, plug-and-play, pipe-and-filter, blackboard and aspect-oriented architectures. It is worth noting that many works use at the same time several software architecture concepts for structuring their architectural elements. We found many different combinations, but there were three quite repeated (see Table 14), which involved the most popular ones: multi-layered architecture and MAS; multi-layered architecture and SOA; and finally multi-layered and component-based architecture.

Table 14. Top three combinations of software architecture concepts

<table>
<thead>
<tr>
<th>Combination of software architecture concepts</th>
<th>No. of selected papers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-layered architecture + MAS</td>
<td>11</td>
</tr>
<tr>
<td>Multi-layered architecture + SOA</td>
<td>10</td>
</tr>
<tr>
<td>Multi-layered + component-based architecture</td>
<td>6</td>
</tr>
</tbody>
</table>

It is worth noting that the application of multi-layered architectures was considerably extended in context-aware systems. The main benefit of using a multi-layered architecture here is the separation of concerns since it enables the creation of different layers, organized hierarchically, where each one is in charge of providing a different functionality of the context-aware system. In this sense, the most popular elements for representing this functionality in the different layers were agents (MAS), services (SOA) and components (component-based architecture), as observed in Table 14.

![Fig. 18. Context treatment in context-aware systems](image)

After analysing the selected papers, we found that some of these elements in the architecture of context-aware systems were commonly used to provide specific context-aware functionality. For example, we found elements for adapting the UI, namely for adapting the content information of the UI, and others for handling services associated to the context of use or any other relevant aspect of the system.

We found a common schema, illustrated in Fig. 18, for dealing with context in context-aware systems. Some of the steps that make up this process are often turned into components, services or agents in the software architecture responsible for such context handling. Context modelling is the first step and consists in establishing what the representation of context information will be, e.g. by means of an ontology or a modelling language such as OWL or RDF. Second stage corresponds to context gathering, i.e. obtaining context data when required or subscribing to event notifications from multiple sources, usually demanded by services and applications (adapted from [41]). All these context data should be modelled as previously established at the first stage (adapted from [48]). Third, some context information cannot be obtained directly, but by processing low-level context information. This task is performed by the context inference part of the system, usually by applying some reasoning mechanism, as mentioned before. For example, to know whether a person is travelling and which means of transport he is using, a reasoning must be carried out using other context information, such as Wi-Fi access points or calendar information (adapted from [50]). Here, it is interesting to highlight that some context-aware systems present a single component/service/agent in charge of both context gathering and inference, usually called context discovery. Once the context inference is executed, and the system has all the context information it needs, the context delivery stage is initiated.
Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation

necessary context information, then context management is performed for solving conflicts among contradictory sources of context information, processing the context information previously gathered (adapted from [11]). Finally, the last step is context delivery, i.e. when an event occurs and matches any rule previously specified, then some context information is provided to the application or service that requires it, or simply when such application or service requires it (adapted from [19]). It is worth noting that some of this context information can be stored in, e.g. a database, for future context requests. This information stored refers to the historical context of the system.

However, not all the selected papers supported all these stages of the context treatment process so its software architecture normally reflected only those steps they were focused on. More details about the software architecture concepts and terms related to context treatment are provided in Sections B. 2 and B. 3, respectively.

5.1.6 RQ 2 – Dimensions of context

This research question revolves around discovering how software architecture concepts proposed for context-aware systems considered the three main dimensions of context (User, Platform and Environment) to support adaptation. We have also included in our study the context dimension other since we noticed many authors exploit other information, e.g. historical context, user activity or interaction task. Fig. 10 presents an overview of the different dimensions of context considered in the selected papers. Most of them (96.06%) considered the Platform in some way, followed by the Environment (91.55%), the User (86.20%), and, finally, Other dimensions of context (52.39%). This confirms that the three main dimensions of context are more or less equally dealt with by context-aware systems. This is highly important, since the more dimensions and terms related to context are considered by the system, the more appropriate and correct the adaptations provided will be, because, although context-aware systems will become more complex, they will be able to react to much more events and changes in the surrounding context, avoiding that users refuse to use the system.

One of the most time-consuming tasks of this mapping study was to extract every context concept from every paper, because most of the selected papers did not present nor describe with enough clarity those context concepts by using, for instance, a table, a figure or just an enumeration. On the contrary, most of the time these context terms were hidden through the text and distributed all over the paper among the different sections. Furthermore, another drawback here was the heterogeneity of the different context terms. Every author uses his/her own words to refer to the different context terms and dimensions, thus hindering the task of classifying them into the correct context dimensions. This led us to define a glossary with different context terms and synonyms that is presented in Section B. 1.

We only found some works that concisely presented and modelled such context dimensions. For example, P65 proposed a Context Dimension Model for defining at design time the possible contexts of use of a given domain of interest by using a hierarchical structure. This structure is made up of context variables, e.g. time, location, current company; and the set of values for such context variables, e.g. evening, Spain, with friends. Context variables values perceived at runtime by means of sensors trigger the activation of a given context. P220 also proposed a context model for categorizing context entities using agents, devices, environment, location and time. Location and time were kept separated from other concepts to emphasize the spatial and temporal aspects of the ubiquitous environment. Finally, P99 distinguished the variety of context elements along with how such context information was exploited to adapt the application in the form of a table. For instance, it specified that “when the battery level is extremely low, the application terminates in a graceful way to avoid data loss. Also the interaction with the user is adapted to his/her preferences, e.g. vibration on/off”.

As seen in Section 4.3.1, we discovered a great variety of terms used for characterizing the user of a context-aware system. Fig. 11 and Table 10 presented those results. The preferences of the user represented the most frequent element that context-aware systems used for the adaptation, along with the profile and information related to the health state or physiological data, especially the heart rate, which determined the current state of the user. We believe that user preferences have become a key element for adapting the system, since they explicitly exhibit the likings of the user regarding many aspects, e.g. the language, colour, type of content, interaction modality, etc. Therefore, if the system is able to adapt its behaviour depending on such preferences, the success of the adaptation process is almost ensured, i.e. there is a strong likelihood that the user will not reject the adaptation proposed, improving, thus, the quality of adaptation desired for every context-aware system. It is also evident that handling at runtime any information about users’ health conditions is quite valuable since healthcare appeared as one of the most important issues worldwide, as previously mentioned.

On the other hand, as seen in Section 4.3.2, we also detected a high heterogeneity of the terms used in the selected papers for characterizing the platform of a context-aware system. Fig. 12 and Table 11 showed those results. Nevertheless, devices and sensors were in the first positions of the ranking of most used concepts. In context-aware systems, devices in general, and sensors in particular, are very important elements. Sensors are essential as they are in charge of monitoring at runtime many context information, from location (hardware sensors) to system logs (software sensors). Furthermore, almost all the selected papers consider them in the design of their architecture, which demonstrates their importance in context-aware systems. But also other types of devices, such as smartphones, computers, monitors, etc., were considered
for adaptation purposes. Many context-aware systems adapt their behaviour attending to network bandwidth or connectivity, screen size, available memory, battery level, or interaction modalities supported by the devices.

Section 4.3.3 presented many features for determining the environment of a context-aware system. Fig. 13 and Table 12 depicted those results. Some of the top terms were location, time, temperature, light or noise. But especially, the location of the user/entity in the interaction place with the 83.38% of appearance rate was the most used one. Many papers include in their architectural designs components exclusively used to locate elements in the environment, such as GPS or GSM. Furthermore, the majority of those works do not talk about context-aware systems, but only about purely location-aware systems.

To conclude, as mentioned in Section 4.3.4, we also wanted to determine if some proposals use other dimensions of context, different from User, Platform or Environment. We distinguished other three dimensions of context widely used by the context-aware research community: Historical data, User activity and Interaction task. Table 13 presented those results. Namely, the interaction task the user was doing with the system was used in 29.01% of the selected papers, the user activity the user was performing (e.g. walking, sleeping, shopping, cooking, etc.) in 18.31%, and historical data in 13.52%. Therefore, it is interesting to highlight that almost one-third of all selected papers also paid attention to other dimensions of context for adapting the system, making them more rigorous and providing a higher quality of adaptation than others which not considered these other dimensions at all. Moreover, in addition to detect a broad consensus regarding the main context dimensions (User, Platform and Environment), we also detected such consensus when considering other dimensions of context. In this way, we were able to classify context data related to Historical data, User activity and Interaction task, but defining them previously in a more formal way (see Section 2.4 for more details).

5.1.7 RQ 3 – Evaluations of context-aware adaptations

This research question focus on discovering to what extent the evaluation of context-aware adaptations in context-aware systems was covered in the literature. The results of this research question are presented in Fig. 14 and Fig. 15. Surprisingly, there were only 9 papers with some kind of proposal for assessing the adaptation, i.e. only the 2.54% of the selected papers, namely P10, P107, P141, P146, P200, P214, P295, P297 and P304. After a deep analysis, we classified them into the following categories for assessing the adaptation:

- Design space evaluation. To assess and compare adaptation levels of different applications based on unified criteria.
- Usefulness evaluation. A user-centred evaluation to verify the usefulness of the adaptations applied.
- Visual validation. To assess the adaptation by showing how it will change the system, usually by means of a Graphical User Interface (GUI) or wizard.
- Satisfaction evaluation. Evaluation of the adaptation by taking into account the satisfaction of the user, usually through a survey or questionnaire.

As aforementioned, every software artefact, such as a user interface or a software architecture, should guarantee a certain degree of quality when delivered in order to prevent the rejection of the final user. Therefore, more concretely, context-aware adaptations should also ensure a certain degree of quality when they are applied to the system at runtime. In this sense, those proposals that apply context-aware adaptations should provide some mechanisms to evaluate them. Nonetheless, we have seen that almost none of the selected papers, except 9 works, considered such evaluation of the quality of adaptation. So, we have confirmed that the maturity of the proposals for adaptation is negligible. Hence, there is a huge gap here to be covered during next years by the research community in order to provide quality context-aware systems.

5.1.8 RQ 4 – Evaluations of software architecture concepts

This RQ was defined to find out what evaluations had been carried out until now to validate the software architecture concepts proposed for context-aware systems. Fig. 19 and Fig. 16 summarize the results to answer this RQ. We categorized the different software architecture concepts found in the selected papers, depending on the type of evaluation presented by using the taxonomy of Wohlin et al. [57]. We also added the category Example for papers that did not present a formal empirical strategy, but provide some example about the proposal.

As Fig. 19 depicts, multi-layered architecture, component-based architecture, SOA and MAS were the software architecture concepts having a higher level of maturity. These four architectural concepts were overwhelmingly the most broadly evaluated in the literature. They all were validated by means of examples, surveys, case studies and experiments, but not in the same number. For the multi-layered architecture, examples were the most widely user form of evaluation, followed by case studies, experiments and surveys in the last position. For component-based architecture, the situation was very similar, but it presented more surveys than experiments. SOA was evaluated most of the time by means of case
studies, followed by examples, surveys and only one experiment. Finally, MAS presented the same results as multi-layered architecture, being example the evaluation strategy more applied, followed by case study, experiment and survey. For these four architectural concepts, there is a good maturity, since we found many papers that used them in their proposals, along with a well-defined validation. Moreover, almost all the experiments identified were done to assess those four software architecture concepts.

Moreover, Fig. 19 illustrates an important fact. For most of the software architecture concepts, there were four or less evaluations, namely for: distributed architecture, P2P, event-triggered architecture, blackboard architecture, micro-architecture, pipe-and-filter and plug-and-play. Aspect-oriented architecture has no evaluation. So, in these cases, we
confirm that the maturity is marginal, because of the low number of evaluations and their type, less formal than for other approaches and without experiments at all.

Finally, we also discovered other software architecture concepts with a medium maturity level since they were not enough validated up to now, such as client-server architecture, MDA, middleware architecture and publish/subscribe architecture.

Fig. 16 presents another overview from the point of view of the evaluation type found in each selected paper. Most of the papers only presented an Example (32.68%), followed by Case Study (25.63%), Survey (5.35%), and finally Experiment (4.79%). Thus, the 31.55% of the total of selected papers did not present any type of evaluation or example, neither formal nor informal, for assessing the SA concept proposed.

In conclusion, we have discovered another important gap related to the lack of maturity regarding software architecture concepts when used for developing context-aware systems. We encourage researchers and practitioners to perform more exhaustive and formal validations over these proposals, describing in more detail the participants involved, their number, etc. By doing so, the desired level of maturity will be reached.

5.2 Implications for Researchers and Practitioners

This mapping study has revealed that context-aware systems are a growing important research topic, both for HCI and Software Architecture fields. As seen in Fig. 3, the interest of the research community in context-aware systems and their architecture has been increasing since 1999 up to now, especially since 2007. We have found 17 relevant papers (as part of the QGS) that propose some kind of framework for context-aware systems (see Section 3.2.2). These important works have influenced others through the years since 1999 until the present day. They have provided a better understanding about context and context-awareness, as well as many advices and recommendations when dealing with context-aware systems in terms of context and user modelling, UI adaptation, and adaptive infrastructures. Nevertheless, the key message derived from all of them is to always keep in mind the final user so the design of every context-aware system should be ultimately user-centred.

This study has proved that there was a real need to analyse which research works highlighted the use of context and its relationship with the software architecture in existing context-aware systems. By doing so, we have provided an overview about the different approaches used in context-aware systems, attending to align the adaptation at both architectural and HCI levels. In this sense, we have seen the effort invested in improving the use of context in context-aware systems and, especially, by means of their architecture. Moreover, we have also identified gaps existing up to now regarding the deficiencies and weaknesses of the systems when dealing with context and its different dimensions. Therefore, we have documented such effort and gaps by means of this mapping study so researchers and practitioners can make use of the 355 selected papers to improve their research and work. Namely, they can follow the guidelines proposed for modelling, retrieving and managing context and its different dimensions, as well as for knowing how to consider such context in the software architecture in order to provide a good adaptation experience to the final user. We have also offered a list of publication venues regarding context-aware systems (see appendix A.2), ordered by popularity, so researchers may use it for choosing an adequate venue to publish their work.

In this sense, as seen in Fig. 18, independently of the software architecture concept/s used for structuring a context-aware system, we have found some important and common elements which should be included in such infrastructure for ensuring a correct context treatment within our system. These components are: context modelling, context gathering, context inference, context management, and context delivery. Thus, researchers should pay attention to these elements and describe their behaviours and mechanisms for adapting the system. Furthermore, the most extended software architecture concepts for building context-aware systems are: multi-layered architecture, component-based architecture, SOA and MAS, and multiple combinations of them. Researchers have a guideline here too that will help them in the task of choosing the most convenient architectural patterns for developing their context-aware systems.

With regard to the context of use, researchers here have an extensive analysis of many works which have evidenced the most common terms used for characterizing such context. In addition, we have presented the context dimensions these context terms belong to, such as User, Platform and Environment, as well as other dimensions quite recurrent used in context-aware systems, such as Historical data, User activity and Interaction task. Thus, researchers can take into account this information in order to model context in an appropriate way and to perform adaptations according to these context variables.

Another implication for researchers is the need of providing some quality during the adaptation process. We have detected that there is almost no proposals that focus on evaluating the adaptations applied. This generates strong negative consequences as users may reject the adaptations, and eventually reject the whole system. Hence, more effort should be made at this point by researchers interested in context-aware adaptations with a certain degree of quality.
Apart from evaluating the adaptation, there is still a need to make researchers aware of the task of validating the software architecture concepts they used when building context-aware systems. Especially, if they do not use the most mature ones, i.e. multi-layered architecture, component-based architecture, SOA and MAS, those validation becomes an indispensable need.

In addition to the above implications, there are other implications derived from this mapping study. For example, researchers and practitioners can pay attention to the application domains identified that make use of context-aware systems (see Section 4.1.3), or the elements most usually adapted by such systems (see Section 4.1.4). Therefore, we encourage both researchers and practitioners to follow the guidelines and the different tips presented here for building context-aware systems and adaptations with a more than acceptable degree of quality.

6 Validity Threats

This mapping study addresses these validity threats: internal, external, conclusion and construct. They are described in the following sections.

6.1 Internal Validity

*Internal validity* [54] refers to the use of statistical analysis in order to know how well the data supports the results of the mapping study. In our case, we have only applied basic statistical analysis in the form of bar charts, bubble plots and tag clouds, among others. Thus, the internal validity of this study is negligible.

6.2 External Validity

*External validity* [54] refers to the strength of generalization claims of the study results. The results of this mapping study reflect the state of the art of software architectures for context-aware systems, involving both the HCI and Software Engineering research fields. We have identified 355 relevant papers under these research topics, after an exhaustive search process and after applying a well-defined selection criteria. Therefore, we believe that we have reached an acceptable external validity as our results can be enough generalized, always considering the selected period of time from 1999 to 2015.

6.3 Conclusion Validity

If other researchers replicate the mapping study, the same results would be obtained, that is *Conclusion validity* [54]. We have exhibited very clearly our selection criteria (see Section 3.1) for including or excluding papers in this mapping study. Therefore, we believe that the conclusions and results are valid and they can be obtained again by using the same research questions. Furthermore, we have followed a systematic mapping study process, along with an exhaustive and well-defined search process based on the Quasi-Gold Standard (see Section 2). The data collection has been also explained in detail, showing all steps followed and the results obtained at every step (see Section 3). Finally, we have clearly described terms and their synonyms in Appendix B.

6.4 Construct Validity

*Construct validity* [54] refers to the correct interpretation and measurement of the theoretical concepts. In our case, these theoretical concepts are software architecture, context-aware and context dimensions given that we wanted to identify the software architecture concepts used for the development of context-aware systems, and also which context dimensions were involved. We have spent quite a lot of time and effort in studying in detail each concept in order to prepare the most suitable search string for the automated search. We reviewed the literature to find the best words for each theoretical concept, and it was a very challenging task. We had to deal with works from two research fields really different: HCI and Software Architecture. With regard to the concept *software architecture*, we noticed that its name depended on the authors and of course on the research field. The architecture concepts identified ranged from a simple design with not many details to a more complex structure that detail clearly the architectural patterns and styles applied. In this manner, we had to incorporate to our search string several terms for referring to the architecture of a system in order to cover as many relevant papers as possible.

Something similar happened with other theoretical concepts. For *context-aware* we found as well many synonyms in the literature, since it is a popular and widely used concept nowadays, but with different interpretations. We encountered many words, such as adaptive, reactive, responsive and situated, used instead of context-aware, but they were too generic to use them in an isolated way. So that we had to accompany them with other words, like user interface, agent or
application, to refine the search. Hence, we included in our search string the most frequently used terms for referring to
context-aware.

Finally, regarding the context dimensions, we considered the three main ones, accepted by the research community: user, platform and environment. However, we also took into account other widely extended words for referring to context dimensions, such as human, device and task. By doing so, we covered the vast majority of context dimensions that we could find in the literature.

Therefore, we conclude that we have achieved construct validity as the final set of papers is complete thanks to the rigorousness while defining the search string (see Section 3.2.3). We have used not only the theoretical concepts for searching relevant papers, but also similar terms, which provides a high-level of confidence over the search process and its results. Moreover, we have performed the search process based on the Quasi-Gold Standard (see Section 3.2). In our case, we considered 17 papers, the most relevant and referenced ones regarding our research topic, for being part of the QGS. Finally, we verified our results from the automated search against this QGS, obtaining an acceptable quasi-sensitivity (see Section 3.2.5).

7 Conclusions

With this mapping study, we have achieved several goals. We have identified the different software architecture concepts proposed for designing and building context-aware systems with the aim of understanding the relationships between them. We have reviewed how context-based adaptations are taken into account in those software architecture concepts. Finally, we have also checked the maturity of both the existing methods for assessing the context-based adaptations, as well as the maturity of those software architecture concepts. To reach these objectives, we have answered some research questions with the aim of:

- Offering an overview to help researchers and practitioners to analyse which software architecture concepts are the most relevant and the most widely used in context-aware systems.
- Discovering how each dimension of context (User, Platform, Environment and others) is addressed and taken into account within the different architectural proposals to carry out the adaptations based on these dimensions of context.
- Discovering to what extent the evaluation of context-aware adaptations in context-aware systems is covered in the literature.
- Finding out what evaluations have been carried out until now to validate the software architecture concepts proposed for context-aware systems.

This mapping study has proved that there was a real need to analyse which research works highlighted the use of context and its relationship with the software architecture in existing context-aware systems. So we have provided an overview about the different approaches used in context-aware systems, attending to align the adaptation at both architectural and HCI levels. We have found out the effort invested in improving the use of context in context-aware systems and, especially, within their architecture. Moreover, we have also identified the gaps encountered up to now regarding the deficiencies and weaknesses of systems when dealing with context and its different dimensions.

Therefore, researchers and practitioners can make use of the results of this mapping study to improve their research and work. They can follow the guidelines proposed for dealing with context and its different dimensions, as well as for knowing how to consider such context for the design of the software architecture. By doing so, they will provide a good adaptation experience to the final user and they will develop context-aware systems and adaptations with higher degree of quality of adaptation.

Acknowledgements

This work was partially supported by Spanish Ministry of Economy, Industry and Competitiveness, State Research Agency / European Regional Development Fund under HA-SYMBIOSIS (TIN2015-72931-EXP) and Vi-SMARt (TIN2016-79100-R). It has also been funded by the Spanish Ministry of Education, Culture and Sport thanks to the FPU scholarship (FPU12/04962).

References

Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation


11. Past and Future of Software Architectures for Context-Aware Systems


Appendix A. Selected Papers and Venues

A. 1 Selected Papers

The 355 selected papers included in this mapping study are listed below (ordered by author). Notice that the papers included in the QGS (see Section 3.2.2) are in bold, e.g. [P2].


Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation


Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation


11. Past and Future of Software Architectures for Context-Aware Systems


Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation


11. Past and Future of Software Architectures for Context-Aware Systems


Considering Adaptation in the Development of Context-Aware Systems for Tele-Rehabilitation


A. 2 Publication Venues

The following table presents the publication venues of the 355 selected papers.

<table>
<thead>
<tr>
<th>Venue</th>
<th>Type</th>
<th>No. of papers</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Personal and Ubiquitous Computing</td>
<td>Journal</td>
<td>14</td>
<td>3.94</td>
</tr>
<tr>
<td>User Modeling and User-Adapted Interaction</td>
<td>Journal</td>
<td>11</td>
<td>3.10</td>
</tr>
<tr>
<td>Wireless Personal Communications</td>
<td>Journal</td>
<td>9</td>
<td>2.54</td>
</tr>
<tr>
<td>World Wide Web</td>
<td>Journal</td>
<td>7</td>
<td>1.97</td>
</tr>
<tr>
<td>International Conference on Embedded and Ubiquitous Computing</td>
<td>Conference</td>
<td>6</td>
<td>1.69</td>
</tr>
<tr>
<td>Procedia Computer Science</td>
<td>Journal</td>
<td>5</td>
<td>1.41</td>
</tr>
<tr>
<td>Journal of Systems and Software</td>
<td>Journal</td>
<td>5</td>
<td>1.41</td>
</tr>
<tr>
<td>International Conference on Universal Access in Human-Computer Interaction</td>
<td>Conference</td>
<td>5</td>
<td>1.41</td>
</tr>
<tr>
<td>Multimedia Tools and Applications</td>
<td>Journal</td>
<td>5</td>
<td>1.41</td>
</tr>
<tr>
<td>Expert Systems with Applications</td>
<td>Journal</td>
<td>4</td>
<td>1.13</td>
</tr>
<tr>
<td>IEEE Pervasive Computing</td>
<td>Journal</td>
<td>4</td>
<td>1.13</td>
</tr>
<tr>
<td>Communications Magazine</td>
<td>Journal</td>
<td>4</td>
<td>1.13</td>
</tr>
<tr>
<td>International Conference on Grid and Pervasive Computing</td>
<td>Conference</td>
<td>4</td>
<td>1.13</td>
</tr>
<tr>
<td>Applied Intelligence</td>
<td>Journal</td>
<td>4</td>
<td>1.13</td>
</tr>
<tr>
<td>The Journal of Supercomputing</td>
<td>Journal</td>
<td>4</td>
<td>1.13</td>
</tr>
<tr>
<td>International Conference on Pervasive Computing and Applications</td>
<td>Conference</td>
<td>3</td>
<td>0.85</td>
</tr>
<tr>
<td>Service Oriented Computing and Applications</td>
<td>Journal</td>
<td>3</td>
<td>0.85</td>
</tr>
<tr>
<td>International Conference on Human-Computer Interaction</td>
<td>Conference</td>
<td>3</td>
<td>0.85</td>
</tr>
<tr>
<td>International Journal of Smart Home</td>
<td>Journal</td>
<td>3</td>
<td>0.85</td>
</tr>
<tr>
<td>Conference on Human-Computer Interaction</td>
<td>Conference</td>
<td>3</td>
<td>0.85</td>
</tr>
<tr>
<td>Wireless Networks</td>
<td>Journal</td>
<td>3</td>
<td>0.85</td>
</tr>
<tr>
<td>Software and Systems Modeling</td>
<td>Journal</td>
<td>3</td>
<td>0.85</td>
</tr>
<tr>
<td>Virtual Reality</td>
<td>Journal</td>
<td>3</td>
<td>0.85</td>
</tr>
<tr>
<td>Universal Access in the Information Society</td>
<td>Journal</td>
<td>3</td>
<td>0.85</td>
</tr>
<tr>
<td>Journal of Network and Computer Applications</td>
<td>Journal</td>
<td>2</td>
<td>0.56</td>
</tr>
<tr>
<td>International Conference on Ubiquitous Intelligence and Computing</td>
<td>Conference</td>
<td>2</td>
<td>0.56</td>
</tr>
<tr>
<td>International Conference on Smart Homes and Health Telematics</td>
<td>Conference</td>
<td>2</td>
<td>0.56</td>
</tr>
<tr>
<td>International Conference on Computational Science and Its Applications</td>
<td>Conference</td>
<td>2</td>
<td>0.56</td>
</tr>
<tr>
<td>International Conference on Computer and Information Technology</td>
<td>Conference</td>
<td>2</td>
<td>0.56</td>
</tr>
<tr>
<td>International Conference on Pervasive Computing and Communications Workshops</td>
<td>Workshop</td>
<td>2</td>
<td>0.56</td>
</tr>
<tr>
<td>International Conference on Digital Ecosystems and Technologies</td>
<td>Conference</td>
<td>2</td>
<td>0.56</td>
</tr>
<tr>
<td>Hawaii International Conference on System Sciences</td>
<td>Conference</td>
<td>2</td>
<td>0.56</td>
</tr>
<tr>
<td>Human-Computer Interaction</td>
<td>Journal</td>
<td>2</td>
<td>0.56</td>
</tr>
<tr>
<td>International and Interdisciplinary Conference on Modeling and Using Context</td>
<td>Conference</td>
<td>2</td>
<td>0.56</td>
</tr>
<tr>
<td>Conference</td>
<td>Type</td>
<td>CiteScore</td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>---------------</td>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>Mobile Networks and Applications</td>
<td>Journal</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>International Conference on Convergence Information Technology</td>
<td>Conference</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Journal of Ambient Intelligence and Smart Environments</td>
<td>Journal</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Journal of Ambient Intelligence and Humanized Computing</td>
<td>Journal</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>International Conference on Mobile Ubiquitous Computing, Systems, Services, and Technologies</td>
<td>Conference</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>International Symposium on Handheld and Ubiquitous Computing</td>
<td>Conference</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Symposium on Applied Computing</td>
<td>Conference</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>International Conference on Ubiquitous Computing and Ambient Intelligence</td>
<td>Conference</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>International Conference on mobile technology, applications, and systems and International symposium on Computer human interaction in mobile technology</td>
<td>Conference</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>International Conference on Distributed, Ambient, and Pervasive Interactions</td>
<td>Conference</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Telecommunication Systems</td>
<td>Journal</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services</td>
<td>Conference</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>International Conference on Multimedia and Expo</td>
<td>Conference</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>International Conference on Mobile and Ubiquitous Multimedia</td>
<td>Conference</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>International Conference on Multimedia and Ubiquitous Engineering</td>
<td>Conference</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>International Conference on Network-Based Information Systems</td>
<td>Conference</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>International Conference on Computer Science and Software Engineering</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Computational Science and Engineering</td>
<td>Workshop</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Information Technology: New Generations</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Computational Intelligence and Communication Technology</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Computer Supported Cooperative Work in Design</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Communication Workshop</td>
<td>Workshop</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Cloud Computing and Intelligence Systems</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Computer Technology and Development</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Bio-Inspired Computing: Theories and Applications</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Autonomic and Autonomous Systems</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Computational Intelligence and Security</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Information Integration and Web-based Applications &amp; Services</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Hybrid Artificial Intelligence Systems</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Human Factors in Computing and Informatics</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Hybrid Intelligent Systems</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Genetic and Evolutionary Computing</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>International Conference on Engineering the Web in the Big Data Era</td>
<td>Conference</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Conference Name</td>
<td>Type</td>
<td>Impact Factor</td>
<td></td>
</tr>
<tr>
<td>--------------------------------------------------------------------------------</td>
<td>--------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>International Conference on Emerging Technologies and Factory Automation</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Cooperative Design, Visualization, and Engineering</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on E-Business and Telecommunication Networks</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Computer-Aided Industrial Design &amp; Conceptual Design</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Information Science, Electronics and Electrical Engineering</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Distributed Multimedia Systems</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Information Technology and Applications</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Distributed Applications and Interoperable Systems</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Developments in eSystems Engineering</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Dependable, Autonomic and Secure Computing</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Convergence and Hybrid Information Technology</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Computing for Sustainable Global Development</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on e-Health Networking Applications and Services</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Conference on Practical Applications of Agents and Multi-Agent Systems</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Conference on E-Commerce Technology and Enterprise Computing, E-Commerce and E-Services</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Computers, Communications and IT Applications Conference</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Computers and Electronics in Agriculture</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Computers and Electrical Engineering</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Computer-Aided Design and Applications</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Computer Supported Cooperative Work</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Computer Standards and Interfaces</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Computer Science - Research and Development</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Computer Animation and Virtual Worlds</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Innovations in Systems and Software Engineering</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Electronic Commerce Research</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Cognition, Technology and Work</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Cluster Computing</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Central European Journal of Computer Science</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>British HCI Group Annual Conference on People and Computers</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Brazilian Symposium on Multimedia and the Web</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>BMC Medical Informatics and Decision Making</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>BLED Proceedings</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Autonomous Agents and Multi-Agent Systems</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Conference/Conference/Workshop/Journal</td>
<td>Title</td>
<td>Type</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>Australasian User Interface Conference</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Asia-Pacific Web Conference</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Annales des Telecommunications/Annals of Telecommunications</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Computer</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>IEEE Distributed Systems Online</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Advances in Human-Oriented and Personalized Mechanisms, Technologies and Services</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Advances in Computer-Human Interactions</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Advanced Information Networking and Applications Workshops</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Advanced Communication Technology</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Advanced Aspects of Software Engineering</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference of the IEEE Engineering in Medicine and Biology Society</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference of the Chilean Computer Science Society</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference Human Society@Internet</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference and Workshops on Engineering of Computer-Based Systems</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Information Visualisation</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Conference on Ubiquitous Computing</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>IET Software</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Education and Information Technologies</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>IEEE Communications Magazine</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Human-centric Computing and Information Sciences</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Health and Technology</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Federated Conference on Computer Science and Information Systems</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Evolving Systems</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>European Symposium on Ambient Intelligence</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>European Conference on Software Maintenance and Reengineering</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>European Conference on Software Architecture</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>European Conference on a Service-Based Internet</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>ER Workshops on Advances in Conceptual Modeling: Challenges and Opportunities</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on AI, Simulation, and Planning in High Autonomy Systems</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Information and Software Technology</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Workshop on System Support for Future Mobile Computing Applications</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Conference on Mobile Web and Information Systems</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Mobile Information Systems</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Medical and Biological Engineering and Computing</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Conference/Workshop/Conference</td>
<td>Type</td>
<td>Year</td>
<td>Impact Factor</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------</td>
<td>------</td>
<td>---------------</td>
</tr>
<tr>
<td>Knowledge and Information Systems</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Journal on Selected Areas in Communications</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Journal of Universal Computer Science</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Journal of Supercomputing</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Journal of Software</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Journal of Intelligent Information Systems</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Journal of Information Science and Engineering</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Network Operations and Management Symposium</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Workshop on Ubiquitous Computing</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Nordic Conference on Human-Computer Interaction</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Workshop on Programming Multi-Agent Systems</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Workshop on Policies for Distributed Systems and Networks</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Workshop on Mobile Development Lifecycle</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Workshop on Enterprise Interoperability</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Workshop on Design, Specification, and Verification of Interactive Systems</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Workshop on Context-aware software technology and applications</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Workshop on Applications and Services in Wireless Networks</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Workshop on Agents and CyberSecurity</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Workshop on Advanced Parallel Processing Technologies</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Wireless Communications and Mobile Computing Conference</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Journal of Information and Computational Science</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Social Network Analysis and Mining</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Workshop on Middleware for Context-Aware Applications in the IoT</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Workshop on Context-Aware Adaptation of Service Front-Ends</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>UKSim European Symposium on Computer Modelling and Simulation</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Transactions on Systems, Man and Cybernetics</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Transactions on Computer-Human Interaction</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>The Visual Computer</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>The International Journal of Advanced Manufacturing Technology</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Symposium on Self-Stabilizing Systems</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Software Engineering Workshop</td>
<td>Workshop</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Multiagent and Grid Systems</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Soft Computing</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>International Symposium on Pervasive Computing and Applications</td>
<td>Conference</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>SIGPLAN Notices</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Scientific and Technical Information Processing</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Science of Computer Programming</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Science China Information Sciences</td>
<td>Journal</td>
<td>1</td>
<td>0.28</td>
</tr>
<tr>
<td>Conference / Journal Title</td>
<td>Format</td>
<td>Issue</td>
<td>Impact Factor</td>
</tr>
<tr>
<td>------------------------------------------------------------------------------------------</td>
<td>---------</td>
<td>-------</td>
<td>---------------</td>
</tr>
<tr>
<td>Scalable Computing: Practice and Experience</td>
<td>Journal</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>Real-Time Systems</td>
<td>Journal</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>Pervasive and Mobile Computing</td>
<td>Journal</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>Pacific Rim Knowledge Acquisition Workshop</td>
<td>Workshop</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>Pacific Rim International Workshop on Multi-Agents</td>
<td>Workshop</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>Pacific Rim Conference on Communications, Computers and signal Processing</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>Software - Practice and Experience</td>
<td>Journal</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Mobile Technology, Applications and Systems</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Recent Trends in Information Technology</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Practical Applications of Agents and Multi-Agent Systems</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on PErvasive Technologies Related to Assistive Environments</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Pervasive Services</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Pervasive Computing Technologies for Healthcare</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Parallel and Distributed Computing, Applications and Technologies</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Next Generation Mobile Applications, Services and Technologies</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Natural Computation</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Multimedia Communications, Services and Security</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>Wuhan University Journal of Natural Sciences</td>
<td>Journal</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Symposium on Web and Wireless Geographical Information Systems</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>Americas Conference on Information Systems</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Semantic Computing</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Mobile Technology, Application &amp; Systems</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Mobile Data Management</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Mobile Computing, Applications, and Services</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Mobile Computing and Networking</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Mobile Business</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Mobile and Ubiquitous Systems: Networking and Services</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Medical Biometrics</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Internet Technology and Applications</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Internet and Web Applications and Services</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Intelligent User Interfaces</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Conference on Mobile Wireless Middleware, Operating Systems, and Applications</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>International Joint Conference on Pervasive and Ubiquitous Computing and International Symposium on Wearable Computers</td>
<td>Conference</td>
<td>1</td>
<td>0,28</td>
</tr>
<tr>
<td>Event</td>
<td>Type</td>
<td>Year</td>
<td>Impact Factor</td>
</tr>
<tr>
<td>--------------------------------------------------------------</td>
<td>----------</td>
<td>------</td>
<td>---------------</td>
</tr>
<tr>
<td>International Conference on Intelligent Environments</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Symposium on Parallel and Distributed Processing with Applications</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Symposium on Intelligent Ubiquitous Computing and Education</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Symposium on Intelligent Signal Processing and Communications</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Symposium on Communications, Control and Signal Processing</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Symposium on Applications and the Internet</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Symposium on a World of Wireless, Mobile and Multimedia Networks</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Symposium Human Factors in Telecommunication</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Software Product Line Conference</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Journal of Pervasive Computing and Communications</td>
<td>Journal</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Conference on Research Challenges in Information Science</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Journal of Advanced Manufacturing Technology</td>
<td>Journal</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Conference on Self-Adaptive and Self-Organizing Systems Workshops</td>
<td>Workshop</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Conference on Wireless and Mobile Computing, Networking and Communications</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Conference on Wireless and Mobile Communications</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Conference on Web Engineering</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Conference on User Modeling</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Conference on Ubiquitous Computing</td>
<td>Workshop</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Conference on Systems, Man and Cybernetics</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Conference on Systems</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Conference on Software Engineering and Data Mining</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Conference on Software Engineering</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Symposium on Ubiquitous Computing Systems</td>
<td>Conference</td>
<td>0,28</td>
<td></td>
</tr>
<tr>
<td>International Journal of Distributed Sensor Networks</td>
<td>Journal</td>
<td>0,28</td>
<td></td>
</tr>
</tbody>
</table>
Appendix B. Glossary

In this section, it is presented a glossary of terms collected throughout the entire process of this mapping study. The main reason for preparing it was that, while analysing the different papers, we realized that different terms were used indistinctly for describing the same thing. This compilation of concepts will be useful for researchers interested in context-aware systems and their software architecture infrastructure. Next, these terms are presented, related to the dimensions of context (Section B. 1), the software architecture concepts (Section B. 2), and the research topics of the selected papers, among others (Section B. 3). Notice that all terms are ordered alphabetically.

B. 1 Dimensions of Context

Some terms were found associated to the dimensions of context and they are represented in Table 15. Notice that the first column corresponds to those terms most commonly used by the research community.

Table 15. Glossary of terms related to the dimensions of context

<table>
<thead>
<tr>
<th>Term</th>
<th>Similar terms</th>
<th>Context dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content</td>
<td>Information</td>
<td>Platform</td>
<td>Information formatted and presented in the UI, e.g. pictures, tables or text.</td>
</tr>
<tr>
<td>Gender</td>
<td>Sex</td>
<td>User</td>
<td>Physical/social condition of the user, such as female or male.</td>
</tr>
<tr>
<td>Historical data</td>
<td>Context history</td>
<td>Other</td>
<td>Information from the past, such as the interaction history of the user, usually used to know the evolution of the user and hence making changes and adaptations in the system that improve the user experience.</td>
</tr>
<tr>
<td>Light</td>
<td>Luminance</td>
<td>Environment</td>
<td>Brightness that comes from one or several points, e.g. from the sun, a lamp or a fire.</td>
</tr>
<tr>
<td>Noise</td>
<td>Sound</td>
<td>Environment</td>
<td>Any type of sound that comes from the surrounding environment.</td>
</tr>
<tr>
<td>Physiological data</td>
<td>Physiological context</td>
<td>User</td>
<td>Information associated to the biological signals of the user, e.g. heart rate, skin conductance, ECG, etc.</td>
</tr>
<tr>
<td>Preference</td>
<td>Need</td>
<td>User</td>
<td>Anything desired by the user of the system, e.g. a light colour for the UI.</td>
</tr>
<tr>
<td>Presentation</td>
<td>Format</td>
<td>Platform</td>
<td>The way the components or widgets are located and formatted in the UI.</td>
</tr>
<tr>
<td>Screen</td>
<td>Display</td>
<td>Platform</td>
<td>Surface of some devices, e.g. smartphones or computers, on which pictures and other type of content are shown.</td>
</tr>
<tr>
<td>Weather</td>
<td>Climate</td>
<td>Environment</td>
<td>Conditions in the air, e.g. rain, fog, wind, temperature or humidity.</td>
</tr>
</tbody>
</table>
Table 16 presents a glossary focused on describing the different software architecture concepts found for developing context-aware systems.

<table>
<thead>
<tr>
<th>Software architecture concept</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aspect-Oriented Architecture (AOA)</td>
<td>An aspect represents a specific concern (safety, coordination, distribution, etc.) that crosscuts the software architecture, i.e. those concerns that do not crosscut the architecture will not be an aspect [36].</td>
</tr>
<tr>
<td>Blackboard architecture</td>
<td>The information of the computation is centrally located and operated on by independent computations which interact only through the shared data [22].</td>
</tr>
<tr>
<td>Client-Server architecture</td>
<td>Particular case of distributed architecture in which a server is a process that provides services to other processes, i.e. the clients. Usually the server does not know in advance the clients that will access it at runtime, but clients know the identity of the server and can access it by remote procedure call (adapted from [22]).</td>
</tr>
<tr>
<td>Component-based architecture</td>
<td>The system is represented as a set of software components, their connections and their behavioural interactions [4].</td>
</tr>
<tr>
<td>Distributed architecture</td>
<td>Distributed systems have developed some common organizations for multi-process systems. They can be characterized by their topological features, such as ring and star organizations, or by the types of inter-process protocols that are used for communication (adapted from [22]).</td>
</tr>
<tr>
<td>Event-triggered architecture</td>
<td>A component broadcasts one or more events, while other components register an interest in an event by associating a procedure with such event. When the event is announced, the system invokes all procedures that have been registered for that event (adapted from [22]).</td>
</tr>
<tr>
<td>Micro-architecture</td>
<td>Micro-architectures are building blocks for designing applications. They represent a higher level of abstraction than individual patterns, and are expressed by a combination of patterns to solve a problem (adapted from [3]).</td>
</tr>
<tr>
<td>Middleware architecture</td>
<td>A middleware decouples the individual software components of the infrastructure from one another, presenting a uniform level of abstraction (adapted from [27]).</td>
</tr>
<tr>
<td>Model-Driven Architecture (MDA)</td>
<td>A conceptual framework that separates business decisions from platform decisions to provide flexibility when building and evolving the system (adapted from [7]).</td>
</tr>
<tr>
<td>Multi-Agent System (MAS)</td>
<td>For modelling real-world and social systems where problems are solved in a concurrent and cooperative way without the need of reaching optimal solutions. Its architecture is made of agents which are reactive, proactive and exhibit an intelligent and autonomous behaviour (adapted from [45]).</td>
</tr>
<tr>
<td>Multi-layered architecture</td>
<td>An architecture organized hierarchically where each layer provides service to the layer above it and serves as a client to the layer below. The connectors are defined by the protocols that determine how the layers interact (adapted from [22]).</td>
</tr>
<tr>
<td>Peer-to-Peer (P2P) architecture</td>
<td>This architecture provides a network characterized by self-organization, symmetric communication (all nodes are both servers and clients) and distributed control (no centralized server) that automatically rearranges itself to joining and leaving nodes (adapted from [40]).</td>
</tr>
<tr>
<td>Pipe-and-filter architecture</td>
<td>Each component (filter) has a set of inputs and outputs and reads streams of data on its inputs and produces streams of data on its outputs, delivering a complete instance of the result in a standard order. There are connectors that transmit outputs of one filter to inputs of another so they are called pipes (adapted from [22]).</td>
</tr>
</tbody>
</table>
Plug-and-Play (PnP) architecture
Architecture made of pluggable components so they can be reused or replaced with others easy and seamlessly.

Publish/Subscribe architecture
Information providers publish events to the system, and information consumers subscribe to events of interest within the system. This architecture ensures the timely notification of events to the interested subscribers (adapted from [58]).

Service-Oriented Architecture (SOA)
Architecture composed of collections of services, connected via well-specified contracts, which make calls on operations defined through their service interfaces (adapted from [7]).

B. 3 Research Topics and Other Terms
Some terms were found associated to the research topics of the selected papers and they are represented in Table 17.

<table>
<thead>
<tr>
<th>Term</th>
<th>Similar terms</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context-aware system</td>
<td>Situation-aware system</td>
<td>System that is able to adapt its behaviour based on the changes of the surrounding context.</td>
</tr>
<tr>
<td>Context delivery</td>
<td>Context dissemination</td>
<td>When an event occurs and matches any trigger previously specified, then some context information is provided to an application or service that requires it (adapted from [19]).</td>
</tr>
<tr>
<td>Context discovery</td>
<td>Context detection</td>
<td>Acquisition and inference over low-level context information to translate it into a more useful format for the application, i.e. high-level context information (adapted from [11]).</td>
</tr>
<tr>
<td>Context gathering</td>
<td>Context acquisition</td>
<td>First step of context discovery process in charge of collecting context data or subscribing to event notifications from different sources and sensors (low-level context information), usually demanded by services and applications (adapted from [41]). These sources have to model the context according to the format previously established and communicate with the system using the predefined set of interfaces and protocols (adapted from [48]).</td>
</tr>
<tr>
<td>Context inference</td>
<td>Context reasoning</td>
<td>Some context information cannot be obtained directly, but can only be derived to transform it into high-level context information. For example, the information whether or not a person is travelling and with which means of transport cannot be sensed directly and it has to be reasoned from other context information, such as Wi-Fi</td>
</tr>
</tbody>
</table>
### Context aggregation
- access points or calendar information. It represents the final step in the context discovery process (adapted from [50]).

### Context composition
- Resolving possible conflicts between contradictory sources of context information. It is in charge of processing the context information gathered by the context discovery component (adapted from [11]).

### Context handling
- Representing context information, by describing context attributes, their relationships and their values (adapted from [50]).

### Context processing
- A self-described, loosely coupled, modularized, self-contained and platform-independent software component that can be discovered and requested to implement a particular functionality (adapted from [12]).

### Service reconstruction
- Combination of some services together into a composed service to fulfil a complex business requirement (adapted from [12]).

### Service integration
- Provision of some functionality in the form of services, by answering simple requests or executing sophisticated processes. Services must be formally described to bring their descriptions in correspondence with existing ontologies (adapted from [53]).

### Service customization
- Resolving consumer requests in terms of advertised services which involves the selection of appropriate services and their prioritization. Service discovery can be done manually, automatically or by a combination (adapted from [53]).

### Service reconfiguration
- Organizing services by domain or location in order to be later discovered and consumed by applications (adapted from [41]).
Chapter 12

An Empirical Evaluation of Visualization Techniques for Architectural Knowledge

Citation: Cristina Roda, Elena Navarro, Carlos E. Cuesta, Dewayne Perry, Javier Jaén, “An Empirical Evaluation of Visualization Techniques for Architectural Knowledge”, Journal of Universal Computer Science, 2017 (Submitted)

Type of venue: International Journal (IF: 0.546, Q4)
Abstract: Recent research highlights the need to capture, represent and manage architectural design decisions and architectural design rationale as key elements of Architectural Knowledge. Despite the variety of techniques and tools that allow the visualization of this type of knowledge, there are no studies that evaluate the adequacy of the different proposed visualization techniques for exploiting Architectural Knowledge. We describe a set of visualization techniques that presents Architectural Knowledge in various ways. These visualization techniques are analyzed, considering their strengths and weaknesses, performing an empirical evaluation with a controlled experiment. The main goal of this experiment is to evaluate which is the most usable visualization technique when exploiting Architectural Knowledge in the process of performing architectural changes. The results of this experiment conclude that depending on the main task to be performed: navigating, overall understanding or updating Architectural Knowledge, different visualization techniques should be used. Consequently, tools that offer a combination of visualization strategies should be designed to provide software architects with specific task-oriented visualization mechanisms.

Keywords: Design Decision; Design Rationale; Architectural Knowledge; Visualization Technique; Ontology; Empirical Evaluation

Categories: M.3, M.4, M.7, M.8, L.0.0, L.1.0, D.2.11

1 Introduction

Software development has to deal with many challenges, such as system complexity, non-functional qualities, maintenance operations, distributed production, frequent personnel changes, etc. [Kruchten et al., 2009]. Furthermore, software companies with high maintenance costs are increasingly demanding flexible, easy-to-maintain designs [Kruchten et al., 2009].
Software Architecture (SA) is a valuable artifact that enables software companies to achieve a variety of goals by representing and communicating the system structure and behavior to all of a system's stakeholders [Perry and Wolf, 1992]. Until recently, the primary focus of SA research has been on Architecture Description Languages (ADLs) focusing on the description of SA elements and form (properties of, and relationships among elements, i.e. constraints on the elements) in the model of SA presented by [Perry and Wolf, 1992]. Unfortunately, no much attention has been paid to the third element of their model – namely, rationale - until [Bosch, 2004] recalled its importance. In the last years this has changed: the importance of Architectural Design Decisions (ADDs) and their Architectural Design Rationales (ADRs) has been recognized and become a significant research focus. ADDs and ADRs are essential aspects in Architectural Knowledge (AK), and its modeling, managing, and sharing has also become a significant research focus [Tang et al., 2009] [Matturro and Silva, 2010].

In this context, it should be highlighted that whenever a design decision is explicitly recorded and documented, new activities arise during the architecting process. The AK information appears as a view that overlaps the information provided by other views [Kruchten et al., 2009], helping in different stages of the software lifecycle, such as evolution [Cuesta et al., 2012]. Therefore, the introduction and application of adequate techniques for visualization become a necessity that should enable the different stakeholders to navigate throughout the different views of the system.

An interaction technique refers to a software and/or hardware interface for the user to interact with a visualization technique which is the visual encoding method [Huang, 2014]. A visualization technique in the context of AK is a specific way of visualizing ADDs and ADRs, as well as the relations between them, i.e. a particular form to support AK capturing in architecture documentation [Jansen et al., 2009].

Currently, there are many visualization techniques available to represent and capture AK. This is a very active research area that has produced a significant number of approaches for representing and capturing ADDs as it will be discussed in the next section. However, despite the variety of different visualization techniques that have emerged during the last years, very little work has been done to analyze their strengths and weaknesses in order to assess which visualization technique is the most suitable to represent and capture AK. This analysis is relevant because, in this manner both architects and designers of tools for AK management would have useful guidelines whenever they would have to make decisions about the best alternatives for AK visualization depending on their needs. Therefore, the purpose of this paper is to analyze the existing approaches to visualize AK and to provide insight into their strengths and weaknesses with respect to the activity to be carried out by software architects. Therefore, the purpose of this paper is to provide insight into the strengths and weaknesses of these approaches in visualizing AK.

This paper is structured as follows. We present related work in [Section 2] and discuss our classification of the visualization techniques used in this paper in [Section 3], along with the identification of the software tools that match with each of these techniques. Then, in [Section 4], we describe the empirical evaluation, and [Section 5] presents threats to the validity of the experiment. Finally, [Section 6] presents our conclusions and future work.
2 Related Work

AK has become an emerging issue of discussion and research within the SA research area. When software is developed in a multinational company or in an open source community, the stakeholders are often geographically distributed [Cataldo and Nambiar, 2012] [Babar, 2012]. Furthermore, SA has been identified as one of the primary mechanisms for organizing distributed software development [Grinter et al., 1999]. This is why AK has become increasingly important and, consequently, there is a significant need for appropriate tool support to store, reuse and share architectural knowledge.

One of the key issues that arises whenever AK is introduced in a software development process is the selection of the best supporting tool. This question is tackled by [Henttonen and Matinlassi, 2009], when they present an evaluation framework for AK tools that was used to evaluate three open source, language and platform independent solutions: WebOfPatterns [Dietrich and Elgar, 2005], Stylebase for Eclipse [Henttonen, 2007] and PAKME [Babar and Gorton, 2007]. Their evaluation framework for AK tools considers four points of view: Problem the tool assists in, Problem solver, Means of problem solving and Maturity of the tool. Each one of these categories is associated with a set of criteria, and each criterion is related to some evaluation questions to reveal the strengths and weaknesses of each tool. However, one of the main drawbacks of this evaluation framework is that it does not include any consideration for evaluating the visualization of information.

Other authors, such as [Tang et al., 2009], have also proposed a framework for comparing AK management tools. This framework comprises several criteria, each with an associated research question to represent the context for the comparison. Each question consists of a description, as well as a list of AK activities in the architectural lifecycle describing the usage contexts. For instance, Types and Representation of AK is a criterion to determine “What are the AK types and representations captured by a tool for general, reasoning and design knowledge?” This framework defines a solid guide to compare AK tools, including the majority of the features required by the architecture lifecycle: Types and representation of architecture knowledge, Relations between AK elements, Architectural analysis support, Architectural synthesis support, Architectural evaluation support, Architectural implementation support, Architectural maintenance support, AK customization, Integration with other tools and Collaborative environments. Unfortunately, the work of [Tang et al., 2009] does not include, once more, any criteria for evaluating AK visualization.

Some other works have empirically assessed the importance of AK documentation in software architecture, using case studies or controlled experiments. For instance, [Falessi et al., 2006] propose the Decision Goal Alternatives (DGA) technique to document design rationale. They have empirically evaluated, using a controlled experiment, the impact of such a technique on the effectiveness of individual and team decision-making, when the requirements change. The results of this study conclude that, for both kinds of decision-making, effectiveness improves when the design rationale documentation technique is available. [Falessi et al., 2008] also make a replicated experiment in order to confirm empirically the feasibility of a value-based approach for documenting design decisions, and the relevance of different Design Decisions Rationale Documentation (DDRDr) information categories. The results confirmed that the usefulness of such DDRDr categories depends on their
purpose. However, these works do not evaluate the usability of a particular approach but they assess its feasibility.

Other works have focused on empirically evaluating the usefulness of design rationale. For instance, [Bratthall et al., 2000] performed a controlled experiment to assess the value of having access to a retrospective design rationale. For a particular complex system, the results confirmed that design rationale is helpful for speeding up changes and improving correctness. Similarly, [Karsenty, 1996] has performed an empirical evaluation of design rationale documents and confirmed that, despite being useful for some designers to support their reasoning, they are not enough to support the full architecture lifecycle. Once again, although these evaluations assess the usefulness of design rationale, they do not assess the effectiveness of how it is represented, i.e. its visualization type.

In conclusion, the analyzed previous works can be classified into two categories, those that evaluate AK management tools in the context of software development processes [Henttonen and Matinlasi, 2009] [Tang et al., 2009] and those such as [Falessi et al., 2006] [Falessi et al., 2008] [Bratthall et al., 2000] [Karsenty, 1996] that evaluate the usefulness of documenting AK. However, none of them is focused on assessing the usability effectiveness of a given visualization technique. As far as we know, the work presented by [Shahin et al., 2014] is the only one that deals with visualization techniques for representing AK. Nevertheless, they focus on presenting which visualization techniques are currently supported by different tools in order to classify them, but they do not evaluate them to provide a practical guideline that reflects their usability in terms of AK manipulation, as we do in this work.

Hence, our goal in this paper is twofold. Firstly, to perform an empirical evaluation of several visualization techniques that capture, represent and maintain AK to obtain evidences that may suggest which visualization technique is the most suitable one in terms of its usability for AK manipulation and, secondly, to provide a useful framework for analyzing tools that support architectural knowledge visualization, to determine their strengths and weaknesses with respect to AK visualization techniques.

3 Visualization Techniques for Knowledge Systems

Several visualization techniques can be used to capture, represent or maintain AK. We want to emphasize that one of the assumptions of our research is that the architectural knowledge is, per se, available in the form of a knowledge base made up of ADDs and ADRs and their corresponding relationships that can be used to understand and reason about the software architecture. However, this base of knowledge needs to be formally defined so that the AK network of interconnected information can be easily explored and navigated. In this sense, AK resembles an ontology [Gruber, 1993], because, in general, knowledge of an Information System can be managed by means of ontologies [Rios-Alvarado et al., 2013]. Other authors have already noticed this relation, such as [Akerman and Tyree, 2006], [de Boer et al., 2009], [de Graaf et al., 2012] or [López et al., 2012]. Therefore, this has led us to use a taxonomy of visualization techniques proposed by [Katifori et al., 2007] which distinguishes different representation types, depending on some factors: 1) presentation of information (how data is represented), 2) interaction style (how the
user interacts with the information in order to manage it, for this aim we have taken into account the interaction styles presented in [Interaction Design Foundation, n.d.], or 3) *functionality supported* (what predominant functionality features are provided). In the following subsections, each visualization technique (Indented list, Wiki, Node-link and tree, Zoomable and Space-filling) is described, along with some AK tools that can be classified under each technique.

Notice that for each visualization technique, a representative tool has been selected in order to perform the empirical evaluation. Hence, we selected five tools which are described in more detail, according to the three factors: presentation of information, interaction style and functionality supported proposed by [Katifori et al., 2007].

It is worth noting that methods or tools grouped in these five types of visualization techniques can be also classified as 2D or 3D, depending on the space dimensions they employ, as [Katifori et al., 2007] stated. In this work, we focus on 2D methods, letting 3D ones for future work.

### 3.1 Indented list

The *Indented list* visualization technique represents AK by means of plain text that looks like a tree view, similar to Windows Explorer. Such representation makes this technique not very popular today to visualize architectural knowledge, as it is not amenable to analysis. An AK system that uses this visualization technique is PHI [McCall, 1991] and some tools that support the PHI methodology are: *JANUS* [Regli et al., 2000], *HOS* [Regli et al., 2000], and *PHIDIAS* [Regli et al., 2000]. However, these tools do not currently have any support and, instead, some generic ontology management ones based on the *Indented list* visualization technique are used in the AK context: *KAON* [KAON, 2011], *OntoRama* [Eklund et al., 2002], *OntoEdit* [Sure et al., 2002] and *Protégé* [Noy et al., 2000].

For this technique of visualization, we have selected *Protégé* [Figure 1] as the tool for our empirical evaluation because it is open-source and it has already been considered in the context of AK management [Jansen et al., 2009] [Katifori et al., 2007]. It is an ontology-editing and knowledge-acquisition environment, where classes are represented as nodes in an indented, retractable and expandable tree whereas instances are displayed in a separate visualization panel.
Figure 1: Protégé 3.4.4

[Table 1] characterizes Protégé in terms of the three factors mentioned before: 1) Presentation of information, 2) Interaction style and 3) Functionality supported.

<table>
<thead>
<tr>
<th>Presentation of information</th>
<th>Interaction style</th>
<th>Functionality supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Windows Explorer-like tree view</td>
<td>Form fill-in</td>
<td>Load and save OWL and RDF ontologies</td>
</tr>
<tr>
<td></td>
<td>Menu selection</td>
<td>Edit and visualize classes, properties and rules</td>
</tr>
<tr>
<td></td>
<td>Direct manipulation</td>
<td>Define logical class characteristics as OWL expressions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Execute reasoners such as description logic classifiers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Edit OWL individuals for Semantic Web markup</td>
</tr>
</tbody>
</table>

Table 1: Characterization of Protégé (Indented list visualization technique)

3.2 Wiki

The Wiki visualization technique can be used for capturing architectural knowledge [Farenhorst et al., 2007] in order to allow designers and architects to collaborate and communicate easily. Thanks to its capabilities, the information can be quickly updated, and stakeholders can always know the current state of the project. Some wiki
tools for architectural knowledge visualization are: C-ReCS [Klein, 1997], PAKME [Babar et al., 2006] and ADDSS [Capilla et al., 2006].

In this case, ADDSS [Figure 2] has been selected to represent AK using the Wiki visualization technique for our empirical evaluation. ADDSS is a web-based tool, as PAKME, that manages and documents ADDs, enabling users the sharing of AK through groupware support. It is chosen over other wiki tools because it is the most complete one, as it provides a query system that allows architects to easily find information about stored requirements, decisions and architectures.

![Figure 2: ADDSS 2.0](image)

[Table 2] characterizes ADDSS in terms of the three factors mentioned before.

<table>
<thead>
<tr>
<th>Presentation of information</th>
<th>Interaction style</th>
<th>Functionality supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>A table view</td>
<td>Form fill-in</td>
<td>Multi-perspective and groupware support</td>
</tr>
<tr>
<td></td>
<td>Menu selection</td>
<td>Visual representation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Complexity control</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Multiple projects and architectures</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Different categories of user and permissions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Architecture iterations</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Design decisions support and dependencies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Patterns and styles</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Architecture visualization and documentation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Support for (non-)functional requirements</td>
</tr>
</tbody>
</table>

*Table 2: Characterization of ADDSS (Wiki visualization technique)*
3.3 Node-link and tree

A *Node-link and tree* approach provides an interconnected node representation, with a top-down or left-right layout. This technique allows users to expand and retract nodes and their sub-nodes, so that the information detail level can be tuned. Some tools that support this kind of visualization technique are: *ARCHIUM* [Jansen and Bosch, 2005], *AREL* [Tang et al., 2007], *DRIMER* [Peña-Mora and Vadhavkar, 1997], *DRL* [Lee, 1990], *IBIS* [Kunz and Rittel, 1970], *gIBIS* [Conklin and Begeman, 1988] (IBIS successor), *Kruchten’s ADD Ontology tool* [Lee and Kruchten, 2008], *ODV* [de Boer et al., 2009], *QOC* [MacLean et al., 1991], *SCRAM* [Sutcliffe and Ryan, 1998], *SEURAT* [Burge and Brown, 2004], *Sysiphus* [Bruegge et al., 2006] and *Compendium* [Compendium, 2013].

![Figure 3: Compendium 1.5.2](image)

In this case, *Compendium* [Figure 3] has been selected for our empirical evaluation. This tool is open source and it is implemented on top of IBIS and supports gIBIS notations. It is preferred over the other ones because it provides explicit support
for rationale visualization of ADDs, presenting this AK as a directed graph, allowing users to navigate and explore the decision network. [Table 3] characterizes Compendium in terms of the three factors mentioned before.

<table>
<thead>
<tr>
<th>Presentation of information</th>
<th>Interaction style</th>
<th>Functionality supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>A directed graph view</td>
<td>Menu selection</td>
<td>Drag and drop documents and websites onto a map</td>
</tr>
<tr>
<td></td>
<td>Direct manipulation</td>
<td>Complete freedom to arrange icons</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Keyword tagging and maps publication</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Map and label connections between concepts to illustrate links</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Create “Dialogue Maps” to display links to ideas in group projects</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Create “Argument Maps” to work collaboratively</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Organize large amounts of information</td>
</tr>
</tbody>
</table>

*Table 3: Characterization of Compendium (Node-link and tree visualization technique)*

### 3.4 Zoomable

The Zoomable visualization technique presents data as hierarchies. In this way, it displays nodes in the lower levels of the hierarchy, nested inside their parents, with a smaller size. Thus, if additional information is needed about a specific node, the nested hierarchy must be traversed by zooming-in to the child nodes in order to expand them and make them the current viewing level [Katifori et al., 2007]. In this category we have considered some ontology visualization systems even though they are not specifically designed as AK visualization tools given the lack of specific AK management tools supporting this visualization technique: Grokker [Rivadeneira and Bederson, 2003], CropCircles [Parsia et al., 2005] [Wang and Parsia, 2006] and Jambalaya [Storey et al., 2001].

In this case, Jambalaya [Figure 4] is the selected tool to represent AK using the Zoomable visualization technique. This tool is an integration of SHriMP [Wu and Storey, 2000] with Protégé [Section 3.1], where SHriMP is a multi-perspective software visualization environment, which combines single view and multi-view techniques to support software exploration. It has been selected because it has more software support than the other tools mentioned, and it works within Protégé, which was also previously selected as it has been considered in previous works for AK management. [Table 4] characterizes Jambalaya in terms of the three factors mentioned before.
Table 4: Characterization of Jambalaya (Zoomable visualization technique)

3.5 Space-filling

The *Space-filling* category presents nodes in a hierarchical form, using all the screen space, i.e. adjusts nodes to the available screen space. It is not considered a very interesting technique for two reasons: its lack of clarity and its complicated connection to software architecture. Given that there are no AK tools for this specific visualization technique, we have considered instead some ontological tools such as *TreeMaps* [Shneiderman, 1992], *Information Slices* [Andrews and Heidegger, 1998] or *SequoiaView* [Bruls et al., 2002] to evaluate the potential use of this visualization technique in the context of AK management.
In this case, *Information Slices* is discarded because it does not have enough software support. Alternatively, *SequoiaView* [Figure 5] is the tool selected to represent AK using the Space-filling visualization technique because it has more support than *TreeMaps*, is freeware, and provides a more efficient file searching and filtering mechanism. [Table 5] characterizes *SequoiaView* in terms of the three factors mentioned before.

<table>
<thead>
<tr>
<th>Presentation of information</th>
<th>Interaction style</th>
<th>Functionality supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical cushions view</td>
<td>Menu selection</td>
<td>Set the height of the ridges</td>
</tr>
<tr>
<td></td>
<td>Direct manipulation</td>
<td>Display global information (high level directories) or detailed (individual files)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Choose between original cushions or the squarified ones</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Select, inspect and open files/directories</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Define color schemes to facilitate identification of file types or load a ready-made color scheme from disk</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Filter files on name, size, as well as creation, modification or last access date</td>
</tr>
</tbody>
</table>

*Table 5: Characterization of SequoiaView (Space-filling visualization technique)*
4 Empirical Evaluation

We present a controlled experiment, following the structure to perform and define an experiment proposed by [Wohlin et al., 2000]. Furthermore, we also have heeded the advices presented by [Kitchenham et al., 2002].

4.1 Experimental context

The main goal of this study is to evaluate which visualization technique makes the most of the architectural knowledge in the process of performing a set of architectural changes (i.e. in making and remaking architectural decisions) to an existing SA.

Thus, following the GQM template [Basili et al., 1994], the goal of the study is redefined, determining its five parameters:

- **Object of study.** The objects of study are the (already mentioned) visualization techniques, and their usability, in the process of performing a set of architectural changes on the EFT system architecture by using Architectural Knowledge.
- **Purpose.** The purpose of our experiment is to evaluate these visualization techniques, as presented in [Section 3].
- **Perspective.** The perspective is from the point of view of the participants, i.e. these subjects would like to know whether there is any difference between the five 2D visualization techniques when they represent and capture architectural knowledge.
- **Quality focus.** The main effect evaluated in this experiment is the usability of the visualization techniques. There are multiple definitions of usability such as the one presented by [Krug, 2000] where it is informally defined as just making sure that something works well, i.e. that a person of average ability and experience can use a given element for its intended purpose without getting hopelessly frustrated. Other more formal definitions are those given by the ISO 9126-1 [ISO/IEC, 2000] that describes usability as a product as the capability of the software product to be understood, learned, used and attractive to the user, when used under specified conditions; or by the ISO 9241-11 [ISO, 1998] that defines usability as a process as the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.
- **Context.** The experiment is run within the context of the People’s Bank of China Guangzhou branch (PBC-GZ) architectural specification. Specifically, this experiment is conducted within an evolution activity of the EFT system architecture. In this context, some architectural changes have to be performed by a study group that will act as a set of software architects. The experimental context characterization is “blocked subject-object study” [Wohlin et al., 2000], because there is more than one subject per object, and more than one object (five 2D visualization techniques).

Given that we have no a priori preference for any of the presented visualization techniques in terms of usability, our null hypothesis will be formulated stating that they would all be equally effective.
4.2 Participants

The study group was composed of 15 subjects (three women and twelve men) whose ages were between 22 and 25. Since the original software architects of the EFT system were not available we involved 4th year undergraduate students in our study. The use of students as subjects of empirical studies is generally considered as suitable provided the tasks to be performed do not require high levels of industrial experience [Basili et al., 1999] [Höst et al., 2000]. The students were selected from an advanced HCI and Software Engineering course at the University of Castilla-La Mancha (Albacete campus, Spain). This is because experienced subjects are needed, according to the guidelines presented by [Wohlin et al., 2000], not with substantial industrial or work experience, but with enough experience in software engineering in order to understand the AK of the EFT system and the tasks to be performed, as well as to be able to complete such tasks in an appropriate manner. Additionally, it was mandatory that the subjects were used to terms such as usability and visualization techniques.

4.3 Experimental Material

Our empirical evaluation uses SA and AK from an example presented by [Tang et al., 2007] and [Tang, 2007] related to the financial control system. The People’s Bank of China Guangzhou branch (PBC-GZ) is a central bank branch which is responsible for the financial control and inter-bank payments and liquidations of the financial center Guangzhou and its surroundings. One of its systems is the Electronic Fund Transfer (EFT) that transfers and liquidates high value payments between all the specialized and commercial banks in the surroundings. This system has to serve over ten million people in southern China, and works as a gateway to connect all local banks to the national payment network.

The design, development and testing of the EFT system took about two years, employing over thirty designers and developers. It is interesting to know that the main architect of this system was Antony Tang. Its design was highly demanding as it was necessary that this system had to be trustworthy, efficient and secure, because it is the main core of the financial system in the region [Tang, 2007]. Because of confidentiality and security reasons, as [Tang, 2007] mentions, detailed descriptions of the internal mechanisms of how each component is implemented are not revealed, such as the lines of code of the implementation.

The main problem this system presented was that its design was difficult to understand for anyone outside the original development team, despite the fact that its design was thoroughly specified. For this reason, it was decided to capture the architectural knowledge, i.e. the ADDs and ADRs, so that anyone could interpret and understand the EFT system design when analyzing the system to make changes.

The architectural knowledge of the EFT system is used in the following section by a set of selected subjects to carry out the evaluation of the different visualization alternatives. Namely, we used the decision network for fault-resilient support, divided into two hierarchical structures, one from the point of view of Requirements [Figure 6], and the other from the point of view of Architecture Elements [Figure 7], where $R$ represents Requirement, $AR$ represents Architecture Rationale, $+$ represents Pros, $-$ represents Cons and $AE$ represents Architecture Element.
Figure 6: Decision network that supports fault-resilient architecture (from the point of view of Requirements)
Figure 7: Decision network that supports fault-resilient architecture (from the point of view of Architecture Elements)
4.4 Experimental design

In this study, we used the visualization technique used to represent the AK of the EFT system architecture, as the independent variable (i.e., as input). Furthermore, we used the usability of the different visualization techniques in response to these treatments as the dependent variable (i.e., as output determined by several treatments – that is, the architecture changes). The USE Questionnaire [Lund, 2001] (see Appendix) has been used to determine the usability of the various visualization techniques (Indented list, Wiki, Node-link and tree, Zoomable or Space-filling) in representing the AK of the EFT system. For each one of the visualization techniques, a questionnaire was created using the Moodle open-source e-learning platform [UCLM, 2014].

It is interesting to note that we previously performed a comparative study in order to determine which usability questionnaire between seven candidates (System Usability Scale or SUS, Questionnaire for User Interface Satisfaction or QUIS, Computer System Usability Questionnaire or CSUQ, Software Usability Measurement Inventory or SUMI, Website Analysis and MeasureMent Inventory or WAMMI, Usefulness, Satisfaction, and Ease of use or USE, and Perceived Usefulness and Ease of Use or PUEU) was the best with respect to several balanced criteria: number of questions (18%), number of measurable factors (30%), free-of-charge (14%), required users (20%) and number of scoring levels (18%). Our preferences here were to minimize the number of questions (100 points to the questionnaire with fewer questions), to maximize the number of measurable factors (100 points to the questionnaire with higher measurable factors), a free questionnaire (100 points if the questionnaire is free, 0 otherwise), to minimize the required users (100 points to the questionnaire with fewer required users), and a balanced number of scoring levels (100 points if the questionnaire has 5 levels of scoring).

<table>
<thead>
<tr>
<th>Quest.</th>
<th>Questions</th>
<th>Measurable factors</th>
<th>Free of charge</th>
<th>Users</th>
<th>Scoring levels</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUS</td>
<td>10 (100p)</td>
<td>1 (20p)</td>
<td>SI (100p)</td>
<td>12 (100p)</td>
<td>5 (100p)</td>
<td>76</td>
</tr>
<tr>
<td>QUIS</td>
<td>27 (37p)</td>
<td>5 (100p)</td>
<td>SI (100p)</td>
<td>19 (63p)</td>
<td>10 (50p)</td>
<td>72.26</td>
</tr>
<tr>
<td>CSUQ</td>
<td>19 (53p)</td>
<td>1 (20p)</td>
<td>SI (100p)</td>
<td>90 (13p)</td>
<td>7 (71p)</td>
<td>44.92</td>
</tr>
<tr>
<td>SUMI</td>
<td>50 (20p)</td>
<td>1 (20p)</td>
<td>SI (100p)</td>
<td>12 (100p)</td>
<td>3 (60p)</td>
<td>54.40</td>
</tr>
<tr>
<td>WAMMI</td>
<td>20 (50p)</td>
<td>1 (20p)</td>
<td>NO (0p)</td>
<td>30 (40p)</td>
<td>5 (100p)</td>
<td>41</td>
</tr>
<tr>
<td>USE</td>
<td>30 (33p)</td>
<td>4 (80p)</td>
<td>SI (100p)</td>
<td>12 (100p)</td>
<td>7 (71p)</td>
<td>76.72</td>
</tr>
<tr>
<td>PUEU</td>
<td>12 (83p)</td>
<td>2 (40p)</td>
<td>SI (100p)</td>
<td>12 (100p)</td>
<td>7 (71p)</td>
<td>73.72</td>
</tr>
</tbody>
</table>

Table 6: Assessment of several usability questionnaires
We have balanced each criterion according to our own preferences so, for example, we have considered the number of measurable factors in a usability questionnaire as the most important feature given that we will be able to gather richer data relating to more usability aspects. Based on this weighted model, as we can see in [Table 6], the usability questionnaire with the best rating is the USE questionnaire which was selected for our usability evaluation.

USE questionnaires evaluate four measurable factors:
- **Usefulness** [Nielsen, 2003] or utility which is referred to the design’s functionality, i.e. does it do what users need?
- **Ease of use** [Privitera, 2008] can be defined as the ability of a user to readily and successfully perform a task with a product without the need for an advanced explanation and/or the instruction manual, i.e. Is a product easy to use?
- **Ease of learning** [Quesenbery, 2003] a product that is easy to learn allows users to build on their prior knowledge without deliberate effort, i.e. how well does the product support both initial orientation and deepening understanding of its capabilities?
- **Satisfaction** [Nielsen, 2003] how pleasant is it to use the design?

Given that we have no a priori preference for any of the presented visualization techniques in terms of usability, our null hypothesis will be formulated stating that they would all be equally effective. In this manner, our formal hypothesis formulation is as follows:
- **Null Hypothesis**, \( H_0 \): There is no difference in usability (measured as the average score provided by the USE Questionnaire [Lund, 2001]) between the five visualization techniques.
- **Alternative Hypothesis**, \( H_1 \): There is a difference in usability between, at least, a pair of visualization techniques.

[Table 7] presents a summary of the main features of the experiment.

<table>
<thead>
<tr>
<th>Null Hypothesis</th>
<th>( H_0 ): There is no difference in usability between the five visualization techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alternative Hypothesis</td>
<td>( H_1 ): There is a difference in usability between, at least, a pair of visualization techniques</td>
</tr>
<tr>
<td>Location</td>
<td>University of Castilla-La Mancha (Albacete campus, Spain)</td>
</tr>
<tr>
<td>Date</td>
<td>March 2011</td>
</tr>
<tr>
<td>Subjects</td>
<td>15 students of the 4th year in the Bachelor in Computer Science degree (5 groups of 3 students each)</td>
</tr>
<tr>
<td>Dependent Variable</td>
<td>The usability of the various visualization techniques in response to the various treatments</td>
</tr>
<tr>
<td>Independent Variable</td>
<td>The AK of the EFT systemarchitecture as represented by the various visualization techniques</td>
</tr>
</tbody>
</table>

**Table 7: Main features of the experiment**
4.5 Tasks

The experiment consisted of (1) subjects performing individually several tasks (each task is considered to be a treatment applied to the EFT architecture) which are usually done by architects while evolving a system, and (2) evaluating the usability of the different techniques using the USE Questionnaire [Lund, 2001]. These two tasks were related to the modification of the architectural knowledge (or the software architecture itself) from the EFT project described in [Section 4.3]. The modifications to be performed were related to the following aspects:

- The selection of the system and software architecture platform is one of the most fundamental architecture issues. The architect must take into account that the EFT system must provide fault-resilient support. So, he/she has to wonder about what is the best system that provides continuous processing with little chance of failing. There are two possible choices for a reliable machine: a fault-resilient system having always a node standing by to take over if another system node fails; or a fault-tolerant system having built-in backup processing modules. For the EFT system, the architects selected a fault-resilient system because it satisfied the reliability requirements of the central bank, whereas a fault-tolerant system had a high cost that made it an unattractive candidate. However, the fault-resilient system entails a disadvantage: other associated platform products are required to maintain the 99.95% uptime. So, the architect had to make another decision: what recovery strategies were necessary to deploy by using the platform environment.

- The architects also had to pay attention to network reliability, because banking operations must be carried out in a secure environment. For this issue two options were considered: either to add a frame-relay link to the system; or to include another frame-relay line as a backup. The architects selected the first option because it made possible to dial-in from member banks through the Public Switched Telephone Network (PSTN), while the second option was uneconomical and more risky, considering that the backup frame-relay line might fail as well.

- Another important aspect was power failure, given that the EFT system had to provide a continuous service. The architect had to answer the question about what is the best way to provide secondary power supply. Two alternatives were evaluated: an uninterrupted power supply (UPS); or a power generator. Given that the second alternative would require a higher budget that could not be justified the architects selected the UPS option.

- To handle natural disasters such as earthquakes or fires which could damage the entire processing center, the architects had to select a suitable mechanism: a remote site which could take over processing; or manual procedures. Finally, the architects selected manual procedures, because the budget did not allow them to choose the first option.

From this system design, we created two different structures related to the AK of the EFT system: one from the point of view of the requirements [Figure 6], and the other one from the point of view of the architectural elements [Figure 7]. Once the subjects were provided with these two structures of the AK of the system, as
represented using the different visualization techniques, the following tasks (treatments) were to be applied in the context of the experiment:

- **Task 1** – The subjects were informed that the PBC-GZ has not had any monetary problems, so that the EFT system had to evolve to deploy the best alternatives, no matter their costs. For this reason, the first change to be performed was to modify a high-level requirement: choose the *Best Solution* instead of the *Cost Effective Solution*.

- **Task 2** – A new requirement (*24h Monitoring*) was added: it allows the system to be aware of any warning due to power supply problems, machine failure, communication failure, and site failure. The introduction of this new requirement affected most of the initial architecture rationale defined for the system.

- **Task 3** – The last change is that the *ORACLE database* is to be replaced by a *MySQL database* because PBC-GZ is shifting to open source software.

Initially, all subjects had to carry out the three tasks proposed. These three tasks led participants to navigate through the two structures of architectural knowledge, and to modify what they considered adequate, taking into account that the first two changes affected the requirement-centric structure, and the last one impacted the architectural element-centric structure. Furthermore, a guideline was prepared to provide assistance on the operation of each tool (available in the technical report presented in [Roda et al., 2014]).

### 4.6 Procedure

In this experiment, subjects must make changes to the EFT software architecture (changes in the initial requirements and architectural elements, such as the type of the database). They have to carry out those changes using each previously selected visualization tool: Protégé, ADDSS, Compendium, Jambalaya and SequoiaView, each belonging to a different visualization technique: Indented list, Wiki, Node-link and tree, Zoomable and Space-filling, respectively. After the set of changes has been successfully completed within a visualization tool, each subject evaluates the usability of this tool answering a usability questionnaire [Lund, 2001].

We organized the work of subjects in five different sub-groups (each one composed of 3 subjects working independently) so that each group started and finished with a different visualization technique, in order to alleviate the learning effect [Figure 8]. Also, the order of treatment for each application has been controlled to be different for the five groups. Furthermore, to reduce the fatigue effect, two points can be stated here: (i) as the tasks were not very complex, they could be solved soon, reducing the fatigue; and (ii) the use of different visualization techniques and the introduction of several tasks helped to motivate the subjects and alleviate the fatigue effect too.
Our experiment was carried out in three different phases:

- First, we introduced the study to all the subjects together, describing its main purpose, the experimental material to be used (the EFT system), and all the tasks to be performed, in addition to the explanation of some basic concepts such as Design Decision or Design Rationale. We prepared a Microsoft Power Point presentation in order to present this information (available in the technical report presented in [Roda et al., 2014]).

- Second, we gave a brief introduction of how to use the selected tools (Protégé, ADDSS, Compendium, Jambalaya and SequoiaView) for each visualization technique (Indented list, Wiki, Node-link and tree, Zoomable and Space-filling), so that the subjects acquired necessary abilities for their manipulation. For this we created a quick guideline (available in the technical report presented in [Roda et al., 2014]) that describes the main steps to use each tool and presents the two architectural structures of the EFT system, one from the point of view of the Requirements and the other from the point of view of the Architecture Elements. Furthermore, we prepared a zipped file that contains some files created in order to use these two structures within each visualization tool, so the subjects start from these structures already created for each tool thus they may only concentrate on the proposed tasks, not on creating the structures. It is worth noting that the subjects were taught how to perform exactly the same tasks (Task 1, Task 2 and Task 3 from the previous section) with each tool, so that they did not focus on the tool facilities but on the visualization technique, that is the aim of the experiment.

- Third, subjects carried out each one of the described tasks (i.e., treatments) Task 1, Task 2 and Task 3 from the previous section, using the five tools and, as soon as they finished them, they were required to fill in the usability questionnaire for each visualization technique. These questionnaires (five
because there is one per visualization technique) were created in the Moodle platform [UCLM, 2014] of the University of Castilla-La Mancha, so that every subject could complete them individually in a simple way and we could gather the information easy and efficiently.

In addition to the usability questionnaire, a zipped file with the projects of each visualization tool, the quick guideline to these tools and the Power Point presentation were left online so that subjects were able to access them at any time.

It is worth noting that at each phase, subjects were advised that the main aim of the experiment was to evaluate the visualization technique and not the tool, given that tools were selected to represent AK, but offered it in a different way, depending on the visualization technique. Furthermore, the subjects were also informed that the experiment would be anonymous.

The summary of the scheduling of the experiment is shown in [Figure 9]. Notice that time distribution of the different phases is illustrative as it can change depending on real circumstances, taking into account that the total time is immovable (2 hours).

![Figure 9: Planning of the experiment (total time: 2h)](image)

### 4.7 Results analysis

As already noted, our empirical evaluation was designed as an experiment, which presents results for usability, based on the USE Questionnaire [Lund, 2001].

To provide an initial global impression about the results, [Figure 10] shows the score that each Visualization Technique obtained which was determined by using the results of the USE questionnaires filled by the subjects and calculating the average for each technique. This figure shows the preference for *Node-link and tree*, followed by *Indented list, Zoomable, Space-filling* and *Wiki*, in this order.
Initially, we considered applying the ANalysis Of VAriance (ANOVA) technique [Wilcoxon, 1995], which generalizes a t-test to more than two groups, and provides us with statistical results that allows us to decide whether or not the averages of several groups are comparable. In order to test this hypothesis, the probability distribution of the answers has to accomplish the following assumptions [Wilcoxon, 1995]:

1. **Independence of cases**, to simplify the statistical analysis.
2. **Normality**, i.e. observations are made on normal populations.
3. **Homoscedasticity** or homogeneity of variances, i.e. the variance of data in all the groups should be the same.

Unfortunately, this is not the case with our experiment. Among many other details, our sample size is relatively small, and subjects were not randomly selected; moreover, observations cannot be considered independent, as the same population was used to evaluate all five techniques (although in different order), i.e. the same subjects gave a score for every technique, at different times. Therefore, this excludes the possibility of applying the ANOVA test. Of course, this does not mean that we have to completely discard a statistical analysis; more appropriately, we should use non-parametric alternatives to ANOVA, which take into account these specific issues.

Hence, we must select a statistical hypothesis test which differs from other usual approaches in several aspects. First, there should be no assumption about the distribution of the population. Second, there should be no assumption about independent observations as the same subjects used the five visualization techniques. Consequently, the Friedman test [Lehmann, 1985] was chosen to perform this analysis in a formal way. This test is a non-parametric statistical method used to detect differences in treatments across multiple test attempts. The procedure involves ranking each row together and considering the values of ranks by columns.

We used the SPSS program to perform the Friedman test. [Table 8] presents the results of this execution. As can be seen, the Friedman test established that the differences were statistically significant in terms of usability, depending on which visualization technique was used whilst modifying the architecture. Specifically, the result for a 95% confidence level is that \( \chi^2(2) = 9.914 \), \( p = 0.042 \). This indicates that at least two visualization techniques have patent differences with regard to usability.
Table 8: Friedman test (95% family-wise confidence level)

To learn where the differences actually happen, the Wilcoxon signed-rank tests must be run on the different combinations of related groups. Therefore, we applied this test using the following combinations:

- Indented list - Node-link and tree
- Indented list - Space-filling
- Indented list - Wiki
- Indented list - Zoomable
- Node-link and tree - Space-filling
- Node-link and tree - Wiki
- Node-link and tree - Zoomable
- Space-filling - Wiki
- Space-filling - Zoomable
- Wiki - Zoomable

A Bonferroni adjustment on the results provided by the Wilcoxon tests must be also carried out because, as we are making multiple comparisons, it is more likely to declare a result significant when we should not (i.e. a Type I error). The Bonferroni adjustment sets that the significance level initially used (in this case, 0.05, as the confidence level is 95%) must be divided by the number of tests to be run. Therefore, in this study, we must use a significance level of 0.05/10 = 0.005, so the confidence level is 99.5%. This means that if the p-value is larger than 0.005, we do not have a statistically significant result.

Therefore, post-hoc analysis with the Wilcoxon signed-rank tests was conducted applying a Bonferroni correction, resulting in a significance level set at p < 0.005. Median (IQR) perceived usability for the five techniques was 4.34 (3.45 to 4.84) for Indented List, 4.96 (4.10 to 5.75) for Node Link and tree, 3.43 (2.02 to 4.90) for Space Filling, 2.67 (1.70 to 3.930) for Wiki, and 2.67 (1.70 to 3.930) for Zoomable. As shown in [Table 9], there were no significant differences between any of the visualization techniques, as all the p-values were higher than 0.005, except for the pair Wiki vs. Node link and tree. Specifically, there was only a statistically significant reduction in usability perceived in this pair (Z = -3.107, p = 0.002) what means that only in this case the Node link technique was actually considered more usable.
Table 9: Pairwise comparison by means of Wilcoxon signed rank test with a 99.5% family-wise confidence level (results (a) based on negative ranks, (b) based on positive ranks)

<table>
<thead>
<tr>
<th>Pairwise comparison</th>
<th>Z</th>
<th>Asymp. Sig. (2-tailed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node Link – Indented List</td>
<td>-2.197</td>
<td>0.028</td>
</tr>
<tr>
<td>Space Filling – Indented List</td>
<td>-1.083</td>
<td>0.279</td>
</tr>
<tr>
<td>Wiki – Indented List</td>
<td>-1.977</td>
<td>0.048</td>
</tr>
<tr>
<td>Zoomable – Indented List</td>
<td>-0.384</td>
<td>0.701</td>
</tr>
<tr>
<td>Space Filling – Node Link</td>
<td>-2.132</td>
<td>0.033</td>
</tr>
<tr>
<td>Wiki – Node Link</td>
<td>-3.107</td>
<td>0.002</td>
</tr>
<tr>
<td>Zoomable - Node Link</td>
<td>-1.601</td>
<td>0.109</td>
</tr>
<tr>
<td>Wiki - Space Filling</td>
<td>-1.099</td>
<td>0.272</td>
</tr>
<tr>
<td>Zoomable - Space Filling</td>
<td>-0.785</td>
<td>0.433</td>
</tr>
<tr>
<td>Zoomable - Wiki</td>
<td>-1.363</td>
<td>0.173</td>
</tr>
</tbody>
</table>

The USE Questionnaire [Lund, 2001] evaluates usability based on four factors (order does not matter): Usefulness, Ease of use, Ease of learning and Satisfaction. Therefore, to enrich our empirical evaluation, we decided to perform an additional analysis by taking into account these four factors, i.e. to use the Friedman test with a significance level of 0.05 (like in the previous case) for each one of them. This way, we could see whether any visualization technique had more influence with regard to a certain usability factor. [Figure 11] shows the summary results for each usability factor. As can be observed, Node-link and tree, Indented List and Zoomable present the highest scores for the usability factors.
In order to provide a formal support for our conclusions, we carried out again the Friedman test on the original data, for each usability factor. These results are presented in [Table 10]. As can be seen, we can find differences among at least two visualization techniques in each usability factor. Therefore, a Wilcoxon signed rank test was again run for each pair of usability factors.

<table>
<thead>
<tr>
<th>Null Hypothesis</th>
<th>Test</th>
<th>df</th>
<th>Chi-square</th>
<th>Sig</th>
</tr>
</thead>
<tbody>
<tr>
<td>All the visualization techniques offer the same usefulness</td>
<td>Related-Sample Friedman’s test</td>
<td>4</td>
<td>9.914</td>
<td>0.042</td>
</tr>
<tr>
<td>All the visualization techniques offer the same ease of use</td>
<td>Related-Sample Friedman’s test</td>
<td>4</td>
<td>10.110</td>
<td>0.039</td>
</tr>
<tr>
<td>All the visualization techniques offer the same ease of learning</td>
<td>Related-Sample Friedman’s test</td>
<td>4</td>
<td>9.868</td>
<td>0.043</td>
</tr>
<tr>
<td>All the visualization techniques offer the same satisfaction</td>
<td>Related-Sample Friedman’s test</td>
<td>4</td>
<td>12.498</td>
<td>0.014</td>
</tr>
</tbody>
</table>

Table 10: Friedman test (95% family-wise confidence level)

[Table 11], [Table 12], [Table 13] and [Table 14] show the results of the Wilcoxon signed rank tests run for the different pairwise comparisons. Most of the null hypotheses cannot be rejected as the p-values were greater than 0.005, except (again) for the pair Wiki vs. Node link and tree. As can be seen, there was a statistically significant reduction in usefulness, ease of use, ease of learning and satisfaction perceived in this pair (Z=-3.107, p=0.002; Z=-3.045, p=0.002; Z=-2.953, p=0.003; Z=-3.048, p=0.002, respectively). Moreover, there was also a statistically significant reduction, just in ease of use, in the space filling vs. node link and tree pair (Z = -2.920, p=0.004).
Therefore, in the light of the test results, we can state that there are no differences between all pairs of techniques, except for the Wiki – Node-link and tree pair. Our original intuition was that the Node-link and tree technique was the most effective in representing architectural knowledge, i.e. its graphical view provides more information than the other techniques and subjects are more used to this kind of visualization so they could manage AK in an easy way. However, our intuition was not right: Node-link and tree cannot be considered better than all the other approaches but only better than the Wiki technique. During the experiment, users’ feedback reflected that Node-link and tree presents architectural knowledge in a simple and clear way that allowed them to easily navigate and explore the EFT system decision network as the initial results shown in [Figure 10]. However, Node-link and tree also presents some deficiencies, which the statistical analysis has allowed us to detect, and that the other visualization techniques seem to alleviate. Once the users’ feedback was reviewed again, i.e. some qualitative data in form of subjects’ comments during the experiment, we observed that most of them highlighted that the Indented List technique is more appropriate to navigate throughout the architectural knowledge thanks to its hierarchical structure. Moreover, the Zoomable technique offers an interesting alternative to have a fast overview of the AK. However, Node-link and tree offers users a more intuitive way of carrying out the modification of tasks than the other alternatives. All in all, this leads us to an interesting conclusion: a combination of tools for exploiting AK should be designed to provide software architects with different visualization techniques by considering the main AK management task (modification, navigation or overall inspection) to be carried out by software architects.

<table>
<thead>
<tr>
<th>Usefulness</th>
<th>Pairwise comparison</th>
<th>Z</th>
<th>Asymp. Sig. (2-tailed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wiki - Node Link</td>
<td>-3.042 (b)</td>
<td>0.002</td>
<td></td>
</tr>
<tr>
<td>Wiki - Space Filling</td>
<td>-1.468 (b)</td>
<td>0.142</td>
<td></td>
</tr>
<tr>
<td>Wiki - Wiki</td>
<td>-1.153 (a)</td>
<td>0.249</td>
<td></td>
</tr>
<tr>
<td>Zoomable - Space Filling</td>
<td>-0.419 (a)</td>
<td>0.675</td>
<td></td>
</tr>
<tr>
<td>Zoomable - Node Link</td>
<td>-1.503 (b)</td>
<td>0.133</td>
<td></td>
</tr>
<tr>
<td>Zoomable - Indented List</td>
<td>-1.957 (b)</td>
<td>0.050</td>
<td></td>
</tr>
<tr>
<td>Zip - Node Link</td>
<td>0.000 (c)</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>Zoomable - Indented List</td>
<td>-0.353 (b)</td>
<td>0.724</td>
<td></td>
</tr>
<tr>
<td>Wiki - Indented List</td>
<td>-2.395 (a)</td>
<td>0.017</td>
<td></td>
</tr>
<tr>
<td>Space Filling - Indented List</td>
<td>-1.468 (b)</td>
<td>0.142</td>
<td></td>
</tr>
</tbody>
</table>

Table 11: Pairwise comparison by means of Wilcoxon signed rank test with a 99.5% family-wise confidence level (results (a) based on negative ranks, (b) based on positive ranks, and (c) the sum of negative ranks equals the sum of positive ranks) (I)
### Ease of use

<table>
<thead>
<tr>
<th>Pairwise comparison</th>
<th>Z</th>
<th>Asymp. Sig. (2-tailed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node Link – Indented List</td>
<td>-2.272  (a)</td>
<td>0.023</td>
</tr>
<tr>
<td>Space Filling – Indented List</td>
<td>-1.490  (b)</td>
<td>0.136</td>
</tr>
<tr>
<td>Wiki – Indented List</td>
<td>-1.789  (b)</td>
<td>0.074</td>
</tr>
<tr>
<td>Zoomable – Indented List</td>
<td>-0.282  (b)</td>
<td>0.778</td>
</tr>
<tr>
<td>Space Filling – Node Link</td>
<td>-2.920  (b)</td>
<td>0.004</td>
</tr>
<tr>
<td>Wiki - Node Link</td>
<td>-3.045  (b)</td>
<td>0.002</td>
</tr>
<tr>
<td>Zoomable - Node Link</td>
<td>-1.642  (b)</td>
<td>0.101</td>
</tr>
<tr>
<td>Wiki - Space Filling</td>
<td>-0.785  (b)</td>
<td>0.433</td>
</tr>
<tr>
<td>Zoomable - Space Filling</td>
<td>-1.412  (a)</td>
<td>0.158</td>
</tr>
<tr>
<td>Zoomable – Wiki</td>
<td>-1.533  (a)</td>
<td>0.125</td>
</tr>
</tbody>
</table>

Table 12: Pairwise comparison by means of Wilcoxon signed rank test with a 99.5% family-wise confidence level (results (a) based on negative ranks, (b) based on positive ranks, and (c) the sum of negative ranks equals the sum of positive ranks) (II)

### Ease of learning

<table>
<thead>
<tr>
<th>Pairwise comparison</th>
<th>Z</th>
<th>Asymp. Sig. (2-tailed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node Link – Indented List</td>
<td>-1.067  (a)</td>
<td>0.286</td>
</tr>
<tr>
<td>Space Filling – Indented List</td>
<td>-0.864  (b)</td>
<td>0.387</td>
</tr>
<tr>
<td>Wiki – Indented List</td>
<td>-1.792  (b)</td>
<td>0.073</td>
</tr>
<tr>
<td>Zoomable – Indented List</td>
<td>-1.248  (b)</td>
<td>0.212</td>
</tr>
<tr>
<td>Space Filling – Node Link</td>
<td>-1.557  (b)</td>
<td>0.120</td>
</tr>
<tr>
<td>Wiki - Node Link</td>
<td>-2.953  (b)</td>
<td>0.003</td>
</tr>
<tr>
<td>Zoomable - Node Link</td>
<td>-1.492  (b)</td>
<td>0.136</td>
</tr>
<tr>
<td>Wiki - Space Filling</td>
<td>-1.290  (b)</td>
<td>0.197</td>
</tr>
<tr>
<td>Zoomable - Space Filling</td>
<td>-0.236  (a)</td>
<td>0.813</td>
</tr>
<tr>
<td>Zoomable – Wiki</td>
<td>-0.966  (a)</td>
<td>0.334</td>
</tr>
</tbody>
</table>

Table 13: Pairwise comparison by means of Wilcoxon signed rank test with a 99.5% family-wise confidence level (results (a) based on negative ranks, (b) based on positive ranks, and (c) the sum of negative ranks equals the sum of positive ranks) (III)
### Satisfaction

<table>
<thead>
<tr>
<th>Pairwise comparison</th>
<th>Z</th>
<th>Asymp. Sig. (2-tailed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node Link – Indented List</td>
<td>-1.922 (a)</td>
<td>0.055</td>
</tr>
<tr>
<td>Space Filling – Indented List</td>
<td>-0.629 (b)</td>
<td>0.529</td>
</tr>
<tr>
<td>Wiki – Indented List</td>
<td>-2.355 (b)</td>
<td>0.019</td>
</tr>
<tr>
<td>Zoomable – Indented List</td>
<td>-0.126 (b)</td>
<td>0.900</td>
</tr>
<tr>
<td>Wiki - Node Link</td>
<td>-3.048 (b)</td>
<td><strong>0.002</strong></td>
</tr>
<tr>
<td>Zoomable - Node Link</td>
<td>-1.696 (b)</td>
<td>0.090</td>
</tr>
<tr>
<td>Wiki - Space Filling</td>
<td>-1.288 (b)</td>
<td>0.198</td>
</tr>
<tr>
<td>Zoomable - Space Filling</td>
<td>-0.863 (a)</td>
<td>0.388</td>
</tr>
<tr>
<td>Zoomable – Wiki</td>
<td>-1.335 (a)</td>
<td>0.182</td>
</tr>
</tbody>
</table>

*Table 14: Pairwise comparison by means of Wilcoxon signed rank test with a 99.5% family-wise confidence level (results (a) based on negative ranks, (b) based on positive ranks, and (c) the sum of negative ranks equals the sum of positive ranks)*

### 5 Threats to the validity of the experiment

In this section, we will discuss the issues that can threaten the validity of this experiment, considering the four types of threats proposed by [Wohlin et al., 2000]: *Conclusion validity*, *Internal validity*, *Construct validity* and *External validity*.

#### 5.1 Conclusion validity

We can affirm that our conclusions are statistically valid, given that we used the Friedman test (complemented with Wilcoxon tests, and applying a Bonferroni correction) to perform the analysis of our data. This is an appropriate test to carry out multiple comparisons within a subgroup, as in our case.

#### 5.2 Internal validity

*Internal validity* suffers somewhat from the use of students for the experiment rather than practicing software architects. Further studies would be needed with practicing architects to see whether their evaluations would match those of the students. The underlying reasons for the current usability evaluations suggest that these studies would have results congruent with our studies.

#### 5.3 Construct validity

*Construct validity* is strong. Usability and its constituent measures are well understood by the various subjects. Further, the use of a well described architecture together with its AK represented in well understood visualization techniques has strong construct validity as well.
5.4 External validity

The strength of external validity lies in the use of a realistic software architecture and its architecture knowledge. Its weakness is analogous to that of internal validity in that the experiment is performed using students but with our expectation of congruent results in further studies with practicing architects we consider external validity overall to be very good.

6 Conclusions and Further Work

As seen throughout this paper, design decisions and their rationales have to be well documented so that the system under development/maintenance can efficiently and easily evolve. However, sometimes this architectural knowledge is presented in an inadequate way that does not facilitate the architects’ task in evolving the system.

In this context, this paper describes five 2D visualization techniques to support architectural knowledge visualization and assesses them by means of an empirical evaluation of the quality factor usability. This experiment has allowed us to observe which visualization technique is the most effective for representing and manipulating architectural knowledge, in terms of four quality usability sub-factors: usefulness, ease of use, ease of learning and satisfaction. Node-link and tree technique has proven to be one of the most usable tools for this purpose, due to its simplicity and clarity in visualizing architectural knowledge as a comprehensible graph that is easy to be interpreted and navigated, using simple and understandable nodes. However, the experiment also shows that it is not the most usable technique, but it should be used in combination with others, such as Indented List and/or Zoomable. In this sense, the results of this experiment conclude that tools for exploiting Architectural Knowledge should be designed to provide software architects with different visualization techniques by considering the main AK management task (modification, navigation or overall inspection) to be carried out by software architects.

Finally, we want to emphasize some learned lessons extracted from this work: the Friedman test is a good option in making multiple comparisons between several subgroups, given that it takes into account the relation between them; to consider seasoned students as a study group is an acceptable choice for evaluating visualization techniques; the USE Questionnaire allows us to carry out a suitable evaluation of usability of each visualization technique, thanks to its simple questions and the seven levels of scoring those items; finally, the availability of several simple tasks allows subjects not to get bored (several) and ensure that they understand and perform them correctly (simple).

Our future work will focus on 3D visualization techniques for capturing architectural knowledge and will try to determine which technique is the most appropriate for each type of AK management activity. We are also interested in investigating what visualization techniques would facilitate a more effective collaborative work between software architects and, finally, we plan to confirm the present results with further experiments including practicing architects where possible.
Acknowledgments

We would like to thank Associate Professor Antonio Alonso-Ayuso, from the Statistics and Operation Research Department at Rey Juan Carlos University, and Associate Professors Mario Plaza and Francisco Parreño from the Statistics Department at University of Castilla-La Mancha their help with the statistical analysis; and Associate Professor Víctor López Jaquero, from University of Castilla-La Mancha, for his collaboration in the development of this study. This work has been partially supported by grants insPIre (TIN2012-34003), CoMobility (TIN2012-31104) and CreateWorlds (TIN2010-20488) from the Spanish Ministry of Economy and Competiveness, and also through the FPU scholarship (FPU12/04962) from the Spanish Government. Professor Perry is supported in part by NSF CISE grants IIS-0438967 and CCF-0820251.

References


of Software Architectures (QoSA 06), LNCS 4214., 2006. Springer.


### Appendix

#### USE questionnaire

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Question</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>N/A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usefulness</td>
<td>It helps me be more effective</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It helps me be more productive</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It is useful</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It gives me more control over the activities in my life</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It makes the things I want to accomplish easier to get done</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It saves me time when I use it</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It meets my needs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It does everything I would expect it to do</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ease of Use</td>
<td>It is easy to use</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It is simple to use</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It is user friendly</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It requires the fewest steps possible to accomplish what I want to do</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It is flexible</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Using it is effortless</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I can use it without written instructions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I don’t notice any inconsistencies when I use it</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Both occasional and regular users would like it</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I can recover from mistakes quickly and easily</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I can use it successfully every time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ease of Learning</td>
<td>I learned to use it quickly</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I easily remember how to use it</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It is easy to learn to use</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I quickly become skillful with it</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Satisfaction</td>
<td>I am satisfied with it</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I would recommend it to a friend</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It is fun to use</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It works the way I want it to work</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It is wonderful</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I feel I need to have it</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>It is pleasant to use</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Biography

Cristina Roda is a software engineer researcher at the LoUISE group of the Albacete Research Institute of Informatics and Ph.D. student in Computer Science at the University of Castilla-La Mancha (Spain), holding a scholarship from the National Government. Prior to this position, she worked as a software engineer at Vector Corp. She received her M.Sc. in Computer Science from the University of Castilla-La Mancha at the Superior Polytechnic School of Albacete, Spain, in 2011. Previously she finished her Bachelor Degree in Computer Science at the University of Castilla-La Mancha at the Superior Polytechnic School of Albacete in 2010. Her research interests are in Software Engineering, Model-Driven Development, Software Architecture and Architectural Knowledge.

Elena Navarro is an Associate Professor of Computer Science at the University of Castilla-La Mancha (Spain). Prior to this position, she worked as a researcher at the Informatics Laboratory of the Agricultural University of Athens (Greece) collaborating in the CHOROCHRONOS project funded by the Training and Mobility of Researchers program of the EU. Previously, she served as a staff member of the Regional Government of Murcia, at the Instituto Murciano de Investigación y Desarrollo Agrario y Alimentario, collaborating in the INTERREG II Project funded by the EU. During her master degree studies at the University of Murcia, she was a holder of several research scholarships funded by the Regional Government of Castilla-La Mancha and the National Government. She got her bachelor degree and Ph.D. at the University of Castilla-La Mancha, and her master degree at the University of Murcia (Spain). She is currently an active collaborator of the LoUISE group of the University of Castilla-La Mancha. Her current research interests are Requirements Engineering, Software Architecture, Model-Driven Development, and Architectural Knowledge.

Carlos E. Cuesta is an Associate Professor of Software Engineering at Rey Juan Carlos University, in Madrid (Spain), where he is also the current Academic Secretary (Deputy Dean) at the School of Computer Science and Engineering. Previously he has held positions as Teaching Assistant (1997-2002), Lecturer (2002-03), and Associate Professor (2003-06) at the Department of Computer Science in the University of Valladolid, also in Spain. Since he moved to Rey Juan Carlos University (2006), he has been the Director of the Master and Doctorate Program on Information Technologies and Computer Systems (2006-2010), and the Associate Dean of Academic Affairs and Head of Studies (2010-2012) of his School. He has been Program Co-Chair of the second Joint meeting of two major conferences in Software Architecture; the Tenth Working IEEE/IFIP Conference on Software Architecture and the Sixth European Conference on Software Architecture (WICSA/ECSA 2012). Previously, he was the Conference Chair for the first edition of the European Conference (ECSA 2007). He is also an active member of the ECSA Steering Committee (continuously since 2006); he also participates in many other Program Committees and editorial efforts. He is a founding member of the VorTIC3 research group at Rey Juan Carlos University, and has been previously linked to the ISSI group at Polytechnic University of Valencia (Spain). His main research interests are related to Software Architecture, and also include many bindings to Service Orientation,

**Dewayne E. Perry** is the Motorola Regents Chair of Software Engineering at The University of Texas at Austin (UT Austin) and the director of the Empirical Software Engineering Laboratory (ESEL). The first half of his computing career was spent as a professional programmer and a consulting software architecture and designer. The next 16 years were spent as a software engineering research MTS at Bell Laboratories in Murray Hill, New Jersey. He has been at UT Austin since 2000. His research interests include empirical studies in software engineering, software architecture, and software development processes. He is particularly interested in the process of transforming requirements into architectures and the creation of dynamic, self-managing, and reconfigurable architectures. He is a member of the ACM SIGSOFT and IEEE Computer Society, has been a coeditor-in-chief of Wiley’s Software Process: Improvement and Practice as well as an associate editor of IEEE Transactions on Software Engineering, and has served as organizing chair, program chair, and program committee member on various software engineering conferences.

**Javier Jaén**, PhD (2006, Universitat Politècnica de Valencia), MSc (1998, Virginia Tech), DEA (1994, INSA de Lyon) is currently an associate professor with the Laboratory of Advanced Information Systems at the Department of Computing and Information Systems. His current research interests include ubiquitous computing, ambient intelligence and tabletop-based computing. He was the recipient of a Fulbright scholarship and is a member of the Upsilon Pi Epsilon International Honor Society for Computing and Information Disciplines. Dr. Jaén was the winner of the Best PhD Dissertation in Computer Science Award and the Social Council's Award at the UPV, and won the eMobility Award in 2003.
Chapter 13

Variability as a Driver for the Adaptation of Tele-Rehabilitation Context-Aware Systems

Citation: Cristina Roda, Elena Navarro, Víctor López-Jaquero, “Variability as a Driver for the Adaptation of Tele-Rehabilitation Context-Aware Systems”, Knowledge-Based Systems, 2017 (To be submitted)

Type of venue: International Journal (IF: 3.325, Q1)
Variability as a Driver for the Adaptation of Tele-Rehabilitation Context-Aware Systems

Cristina Roda¹, Elena Navarro¹, Víctor López-Jaquero¹
¹LoUISE Research Group, University of Castilla-La Mancha, Spain
(Cristina.Roda, Elena.Navarro, VictorManuel.Lopez)@uclm.es

Abstract. It is worth noting that nowadays healthcare has become a must all over the world given the increasing aging of the population, and also the number of people affected with some type of impairment, such as Acquired Brain Injury (ABI). However, healthcare services are often provided at hospitals, neglecting the needs of these vulnerable social collectives, such as older people and people with ABI, which are usually impaired of moving, and, many times, they are also living in poverty. In this direction, e-Health and assisted rehabilitation have emerged to overcome these shortcomings. Tele-rehabilitation arises here as a solution to solve the lack of resources, both human and material ones, since it is able to bring rehabilitation closer to the patient’s home. Context-aware systems that can change its behaviour at runtime, depending on the surrounding conditions, are good candidates to be exploited in tele-rehabilitation domain. Nevertheless, the high level of variability in rehabilitation, especially for older people and people with ABI, should be taken into account to provide a suitable bespoke therapy. For this aim, this work proposes a variability approach for facilitating the adaptation of tele-rehabilitation context-aware systems. We present a feature model for modelling such variability in tele-rehabilitation domain in order to create cognitive rehabilitation activities for people affected with ABI. Moreover, it is also presented how context in these application domains could be modelled with the aim of facilitating adaptation tasks. Namely, we propose a user model for modelling the user characteristics, as well as an interaction model for modelling the interaction modalities available to interact with the system when carrying out the rehabilitation. Thus, the exploitation of all these models will provide facilities to adapt tele-rehabilitation context-aware systems to the current needs of the patients.

Keywords: variability, adaptation, tele-rehabilitation, feature model, interaction modality, context model, user model, context-aware system

1 Introduction

It is worth noting that nowadays governments and institutions all over the world are putting more and more effort on healthcare services given the increasing aging of the population, and also the increasing number of people with some type of disability during the last decades. Namely, since recently, people with disabilities have become an important research focus. As stated on the 10th Anniversary of the Convention on the Rights of Persons with Disabilities in 2016 [3], in the past decade, much progress was made in order to improve the quality of life of people with disabilities, but those persons continue to face grave disadvantages. They are still commonly denied fundamental rights, and are more likely to live in poverty. Moreover, people with disabilities also have more difficulty in accessing healthcare providers with appropriate skills. People with Acquired Brain Injury (ABI) belong to this collective of vulnerable people that requires continuous and customized attention from the very beginning of its treatment in order to provide them with enough quality of life. According to the Toronto ABI Network [6], in charge of connecting people with ABI to some services, ABI is related to a damage to the brain that occurs after birth, and which may be caused traumatically (e.g. from an external force), or through a medical problem or disease process (e.g. infection or aneurysm).

In this context, e-Health and assisted rehabilitation have emerged to overcome these shortcomings that people with disabilities usually express. Thus, tele-rehabilitation arises here as a solution to solve the lack of resources, both human and material ones, since it is able to bring rehabilitation closer to the patient’s home. This makes the therapy less expensive and also without the need of going to the hospital which, in most of the cases, constitutes a challenging task. People with ABI may require rehabilitation at three different levels: physical/motor, cognitive/intellectual or conductual/emotional. So, people with ABI may present physical/motor deficits which are disorders that affect the balance, the bipedalism or gait, or any other problem regarding any part of the body and its movements. With regard to cognitive/intellectual deficits, they
are related to problems in memory, attention, language, calculation, executive function, skills, gnosia, praxia or orientation. Finally, some people with ABI may present conductual/emotional deficits as well, which are related to behavioural problems, such as the lack of initiative or sociability problems.

Context-aware systems that can change its behaviour at runtime, depending on the surrounding conditions, are good candidates to be exploited in tele-rehabilitation domain, especially when dealing with older people or people with ABI. In this sense, if the system detects that, e.g. the patient is so fatigued when carrying out the rehabilitation activity, it can take the decision of changing the activity to a less-effort one, or even stopping the rehabilitation therapy, in order to fulfil the current needs of the patient. Nevertheless, the high level of variability in rehabilitation, especially for older people and people with ABI, should be taken into account to provide a suitable bespoke therapy. For this aim, this work proposes a variability approach for facilitating the adaptation of tele-rehabilitation context-aware systems. We present a feature model for modelling such variability in tele-rehabilitation domain in order to create cognitive rehabilitation activities for people affected with ABI. Moreover, it is also presented how context in these application domains could be modelled with the aim of facilitating adaptation tasks. Namely, we propose a user model for modelling the user characteristics, as well as an interaction model for modelling the interaction modalities available to interact with the system when carrying out the rehabilitation. Thus, the exploitation of all these models will provide facilities to adapt tele-rehabilitation context-aware systems to the current needs of the patients.

This work is structured as follows. Section 2 presents the rationale for modelling and managing variability regarding people in rehabilitation. Section 3 justifies the exploitation of models when designing rehabilitation systems for people with ABI. Section 4 presents how the ABI domain can be modelled, including the classification of ABI deficits, ABI groups, most popular devices used in ABI rehabilitation, and the activity patterns for ABI cognitive rehabilitation. Section 5 presents our approach for modelling the variability of the context in ABI rehabilitation. Section 6 depicts our proposal for modelling the user in ABI rehabilitation, which is based on the one proposed by Kaklanis et al. [8]. Section 7 describes how the interaction of the user with the rehabilitation system can be modelled to provide better interaction experiences by the adaptation of the interaction modality. Finally, Section 8 presents the conclusions and future lines derived from this work.

2 What is so Assorted in People in Rehabilitation?

In the field of Human-Computer Interaction, there are many different proposals for modelling the context. However, as stated in [2], there is a consensus that the three main cornerstones for defining the context are: User, Platform and Environment. The user dimension of context refers to relevant information for modelling the user of the system, such as his/her preferences, emotional state, skills, physiological data (heart rate, EEG, skin conductance, etc.), profile data (name, surname, age, gender, email, telephone, hobbies, job, etc.), or health state. The platform dimension of context refers to all hardware and software components of the system, such as the devices, applications, bandwidth, etc. The environment dimension of context refers to the surrounding conditions of the system, such as location, temperature, humidity, time, date, noise, light, etc. Over the other two context dimensions, the user one is widely exploited. The human side has been always appealing so many software systems follow a user-centred approach. In this sense, user needs and preferences, as well as any other characteristic of the user, guide the design, development and adaptation of those systems. Obviously, tele-rehabilitation systems are good candidates for following this user-centred approach since the patient (user) should be one of the key elements of the system in order to provide him/her the best rehabilitation therapy.

But modelling the user is a challenging task. Every user is unique and has many diverse features that make the difference with respect to the other individuals. Moreover, people with ABI and older people are two collectives widely assorted. When modelling the context of these concrete users for rehabilitation purposes, special features have to be considered. For example, the different deficits they can present, along with its degree of severity, as well as the degree of mobility of each body part, or the preferences when performing a rehabilitation activity, such as the font size or the pointer colour. Therefore, a rehabilitation therapy can be customized and adapted, depending on concrete characteristics of the user. In this sense, it is obvious that the better user modelling, the better bespoke rehabilitation therapy provision.

In addition to this, it is worth noting that nowadays there is a great diversity of hardware and novel devices that can be used in rehabilitation, such as MS Kinect, Nintendo Wii or Leap Motion. They are mainly used to monitor the user gestures, postures or even his/her emotions. Thus, the power to customize rehabilitation therapies using those novel devices for people with ABI and older persons justifies the need for variability beyond regular applications.
3 Exploiting Models to Design Rehabilitation Systems for a Diverse World

The design of rehabilitation systems is not an easy task. There are many different types of software required to provide a complete rehabilitation system in order to execute, edit, manage or monitor the rehabilitation therapies. Thus, finding a suitable mechanism is a must to assist in the design of such complex systems. Models appear here as this powerful artefact which can provide a common foundation to all this diverse software. As Paternò stated in [10], making systems easier to use implies taking into account many factors, such as tasks to support, context of use, user preferences, or the interaction techniques available. So this author encouraged designers to use structured methods (i.e. models) in order to manage such complexity. Models allow us to understand a complex problem and its potential solutions through abstraction [13]. So they aid us in understanding reality and lead our way to interact with it [10].

Models can be exploited in several ways to build a software system. For instance, a model-based design [10] aims at identifying high-level models to specify and analyse software applications from a more semantic-oriented level, rather than starting immediately to address the implementation level. By doing so, designers do not have to be strongly focused on implementation details which can confuse them. They can delegate the task of updating the implementation to specific tools to be consistent with such high-level models. On the other hand, a model-driven development goes beyond a model-based design since it has the peculiarity of generating software automatically from the corresponding models previously specified. However, despite all the benefits of abstraction and automation, these model-oriented methods are only as good as the models they help us construct [13].

In rehabilitation, and especially in ABI domain, there are many aspects to be modelled. As aforementioned in Section 2, the three main dimensions of context should be modelled given their importance for rehabilitation, as it is shown in our context metamodel proposal [11][12]. But especially those characteristics associated with the user are quite relevant in ABI rehabilitation, as well as other aspects, such as the rehabilitation activities the user can perform, or the possible interaction modalities between the user and the different devices.

All these arguments, presented in this section and the previous one, reflect the high variability of the context in ABI and rehabilitation domain and how the exploitation of models could be an excellent ally for designing and developing adaptive rehabilitation software. Thus, we have detected two different needs to be covered: (i) first, the need of modelling this context variability, and (ii) second, the need of modelling the context itself in a suitable manner to facilitate adaptations tasks based on models exploitation. By doing so, bespoke rehabilitation activities will be provided to the patients with ABI, depending on their characteristics and preferences, as well as depending on the rest of the surrounding context, such as the available devices. Thus, in this work, we have focused our effort on building models to be exploited for adapting tele-rehabilitation systems to older people and people with ABI, taking into account such context and its high variability.

Next sections exhibit these proposals for modelling variability (Section 5) and context (Sections 6 and 7) in rehabilitation scenarios for people with ABI, taking into account the domain of ABI (Section 4). Notice that this modelling has been always supported by the specialists [1][4].

4 Modelling the Domain of ABI

When dealing with people that present ABI, there are many aspects to be considered for providing them with the best rehabilitation therapy. Hence, this section describes the domain of ABI in terms of the different deficits the patient can reveal (Section 4.1), the most popular devices used in ABI rehabilitation (Section 4.2), and, finally, the activity patterns encountered for ABI cognitive rehabilitation (Section 4.3).

4.1 Deficits and Grade of ABI

People with ABI can present several deficits, classified in three main categories, according to the part of the brain damaged and/or the gravity of the injury: physical/motor deficits, cognitive/intellectual deficits, and conductual/emotional deficits. Physical/motor deficits are related to those disorders that affect the balance, the bipedalism or gait, or any other problem regarding any part of the body and its movements. These physical/motor deficits can be classified in 10 different categories, as presented in Table 1.
### Physical/Motor deficits

<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Hemiparesis</td>
</tr>
<tr>
<td>P2</td>
<td>Balance problems</td>
</tr>
<tr>
<td>P3</td>
<td>Bipedalism problems</td>
</tr>
<tr>
<td>P4</td>
<td>Gait problems</td>
</tr>
<tr>
<td>P5</td>
<td>Head and trunk control problems at rest</td>
</tr>
<tr>
<td>P6</td>
<td>Shaking and trembling (tic)</td>
</tr>
<tr>
<td>P7</td>
<td>Shaking and trembling (epilepsy)</td>
</tr>
<tr>
<td>P8</td>
<td>Gross motor skills (upper)</td>
</tr>
<tr>
<td>P9</td>
<td>Fine motor skills (upper)</td>
</tr>
<tr>
<td>P10</td>
<td>Dysarthria</td>
</tr>
</tbody>
</table>

Table 1. Physical/Motor deficits in ABI

Cognitive/intellectual deficits are related to problems in memory, attention, language, calculation, executive function, skills, gnosia, praxia or orientation. There are 36 different cognitive/intellectual deficits which are listed in Table 2.

<table>
<thead>
<tr>
<th>Cognitive/Intellectual deficits</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Memory</strong></td>
<td></td>
</tr>
<tr>
<td>Short-term or verbal memory</td>
<td>C1</td>
</tr>
<tr>
<td>Short-term or spatial memory</td>
<td>C2</td>
</tr>
<tr>
<td>Historical episodic memory</td>
<td>C3</td>
</tr>
<tr>
<td>Autobiographical episodic memory</td>
<td>C4</td>
</tr>
<tr>
<td>Semantic memory</td>
<td>C5</td>
</tr>
<tr>
<td>Procedural memory</td>
<td>C6</td>
</tr>
<tr>
<td>Working memory (Activities in daily life)</td>
<td>C7</td>
</tr>
<tr>
<td><strong>Attention</strong></td>
<td></td>
</tr>
<tr>
<td>Loss of attention (sustained)</td>
<td>C8</td>
</tr>
<tr>
<td>Loss of attention (divided)</td>
<td>C9</td>
</tr>
<tr>
<td>Oral comprehension</td>
<td>C10</td>
</tr>
<tr>
<td>Written comprehension</td>
<td>C11</td>
</tr>
<tr>
<td><strong>Language</strong></td>
<td></td>
</tr>
<tr>
<td>Reading comprehension</td>
<td>C12</td>
</tr>
<tr>
<td>Oral expression</td>
<td>C13</td>
</tr>
<tr>
<td>Written expression</td>
<td>C14</td>
</tr>
<tr>
<td>Reading expression</td>
<td>C15</td>
</tr>
<tr>
<td><strong>Calculation</strong></td>
<td></td>
</tr>
<tr>
<td>Calculation</td>
<td>C16</td>
</tr>
<tr>
<td><strong>Executive Function</strong></td>
<td></td>
</tr>
<tr>
<td>Logical-deductive reasoning</td>
<td>C17</td>
</tr>
<tr>
<td>Problem solving</td>
<td>C18</td>
</tr>
<tr>
<td>Cognitive flexibility</td>
<td>C19</td>
</tr>
<tr>
<td>Planning</td>
<td>C20</td>
</tr>
<tr>
<td>Verbal fluency and flow (Aphasia deficit)</td>
<td>C21</td>
</tr>
<tr>
<td>Categorization</td>
<td>C22</td>
</tr>
<tr>
<td><strong>Skills</strong></td>
<td></td>
</tr>
<tr>
<td>Visual-constructive skills</td>
<td>C23</td>
</tr>
</tbody>
</table>
Table 2. Cognitive/Intellectual deficits in ABI

Finally, conductual/emotional deficits are related to behavior problems exposed by the patient. We can distinguish 5 different conductual/emotional deficits, as seen in Table 3.

Table 3. Conductual/Emotional deficits in ABI

It is worth noting that the same person can exhibit more than one type of deficit at the same time, e.g. physical/motor deficits, such as P2, P3 and P10, and cognitive/intellectual deficits, such as C13, C20, C30, C31 and C33. Hence, it is really challenging to generalize when we are talking about brain injuries. No two brains are alike before a lesion, neither after it. This variability should be considered to provide the best bespoke rehabilitation therapy to every patient.

In addition, people with ABI can be classified into 5 groups or grades, according to the deficits they present, to facilitate their rehabilitation (notice that a person affected with ABI can only and must belong to one group):

- **Group A**: Low or no physical deficit and severe psychologic dysfunction
- **Group B**: Low or no physical deficit and moderate or low psychologic dysfunction
- **Group C**: Physical deficit with low or no psychologic dysfunction
- **Group D**: Physical deficit with low or no psychologic dysfunction. Maintenance.
- **Group E**: Coma or vegetative

### 4.2 Popular Devices Used in ABI Rehabilitation

It is also interesting to know which devices are widely used in ABI rehabilitation for supporting therapists when dealing with such patients. The devices most commonly used are listed below:

- Smartphone
The power of considering as much devices as possible facilitates the adaptation of the system to the different needs, preferences and user characteristics. For example, in terms of adapting the interaction modality, if the patient is blind, the system can avoid using the monitor and use instead the speakers to provide him/her with feedback.

4.3 Activity Patterns for ABI Cognitive Rehabilitation

There are many different activities for rehabilitating people with ABI at every level, i.e. physical, cognitive and emotional. Five different activity patterns for cognitive rehabilitation were already identified in previous work [9] (see Fig. 1). Each one of them is focused on the main cognitive function stimulated: Front executive, Attentional, Mnesic, Constructive and Linguistic. In this work, we are focused on this cognitive side with the aim of providing the cognitive rehabilitation activity that best fits the patient needs.

![Activity patterns for ABI cognitive rehabilitation](image)

Fig. 1. Activity patterns for ABI cognitive rehabilitation

Namely, we are interested in the creation of association activities which belong to the front executive activity pattern. During the execution of these activities, the patient has to associate one element to another which can be images, texts, sounds, symbols or praxias. Different types of association activities can be performed, depending on the elements to associate:

- Image - Image
- Image - Sound
- Image - Symbol
- Image - Text
In conclusion, all these aspects that define and model the domain of ABI, such as the different type of deficits, the ABI group, the novel and popular devices used, as well as the activity proposed for rehabilitation, should be carefully considered because of the high variability it entails. For the purpose of managing and modeling such context variability in ABI rehabilitation, we propose a feature model that will be presented in detail in the next section.

5 Modelling the Variability of the Context in ABI Rehabilitation

Now, we know that we have a variable context when dealing with rehabilitation therapies for people with ABI. Thus, we propose modelling such variability as it is done in Software Product Lines (SPLs) to define and manage the commonalities and variabilities encountered in ABI rehabilitation. By doing so, we will be able to provide the best rehabilitation activity adapted to every patient. In order to tackle and model such variability, there are numerous approaches that can be applied. However, as stated in [5], there are two approaches widely accepted by far: Feature Modelling (FM) and Decision Modelling (DM). DM is based on the creation of a decision model which is defined as [5] a set of decisions that are adequate to distinguish among the members of an application engineering product family and to guide adaptation of application engineering work products. On the other hand, FM is focused on capturing features (the end-user’s understanding of the general capabilities of systems in the domain), and the relationships among them, by means of feature models [5].

Among these two approaches, DM and FM, we have chosen FM since, as stated in [7], it is currently a standard de facto to support variability management and, apart from variable features, it also allows us to express the common ones, in contrast to DM. Furthermore, as Czarnecki et al. highlight in [5], DM emphasizes too much the product derivation, as opposed to describing the domain, which is the main purpose of FM and ours too. In the following sections, our feature model for modelling context variability in ABI rehabilitation domain is described (Section 5.1), as well as the different constraints that every valid configuration of features should fulfil (Section 5.2). It is also presented an example in order to show how a bespoke rehabilitation activity can be provided by exploiting this feature model (Section 5.3).

5.1 Our Feature Model

As mentioned before, our feature model has been designed to create rehabilitation activities for people with ABI. We have used the FeatureIDE framework presented in [14] in order to create such feature model. This framework enables the specification and validation of feature models. With this aim, FeatureIDE provides support for the creation of abstract and concrete features, as well as for the different types of connections between a feature and its set of sub-features: mandatory (logic and), logic or, and alternative. Moreover, it also provides a constraints editor to specify the different cross-tree restrictions within the feature model. All these elements supported by FeatureIDE will be further explained while describing our feature model.

![Fig. 2. Feature model (simplified view)]
Fig. 3. Feature model (partial view: ABI deficit type)
Fig. 4. Feature model (partial view: ABI grade and devices)
Fig. 5. Feature model (partial view: activity types for cognitive rehabilitation in ABI)

Fig. 6. Feature model (partial view: association activities for cognitive rehabilitation in ABI)
The simplified view of our feature model is presented in Fig. 2. This view only shows the main (abstract) features (without its sub-features) that we have to consider in the creation of rehabilitation activities for ABI patients. Namely, the type of ABI deficit, the grade of ABI, the devices, and the type of rehabilitation activity. Note that all these features are mandatory so, if we follow the model from left to right:

1. First, we have to select the deficits presented in the patient with ABI
2. Second, we have to select the ABI grade of the patient
3. Third, we have to specify the available devices for performing the rehabilitation task
4. And, finally, according to the previous characteristics selected, the system will return the different types of activities the patient could perform for the rehabilitation

Now, we are going to detail each one of these four abstract features, presenting all its sub-features and the type of connections between them. Firstly, Fig. 3 depicts the different ABI deficits that we have modelled by following the categorization presented in Section 4.1.

Virtually, all connections in this sub-tree are or- ones which means that we can select whatever deficits necessary to characterize a patient and they can be associated with any type of deficit. So, for example, the same patient can present the physical deficits P1 and P7, as well as the cognitive deficits C5, C6, C19, C32 and C33, but no emotional ones. Notice that whenever a patient suffers from Calculation problems, the feature C16 must be always selected since there are no other features to characterize such type of deficit. Then, regarding the ABI grade (see Fig. 4), a patient can only and must belong to a unique ABI group or grade. Hence, the sub-features related to those grades (A, B, C, D and E) are grouped as alternatives in our feature model.

Secondly, regarding the devices (see Fig. 4), we have represented in the feature model the most popular ones used in ABI rehabilitation, as explained in Section 4.2. Note that this model allows us to select as many type of devices as needed so all of them are connected by an or- relation.

Regarding the type of rehabilitation activity (see Fig. 5), we have modelled all activity patterns related to cognitive rehabilitation as we previously presented in Section 4.3. All these activities are grouped as alternatives since only one activity will be finally provided to the user. As aforementioned, this work is focused on the creation of association activities, so the feature model only presents the different types of association activities for the shake of clarity and legibility, as seen in Fig. 6.

5.2 Constraints in the Configuration of Features

In short, from a specific selection of features (deficits and ABI group of the patient, and also the available devices), an association activity for the cognitive rehabilitation of a concrete patient with ABI is suggested. In this way, at the end of the whole process, an outcome is provided in the form of configuration of features regarding the deficits and ABI group of the patient, available devices, and association activity recommended. However, not every configuration is valid, and only the valid ones will be finally returned by the system. A configuration of features will be valid if it fulfils all constrains specified for such feature model. With regard to the use of some devices, there are some limitations. For example, to use a smartphone during the rehabilitation activity, the person affected must not present none of the following deficits: P1, P5 and P9.

This constrain is established as follows:

\[
\text{Smartphone} \Rightarrow \neg P1 \land \neg P5 \land \neg P9
\]

There are also other constraints regarding the type of association activity the patient can perform, depending on his/her deficits and ABI grade. For example, the following restriction is related to the association activity Image-Image in which the patient has to associate two images:

\[
\text{Image-Image} \Rightarrow (P8 \lor P9 \lor C8 \lor C25) \land A \lor (P8 \lor P9 \lor C8 \lor C25 \lor C1 \lor C2 \lor C9) \land B \lor (P8 \lor P9 \lor C8 \lor C25 \lor C1 \lor C2 \lor C9 \lor C5 \lor C7 \lor C17 \lor C19 \lor C22 \lor C24) \land (C \lor D)
\]

This constrain indicates that this type of association activity:

i. Tackles the deficits P8, P9, C8 and/or C25 of persons affected with A grade, specified as: \((P8 \lor P9 \lor C8 \lor C25) \land A\)

ii. Apart from the previous deficits (i), the deficits C1, C2 and/or C9 of persons affected with B grade can also be treated, specified as: \((P8 \lor P9 \lor C8 \lor C25 \lor C1 \lor C2 \lor C9) \land B\)

iii. Finally, apart from all the previous deficits (i and ii), the deficits C5, C7, C17, C19, C22 and/or C24 of persons affected with C or D grade can also be treated, specified as: \((P8 \lor P9 \lor C8 \lor C25 \lor C1 \lor C2 \lor C9 \lor C5 \lor C7 \lor C17 \lor C19 \lor C22 \lor C24) \land (C \lor D)\)
Finally, there are also constraints regarding the ABI grade the patient can have, depending on his/her deficits. For example, giving the following constraint:

$$A \Rightarrow C_1 \lor C_5 \lor C_9 \lor C_{10} \lor C_{11} \lor C_{12} \lor C_{13} \lor C_{14} \lor C_{15} \lor C_{16} \lor C_{19} \lor C_{21} \lor C_{22} \lor C_{30} \lor C_{31} \lor C_{32} \lor C_{33} \lor C_{35} \lor E_2 \lor E_3$$

It means that one or more of these deficits should be presented in the patient in order to consider him/her as an A-grade ABI patient. Finally, a summary of all the constraints specified to validate the different configurations of features in our feature model are shown in Table 4, Table 5, and Table 6. The constraints presented in these tables are related to the **ABI grade** of the patient, the **devices** that can be used during the rehabilitation, and the **association activities** that can be performed, respectively. Note that all these restrictions depend on the ABI deficits of the patient, except constraints for association activities which also depends on the ABI grade of the patient.

<table>
<thead>
<tr>
<th>Table 4. Constraints regarding the ABI grade</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ABI grade</strong></td>
</tr>
<tr>
<td>A</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>D</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 5. Constraints regarding the devices</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Devices</strong></td>
</tr>
<tr>
<td>Smartphone $\lor$ Tablet $\lor$ Webcam $\lor$ Speakers $\lor$ Headphones</td>
</tr>
<tr>
<td>Keyboard $\lor$ Mouse $\lor$ Digital_pad $\lor$ Optical_pen</td>
</tr>
<tr>
<td>Microphone</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 6. Constraints regarding the association activities</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Association activity</strong></td>
</tr>
<tr>
<td><strong>Image_Image</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Image_Sound</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>Image_Symbol</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
13. Variability as a Driver for the Adaptation of Tele-Rehabilitation Context-Aware Systems

| Praxia_Writing | \( (P9 \lor C8 \lor C11 \lor C25) \land A \)
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Rightarrow )</td>
<td>( \lor (P9 \lor C8 \lor C11 \lor C25 \lor C1 \lor C2 \lor C9 \lor C23 \lor C24) \land B )</td>
</tr>
<tr>
<td></td>
<td>( \lor (P9 \lor C8 \lor C11 \lor C25 \lor C1 \lor C2 \lor C9 \lor C23 \lor C24 \lor C5 \lor C7 \lor C17 \lor C19 \lor C22) \land (C \lor D) )</td>
</tr>
</tbody>
</table>

| Sound_Sound | \( (P8 \lor P9 \lor P10 \lor C5 \lor C10 \lor C13 \lor C26 \lor C33) \land A \)
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Rightarrow )</td>
<td>( \lor (P8 \lor P9 \lor P10 \lor C8 \lor C10 \lor C13 \lor C26 \lor C33 \lor C1 \lor C2 \lor C9 \lor C12) \land B )</td>
</tr>
<tr>
<td></td>
<td>( \lor (P8 \lor P9 \lor P10 \lor C8 \lor C10 \lor C13 \lor C26 \lor C33 \lor C1 \lor C2 \lor C9 \lor C12 \lor C5 \lor C7 \lor C17 \lor C19 \lor C22 \lor C24) \land (C \lor D) )</td>
</tr>
</tbody>
</table>

| Sound_Symbol | \( (P8 \lor P10 \lor C8 \lor C10 \lor C13 \lor C25 \lor C26 \lor C33) \land A \)
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Rightarrow )</td>
<td>( \lor (P8 \lor P10 \lor C8 \lor C10 \lor C13 \lor C25 \lor C26 \lor C33 \lor C2 \lor C9 \lor C12 \lor C16 \lor C23 \lor C24) \land B )</td>
</tr>
<tr>
<td></td>
<td>( \lor (P8 \lor P10 \lor C8 \lor C10 \lor C13 \lor C25 \lor C26 \lor C33 \lor C2 \lor C9 \lor C12 \lor C16 \lor C23 \lor C24 \lor C5 \lor C7 \lor C17 \lor C19 \lor C22) \land (C \lor D) )</td>
</tr>
</tbody>
</table>

| Sound_Writing | \( (P8 \lor P9 \lor P10 \lor C8 \lor C10 \lor C13 \lor C14 \lor C25 \lor C26 \lor C33) \land A \)
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Rightarrow )</td>
<td>( \lor (P8 \lor P9 \lor P10 \lor C8 \lor C10 \lor C13 \lor C14 \lor C25 \lor C26 \lor C33 \lor C1 \lor C9 \lor C12 \lor C23 \lor C24) \land B )</td>
</tr>
<tr>
<td></td>
<td>( \lor (P8 \lor P9 \lor P10 \lor C8 \lor C10 \lor C13 \lor C14 \lor C25 \lor C26 \lor C33 \lor C1 \lor C9 \lor C12 \lor C23 \lor C24 \lor C5 \lor C7 \lor C17 \lor C19 \lor C22) \land (C \lor D) )</td>
</tr>
</tbody>
</table>

| Writing_Writing | \( (P8 \lor P9 \lor P10 \lor C8 \lor C11 \lor C14 \lor C25) \land A \)
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Rightarrow )</td>
<td>( \lor (P8 \lor P9 \lor P10 \lor C8 \lor C11 \lor C14 \lor C25 \lor C1 \lor C2 \lor C9 \lor C16 \lor C23 \lor C24) \land B )</td>
</tr>
<tr>
<td></td>
<td>( \lor (P8 \lor P9 \lor P10 \lor C8 \lor C11 \lor C14 \lor C25 \lor C1 \lor C2 \lor C9 \lor C16 \lor C23 \lor C24 \lor C5 \lor C7 \lor C17 \lor C19 \lor C22 \land (C \lor D) )</td>
</tr>
</tbody>
</table>

| Writing_Symbol | \( (P8 \lor P9 \lor P10 \lor C8 \lor C11 \lor C14 \lor C25) \land A \)
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Rightarrow )</td>
<td>( \lor (P8 \lor P9 \lor C8 \lor C11 \lor C14 \lor C25 \lor C1 \lor C2 \lor C9 \lor C16 \lor C23 \lor C24 \land B )</td>
</tr>
<tr>
<td></td>
<td>( \lor (P8 \lor P9 \lor C8 \lor C11 \lor C14 \lor C25 \lor C1 \lor C2 \lor C9 \lor C16 \lor C23 \lor C24 \lor C5 \lor C7 \lor C17 \lor C19 \lor C22 \land (C \lor D) )</td>
</tr>
</tbody>
</table>

| Praxia_Image | \( (P8 \lor P9 \lor P10 \lor C5 \lor C10 \lor C13 \lor C23 \lor C24 \lor C30 \lor C31 \lor C32 \lor C33) \land A \)
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Rightarrow )</td>
<td>( \lor (P8 \lor P9 \lor P10 \lor C8 \lor C10 \lor C13 \lor C23 \lor C24 \lor C30 \lor C31 \lor C32 \lor C33 \lor C1 \lor C2 \lor C9 \lor C12) \land B )</td>
</tr>
<tr>
<td></td>
<td>( \lor (P8 \lor P9 \lor P10 \lor C8 \lor C10 \lor C13 \lor C23 \lor C24 \lor C30 \lor C31 \lor C32 \lor C33 \lor C1 \lor C2 \lor C9 \lor C12 \lor C5 \lor C6 \lor C7 \lor C17 \lor C19 \lor C22) \land (C \lor D) )</td>
</tr>
</tbody>
</table>

| Praxia_Sound | \( (P8 \lor P9 \lor P10 \lor C5 \lor C11 \lor C14 \lor C23 \lor C24 \lor C30 \lor C31 \lor C32 \lor C33) \land A \)
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Rightarrow )</td>
<td>( \lor (P8 \lor P9 \lor P10 \lor C8 \lor C11 \lor C14 \lor C23 \lor C24 \lor C30 \lor C31 \lor C32 \lor C33 \lor C1 \lor C2 \lor C9) \land B )</td>
</tr>
<tr>
<td></td>
<td>( \lor (P8 \lor P9 \lor C8 \lor C11 \lor C14 \lor C23 \lor C24 \lor C30 \lor C31 \lor C32 \lor C33 \lor C1 \lor C2 \lor C9 \lor C5 \lor C6 \lor C7 \lor C17 \lor C19 \lor C22 \land (C \lor D) )</td>
</tr>
</tbody>
</table>

| Praxia_Writing | \( (P8 \lor P9 \lor C8 \lor C23 \lor C24 \lor C30 \lor C31 \lor C32 \lor C33) \land A \)
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Rightarrow )</td>
<td>( (P8 \lor P9 \lor C8 \lor C23 \lor C24 \lor C30 \lor C31 \lor C32 \lor C33) \land A )</td>
</tr>
</tbody>
</table>
5.3 Providing a Bespoke Cognitive Rehabilitation Activity: an Example

In this section, an example is shown in order to understand how this feature model can be exploited for providing a bespoke cognitive rehabilitation activity to a specific patient. For example, we specify that the patient presents the following deficits: P5, P10, C1, C6, E2 and E3, as seen in Fig. 7.

Then, we also specify that the patient belongs to the A group, and, as seen in Fig. 8 as grey font, there are some devices that are not allowed to be used in this rehabilitation therapy, given the restrictions previously established (see Table 5). Thus, the available devices for this concrete patient are: a Monitor, a MS Kinect, a Leap Motion and a Wii. We select the Monitor and MS Kinect devices, so the patient will be able to perform some gestures in front of the MS Kinect to complete the activity, following the instructions depicted in the Monitor.

Then, the system shows the possible cognitive rehabilitation activities that can be performed by this particular patient (see Fig. 9). As aforementioned, we have especial interest in association activities and, as seen in this figure, there are some of them available for this patient (the ones with green font), such as Image-Sound or Sound-Symbol, that will provide a valid configuration of features. However, there are others restricted (those with black font), such as Image-Image or Writing-Symbol, since they will be not adequate to treat those ABI deficits previously established.
Fig. 8. ABI grade and devices allowed, depending on the ABI deficits

Fig. 9. Possible cognitive rehabilitation activities, depending on the features selected
6 Modelling the User in ABI Rehabilitation

This section describes our proposal for modelling the user in ABI rehabilitation, which is based on the user model proposed by Kaklanis et al. [8]. In the following sections, our user model is explained in detail by comparing it with the Kaklanis’ one.

6.1 User Attributes and Preferences

As seen in Fig. 10, a user in our system can be described by means of some attributes, such as his/her age, name, sex, occupation before the brain injury, etc. In the user model presented by Kaklanis et al. [8], similar characteristics are modelled separately in a new entity. By contrast, our user model includes those user features as attributes of the User entity since we consider this is a better way to specify such kind of user information.

The user Preferences are modelled as a separate entity from the User one. Our user model has some attributes in common with Kaklanis’ one, such as FontColour, FontSize, PointerColour or PointerSize. In this sense, the user interface will be adapted to those preferences to make it friendlier. We have also included others since they represent relevant requirements of tele-rehabilitation systems for people with ABI. For example, this social collective like knowing what is the time consumed while performing an activity (ActivityTimeConsumed), as well as the current date or the current time to avoid disorientation and frustration. Moreover, it is also interesting to consider the preferred activity topics of the users in order to motivate them by providing more attractive and customized activities. Finally, it is also useful to give the specific option to the user of spoken instructions when learning how to carry out a particular activity. In this manner, the fact of being aware of such information makes these patients feeling more comfortable and relaxed during the therapy process. Notice that we have considered all this information about the user in collaboration with experts and specialists in the domain of rehabilitation [1][4].

Fig. 10. User model (partial view: user attributes and preferences)

6.2 ABI User Deficits

Kaklanis et al. [8] identify some cognitive variables when characterizing a user, such as behaviour or working memory. However, as previously seen, when talking about people affected with ABI, there are many more issues to consider, so we have modelled this in a different manner. A user can manifest not only cognitive/intellectual deficits, but also physical/motor ones. Furthermore, our deficit specialization includes conductual/emotional as another type of deficit, differing in the way Kaklanis et al. present such information.
For example, they model the behaviour of the user as part of her cognitive skills, while, by contrast, we separate these concepts into two different type of deficits: Conductual/Emotional and Cognitive/Intellectual. The rationale that has supported such modelling decision belongs to the way professionals in the ABI field [1] classify every type of deficit in order to treat each patient in a customized manner.

Fig. 11 shows how our user model integrates all these elements and relationships. This design also enables to specify the percentage of seriousness regarding each deficit (Severity attribute in the Deficit entity). We have modelled the three main types of ABI deficits in three single entities, following the categorization presented in Section 4.1: PhysicalMotorDeficit, Cognitive/IntellectualDeficit and Conductual/EmotionalDeficit. Every deficit category includes an attribute, called Type, to indicate the concrete deficit the patient suffers, e.g. C6 regarding MemoryDeficit.

6.3 Body Parts of the User

Against the work of Kaklanis et al. [8], we have modelled the body parts of the user slightly different. Mainly, the body parts that we have included in our user model are those that can be detected by Microsoft Kinect and Leap Motion devices. The primary reason of doing so is that nowadays these types of devices are widely used, especially in the context of tele-rehabilitation systems. Such both gadgets are considered part of the new generation of devices used in post-WIMP applications which go beyond the initial interaction and presentation techniques through Windows, Icons, Menus and Pointers. In this sense, the body parts included for MS Kinect detection are the following ones:

- Head
- Shoulder (Left, Center and Right)
- Elbow (Left and Right)
- Wrist (Left and Right)
- Hand (Left and Right)
• Spine
• Hip (Left, Center and Right)
• Knee (Left and Right)
• Ankle (Left and Right)
• Foot (Left and Right)

Leap Motion is able to detect the hand, along with the palm and wrist position. It also detects the following parts of every finger (Thumb, Index, Middle, Ring and Pinky):

• Tip
• Distal phalanx
• Intermediate phalanx
• Proximal phalanx
• Metacarpal (except for the Thumb finger which has not got this bone)

Besides these body parts, we have also modelled some face/head parts, such as the Ears, Mouth and Eyes, due to the increasingly use of new devices that are emerging to recognize or interact with them. Therefore, as seen in Fig. 12, all body parts of the user are classified by four fairly differentiated groups: Head, Trunk, UpperLimb and LowerLimb.

Initially, we have not included any anthropometric variables with respect to the work presented by Kaklanis et al. with the aim of avoiding a highly complex user model in its preliminary version.

6.4 Body Movements

Both upper and lower limbs movements have been taken into account in our user model based on the mobility elements presented by Kaklanis et al. [8]. Nevertheless, for the sake of clarity, only the movements of the upper limbs will be described here.

As shown in Fig. 13, at the upper limb level, we can distinguish four types of movements, carried out by the whole arm:

• InForceMove: The in force performed when one hand comes closer to the chest.
• OutForceMove: The out force performed when one hand moves far from the chest.
• PushForceMove: The pushing force performed.
• PullForceMove: The pulling force performed.

On the other hand, there are specific movements that only belong to specific body parts. For instance, the Shoulder can make the following moves:

• TorqueMove: Torque produced in the particular shoulder.
• ExternalRotationMove: Rotary movement around the longitudinal axis of the bone away from the centre of the body, turning the arm outward.
• InternalRotationMove: Rotary movement around the longitudinal axis of the bone toward the centre of the body, turning the arm inward.
• FlexionMove: Flexion of the shoulder moves the limb forward (towards the anterior side of the body).
• ExtensionMove: Extension of the shoulder moves the limb backward (towards the posterior side of the body).
• AbductionMove: Abduction occurs when the arms are held at the sides, parallel to the length of the torso, and are then raised in the plane of the torso. This movement may be broken down into two parts: true abduction of the arm, which takes the humerus from parallel to the spine to perpendicular; and upward rotation of the scapular, which raises the humerus above the shoulders until it points straight upwards.
• AdductionMove: Adduction is the opposite motion of arm abduction. It can be broken down into two parts: downward rotation of the scapula and true adduction of the arm.

With respect to the Elbow, it can perform the following movements:

• TorqueMove: Torque produced in the particular elbow.
• FlexionMove: Elbow flexion is a movement that occurs when the arm is bent at the elbow and the forearm and the arm come together.
• HyperExtensionMove: The opposite of elbow flexion, during which the arm is straightened and the forearm and arm move away from one another.
Fig. 12. User model (partial view: body parts of the user)
The Wrist motions are:

- **FlexionMove**: Bending the joint resulting in a decrease of angle; moving the palm of the hand toward the front of the forearm.
- **ExtensionMove**: Straightening the joint resulting in an increase of angle; moving the back of the hand toward the back of the forearm.
- **RadialDeviationMove**: Lateral movement away from the midline of the body; moving the thumb side of the hand toward the lateral side of the forearm.
- **UlnarDeviationMove**: Medial movement toward the midline of the body; moving the little finger side of the hand toward the medial side of the forearm.

Regarding the movements carried out by the **Hand**:

- **PronationMove**: Pronation is the rotation of the forearm that moves the palm from an anterior-facing position to a posterior-facing position, or palm facing down. The hand is supine (facing anteriorly) in the anatomical position.
- **SupinationMove**: Supination is the opposite of pronation, the rotation of the forearm so that the palm faces anteriorly, or palm facing up.

Finally, the **Finger** movements are classified depending on the phalanx in charge of its performing:

- DistalPhalanx
  - **FlexionMove**: Flexion of the distal phalanx.
  - **HyperExtensionMove**: Hyper-extension of the distal phalanx.
- IntermediatePhalanx
7 Modelling the Interaction of the User with the Rehabilitation System

Apart from modelling the previous user capabilities, there are also other elements that are good candidates to be modelled in order to facilitate adaptation tasks in rehabilitation systems. In this way, the idea of a system able to change the interaction modalities, according to the user needs and preferences, and the surrounding context in general, is much more than appealing. For this purpose, we have designed an interaction model which enables the specification of Interaction Modalities (IMs) the user and the possible devices of the system could present.

Formally, interaction modality is defined as [15] the “sensory channel used by a communicating agent to convey information to a communication partner, e.g. spoken language, intonation, gaze, hand gestures, body gestures or facial expressions”. IMs can be classified in two main categories: Output Interaction Modalities and Input Interaction Modalities. The former is related to the different communication channels used by the system to provide some information/feedback to the user, as well as by the user to the system. The latter is related to the different communication channels supported by the system to receive some information/feedback from the user, as well as by the user from the system. In the following sections, both output IMs and input IMs are described from the point of view of the user (Section 7.1) and the devices (Section 7.2). Finally, Section 7.3 presents the relationships between the IMs of the user and the IMs of the devices in order to know how the user-system interaction can be performed. That knowledge about what communication channels are available between the user and the system will provide a practical guide to assist in the task of adapting the interaction modalities of the system, depending on the user needs and capabilities.

7.1 Interaction Modalities of the User

In this section, we present the portion of the interaction model related to the different IMs of the user, both input and output ones. Notice that our vision about the interaction modalities of the user is not the same as the proposal of Kaklanis et al. [8]. They distinguish three main variables: visual, auditory and speech variables, reaching a wider detail level when describing the user in such terms. In our case, we have modelled the user IMs relevant for the adaptation process in post-WIMP applications used by people with ABI. Furthermore, it is also worth noting that we have not included these user IMs in the previous user model for the sake of clarity.

7.1.1 User Input IMs

The user can present up to five different input IMs since each one of them is related to a different sense, so the system is able to interact with him. Thus, the user can perceive stimuli sent by the system through the following five natural communication channels (see Fig. 14):

- **Visual.** The user is able to perceive visual signals sent by the system by means of his eyes, e.g. an image of a family member in order to associate it with the corresponding name while performing cognitive rehabilitation activities.
- **Auditory.** The user is able to perceive auditory signals sent by the system by means of his ears, e.g. spoken instructions about a rehabilitation exercise.
- **Olfactory.** The user is able to perceive olfactory signals sent by the system by means of his nose, e.g. a smell of grass in order to associate it with a picture of grass while performing cognitive rehabilitation activities.
• **Gustatory.** The user is able to perceive gustatory signals sent by the system by means of his mouth, e.g. a flavour of mint in order to associate it with the text “mint” while performing cognitive rehabilitation activities.

• **Haptic.** The user is able to perceive haptic signals sent by the system by means of many parts of his body, such as the fingertips or any part with nerve endings, e.g. a strong vibration in his right hand while performing physical rehabilitation activities in order to advise him that he has reached the objective.

The class `UserInputIM` has the attribute `Accuracy` which indicates the percentage of correctness regarding a user input IM. In addition, there is also another useful attribute, `SensitivityDegree`, which indicates the percentage of sensitivity regarding each sense. For example, `GustatorySensitivityDegree` represents the percentage of sensitivity of the mouth to perceive different flavours. Note that the `SensitivityDegree` attribute appears in those classes related to body parts able to perceive a particular stimulus. In the current example, `GustatorySensitivityDegree` only appears in the `Mouth` class since it is the only part of the body that can perceive gustatory stimulus. By means of this attribute, the system will be able to select the interaction modality that best fits the user needs to interact with him.

As seen in Fig. 14, all the input IMs are connected to some body parts, namely to those body parts that perceive the corresponding stimulus as input. For instance, the `Visual` IM is related to the `Eyes` of the user as this is the part of the body that perceives visual stimuli.

### 7.1.2 User Output IMs

If the user is wearing some kind of heart rate sensor, he is providing heart rate data (`HeartRate` entity in the interaction model) to the system implicitly. Other example of output IM is a voluntary gesture done by the user with both arms as part of the therapy (`BodyGesture` entity). Hence, the user can present up to three different output IMs, so that he can interact with the system. In this sense, the system is able to perceive stimuli sent by the user by means of these three different communication channels (see. Fig. 14):

• **Speech,** vocal abilities of the user used to communicate with the system, including:
  o **Phonation,** muscular work for making an intelligible sound which enables oral communication. Its main purpose is articulating words.
  o **Prosody,** intonation and accentuation of the articulated words.

• **Gesture,** movement the user carries out to interact with the system. This motion can be done in three different ways to be captured by the system:
  o **GestureWithTangible,** a gesture with a tangible object whose movements or presence are tracked by the system, e.g. the tag objects provided by Microsoft PixelSense or the remote controls of Nintendo Wii. In this case, the interaction with the system is done by means of these tangible objects, so that our system should be able to interpret the input data sent by those objects.
  o **BodyGesture,** a gesture made by any part of the body that must be captured by specific devices able to interpret such moves, such as MS Kinect or Leap Motion. Notice that, in this case, the gesture is tracked, but without the support of any gadget or device, in contrast to the previous case.
  o **TouchGesture,** a gesture made by the fingers or hand palms by touching a concrete tactile device, e.g. a double touch with the index finger or a rotation gesture made with the index and thumb fingers in the clockwise direction in a touch screen.

• **Biosignal,** any signal in living beings that is constantly been measured and monitored so the user is continuously sending it in an implicit way, i.e. without his/her explicit knowledge and unintentionally. Namely, for the time being, we have considered the following biosignals since they constitute crucial variables to be controlled in patients during rehabilitation activities:
  o **HeartRate,** the heart rate of the user.
  o **EEG,** the Electroencephalography signal of the user.
  o **BodyTemperature,** the temperature of the user body.
  o **Balance,** the ability of the user to maintain the line of gravity of his body.
  o **SkinConductance,** the electrodermal activity of the user body.

The entity `UserOutputIM` has also the attribute `Accuracy` to indicate the correctness of a concrete user output IM. In the case of the Speech modality, there must be a minimum level of accuracy from which the user was able to articulate, at least, some sounds (not necessary words) that have to be considered as well as other type of interaction. As seen in Fig. 14, all the output IMs are connected to some body parts, namely to those body parts that perform such output. For instance, the `Speech` IM is related to the `Mouth` of the user as this is the part of the body that emits sounds for making a speech.
As aforementioned, we have not included any user IM in the previous user model specified for the shake of clarity. However, it is possible to follow the trace between both models (user and interaction ones) since the classes related to the different body parts, as well as the User class itself, in the interaction model belong to the user model.

Fig. 14. Interaction model (partial view: user IMs)
7.2 Interaction Modalities of the Devices

In this section, we present the portion of the interaction model related to the different interaction modalities of the devices, both input and output ones.

7.2.1 Device Input IMs

We can distinguish up to seven different input IMs to interact with the system, depending on the available devices (touchscreen, mouse, Wii remote, Leap Motion, MS Kinect, keyboard, smartphone, etc.). Each input IM is related to a different perception, and the same device can have several input IMs to perceive up to seven different stimuli sent by the user (see Fig. 15):

- **SoundPerception**. The device is able to perceive sounds sent by the user, e.g. a microphone device.
- **TangiblePerception**. The device is able to perceive/interpret/read a specific type of gesture that the user performs with a tangible object, e.g. a gesture made with a tag object on a Microsoft PixelSense device.
- **PositionPerception**. The device is able to perceive the concrete position of the user’s body or some parts of it, relative to a particular space, e.g. a MS Kinect device can capture the position of the whole body, recognizing whether the user is standing up or not, among other positions. MS Kinect can also capture the position of a single part of the body, e.g. the arm, as a Leap Motion device does with the hand and its different fingers.
- **MotionPerception**. The device is able to perceive specific movements performed by the user’s body, e.g. the movement that involves rising both arms at the same time can be recognized by a MS Kinect device, or specific fingers/hand moves, such as fisting the hand, can be recognized by a Leap Motion device.
- **PressPerception**. The device is able to perceive a touch gesture that implies a certain level of pressure to be performed on, e.g. a button or a key, such as in a keyboard device.
- **TouchPerception**. The device is able to perceive a touch gesture, but where the press made by the user’s finger is not relevant (or lighter than in the previous case) to carry out the gesture, since his is not touching a button or a keyboard key, e.g. a rotation or scroll move on a touchscreen.
- **BiosignalPerception**. The device is able to perceive concrete biosignals emitted by the user, e.g. the heart rate or the corporal temperature.

It is interesting to highlight that we have considered these seven input IMs above others given their relevance in the domain of tele-rehabilitation for treating people with ABI and older people. It is very common that these subjects suffer from several mobility problems, even at low-level granularity, e.g. when moving fingers or making some pressure gestures. For this reason, we have distinguished, e.g. between PressPerception and TouchPerception, as a patient can have the strength to make a light touch on a tangible surface, but not enough to press a key over a keyboard. Thus, the tele-rehabilitation system should be prepared to overcome such situations and adapt the input IM to the user needs and capabilities. We have included as well PositionPerception and MotionPerception, among others, to provide the system with suitable input IMs for tracking physical rehabilitation activities.

7.2.2 Device Output IMs

A device can present up to five different output IMs, so that it can interact with the user. In this sense, the user is able to perceive stimuli sent by the system through his five senses. These five device communication channels are presented below (see Fig. 15):

- **Graphic**. The device is able to send visual stimuli, perceived by the user by means of his eyes, e.g. a text or a picture presented in the screen of a smartphone device.
- **Acoustic**. The device is able to send acoustic stimuli, perceived by the user by means of his ears, e.g. a speaker is able to emit sounds to the user.
- **Olfactory**. The device is able to send olfactory stimuli, perceived by the user by means of his nose, e.g. a smell launched by a specific device.
- **Gustatory**. The device is able to send gustatory stimuli, perceived by the user by means of his mouth, e.g. a flavor provided by a specific device.
- **Haptic**. The device is able to send haptic stimuli, perceived by some parts of the user’s body, e.g. a glove device with haptic capabilities that can send vibrational signals to the user’s hand.
As seen in Fig. 15, a Device supports, at least, one I/O (Input/Output) capacity which makes it act as a Sensor (with one or more DeviceInputIMs), or as an Actuator (with one or more DeviceOutputIMs).

7.3 Interaction between the User and the System

As seen in the previous sections, every device output IM has to be irredeemably connected to, at least, a user input IM, and vice versa, so that the interaction takes place. In the same way, every user output IM has to be connected to, at least, a device input IM, and vice versa. In the following sections, all these relationships are described.

7.3.1 Device Output IMs vs. User Input IMs

The Device output IMs are associated with the User input IMs in the following way (see Fig. 16):

- Graphic device output IM is connected to the Visual user input IM to establish a visual communication channel. For example, if the device shows an image (graphic output IM), the user must use his eyes to visualize it (visual input IM).
- Acoustic device output IM is connected to the Auditory user input IM to establish an auditory communication channel. For instance, if the device plays spoken instructions (acoustic output IM), the user must use his ears to hear them (auditory input IM).
• **Olfactory** device output IM is connected to the **Olfactory** user input IM to establish an olfactory communication channel. For example, if the device launches a smell (olfactory output IM), the user must use his nose to perceive it (olfactory input IM).

• **Gustatory** device output IM is connected to the **Gustatory** user input IM to establish a gustatory communication channel. For example, if the device provides a flavor (gustatory output IM), the user must use his mouth to taste it (gustatory input IM).

• **Haptic** device output IM is connected to the **Haptic** user input IM to establish a tactile communication channel. For example, if the device provides a vibration to the right hand of the user (haptic output IM), the user must use his right hand to perceive it (haptic input IM).

Thus, whether a part of the user’s body is able to perceive some type of stimulus from the system, then it should have a certain degree of sensitivity so that the interaction occurs. In other words, if the user wants to be able to perceive a particular type of stimulus from the system (device output IM), then he should present the corresponding input IM so that the interaction takes place.

### 7.3.2 User Output IMs vs. Device Input IMs

With regard to the **User output IMs**, the connections with the **Device input IMs** are presented below (see Fig. 16):

• **Speech** user output IM is connected to the **SoundPerception** device input IM, so the system should include a device/s able to perceive sounds in order to interact with the user.

• **GestureWithTangible** user output IM is connected to the **TangiblePerception** device input IM, so the system should include a device/s able to interpret/perceive/read tangible gestures.

• **BodyGesture** user output IM is connected to **PositionPerception** and **MotionPerception** device input IMs, so the system should include a device/s able to interpret/perceive certain positions and/or movements of the user’s body.

• **TouchGesture** user output IM is connected to **PressPerception** and **TouchPerception** device input IM, so the system should include a device/s able to perceive pressure (where the gesture is not as relevant as the press performed, e.g. when pressing a key or button), and/or touch gestures (where the pressure is not as relevant as the type of movement performed, e.g. when making zoom with two fingers).

• **Biosignal** user output IM is connected to the **BiosignalPerception** device input IM, so the system should include a device/s able to perceive certain user’s biosignals, such as the heart rate or balance.

Therefore, whether the system wants to be able to perceive a particular type of stimulus from the user (user output IM), then it should include a device with the corresponding input IM so that the interaction takes place.

### 8 Conclusions and Future Work

As seen in this work, variability in the domain of rehabilitation should be seriously considered, especially when dealing with older people and people with ABI. Namely, this last collective has demonstrated to be highly assorted due to the different types of deficits the person affected may present, as well as the different devices that he can use for the rehabilitation, depending on his deficits. There are also a wide variety of rehabilitation activities that those people with ABI may carry out. Specially, we have described different activity patterns for cognitive rehabilitation. With the aim of modelling this variability, we have proposed a feature model able to provide the association activity (regarding cognitive rehabilitation) that best fits the special needs and characteristics of the patient with ABI. Hence, this feature model takes into account all possible ABI deficits, the different ABI grades, and the devices available for the rehabilitation therapy. Moreover, it also considers the possible constraints for using certain devices, or regarding the ABI grade the patient may present, or regarding the association activity that the system will finally offer. All of these restrictions depend directly on the ABI deficits of the user.

In addition, other two different models have been proposed in order to provide a useful scenario for modelling the context of use in ABI rehabilitation. On one hand, a user model has been presented for modelling the user characteristics and preferences while performing rehabilitation tasks. It also enables the specification of the ABI deficits of the user, as well as his body parts and movements associated. We can indicate the level of mobility and sensitivity degree regarding each body part which will make the adaptation easier.
Fig. 16. Interaction model proposed
On the other hand, an interaction model has been also proposed for modelling the different interaction modalities, both input and output ones, of the user and the available devices. Furthermore, this model describes how the interaction of the user with the system should be, depending on the different interaction modalities (i.e. communication channels) currently available.

To conclude, in this work, we have focused our effort on building models to be exploited for adapting tele-rehabilitation systems to older people and people with ABI, taking into account such context and its high variability. Our future work will be oriented to manage this variability by applying e-learning mechanisms able to facilitate adaptation tasks efficiently.

Acknowledgements

This work was partially supported by the Spanish Ministry of Economy, Industry and Competitiveness, State Research Agency / European Regional Development Fund under the projects HA-SYMBIOSIS (TIN2015-72931-EXP) and Vi-SMARt (TIN2016-79100-R). It has also been funded by the Spanish Ministry of Education, Culture and Sport thanks to the FPU scholarship (FPU12/04962).

References


Chapter 14

Discussion and Future Work

This chapter presents the discussion and conclusions regarding this dissertation, as well as the ongoing and future work derived from it.

14.1 Discussion and Conclusions

As previously mentioned, among all the works presented from Chapter 3 to Chapter 13, three have been selected to endorse this dissertation by compendium of publications due to their importance:


As observed, the main aspiration of this thesis has been to provide best practices and useful guidelines for assisting designers, stakeholders and practitioners in the task of adapting a system at both architectural and HCI level. It has been proposed that the adaptation activities aimed at adapting the system at both levels are driven by the surrounding context of use, which has been accurately specified. For this aim, some metamodels have been devised and validated by using different case studies. Moreover, different proposals about the architecture of context-aware systems have been also analyzed to align the adaptation at architectural and HCI levels. All this work has been performed to answer the four RQs previously established (Section 1.2).
The first research question (RQ 1) was related to the alignment of adaptation at architectural and HCI level. Thus, an overview of SAs for context-aware systems [55] (Chapter 11) was presented by means of a systematic mapping study. This study achieved several goals: i) to review the different software architecture concepts proposed for designing and building context-aware systems in order to understand the relationship between them; ii) to review how those software architecture concepts take into account the context-based adaptations; iii) to check the maturity of the existing methods for assessing context-based adaptations; iv) and last, but not least important, to check the maturity of those software architecture concepts. This mapping study proved that there was a real need to analyze which research works highlighted the use of context and its relationship with the SA in existing context-aware systems. We noticed the effort invested in improving the use and management of context in context-aware systems and, especially, within their architecture. Moreover, we also identified the gaps existing up to now regarding the deficiencies and weaknesses of systems when dealing with context and its different dimensions.

Therefore, researchers and practitioners can make use of the results of this mapping study to improve their research. They can follow the guidelines proposed for dealing with context and its different dimensions, as well as for knowing how to consider such context in the SA. By doing so, they will provide a good adaptation experience to the final user by developing context-aware systems and adaptations with enough quality.

The second research question (RQ 2) was related to evolution at architectural level driven by AK. The main conclusions here are that AK can be considered itself as an evolution driver, in the sense that it provides much information of particular relevance to the evolution process; and that much of the evolution process itself can be captured as part of the AK, using evolution concepts at the architectural level. To simplify the reuse of this knowledge, a specific structure (i.e. evolution style) was proposed for capturing the information and decisions related to architectural evolution [12] (Chapter 3). Moreover, sometimes this AK is presented in an inadequate way that hinders the architects’ task of evolving the system. Thus, five 2D visualization techniques to support AK visualization were described and assessed by means of an empirical evaluation of the quality factor usability for discovering the most effective one. It was proven that Node-link and tree technique was one of the most usable due to its simplicity and clarity in visualizing AK. However, the experiment also showed that it should be used in combination with others, such as Indented List and/or Zoomable. In this sense, the results of this experiment concluded that tools for exploiting AK should be designed to provide software architects with different visualization techniques by considering the main AK management task (modification, navigation or overall inspection) to be carried out by software architects [48][49][50] (Chapter 12).

Additionally, several shortcomings were identified that can prevent organizations from exploiting AK, such as the variety of formats for documenting and representing it, or the difficulties in meeting its requirements for evolution. Linked Data (LD) techniques were suggested to solve these shortcomings, so we recommended storing the network of decisions using the Resource Description Framework (RDF) to be efficiently retrieved by means of SPARQL queries (i.e. an RDF query language). Several tools for managing LD were described and compared with the purpose of deciding to which extent they provided relevant features regarding AK, and which ones were the best/worst for sharing and reusing AK as LD. Given the outcomes, we concluded that the best choice was to use a LD tool with some SDK support, such as Sesame or Apache Jena & Fuseki, to build some specific tool or interface which simplified the use of this tool in an AK context [46][47] (Chapter 4).
The third research question (RQ 3) was related to offer a certain degree of quality to the adaptation process. With this aim, the concept QoA was refined to better understand what this term encompasses when it was applied to the different stages of an adaptation process. Throughout the adaptation process, how QoA could be considered was also presented. By doing so, it can be proposed what techniques or metrics best fit each adaptation process stage, contributing to a better understanding about what QoA is and how it can be used in a full adaptation process. Moreover, as future work, we aim at providing a complete adaptation quality model that will guide any adaptation designer thoroughly during the design process [42][44] (Chapter 5). Following this line, a gap between the definition of Expected quality and Wished quality was detected, studied, and analyzed by using a survey. It was identified the different terms that persons either related or not to the interaction field used to describe quality. Finally, it was shown that there was a gap between the views that both groups expressed about the notion of interaction quality. This gap should be considered in the understanding of quality to provide meaningful information about QoA at every stage of an adaptation process [43][45][41] (Chapter 6).

Finally, the fourth research question (RQ 4) was related to the context handling in a high variability application domain as tele-rehabilitation. With this aim, it was presented how the design of therapies was tackled, as well as a set of editors for both therapies and fuzzy rules to control the rehabilitation process. To execute the therapies designed, a runtime environment for the execution of therapies was also proposed relying on the multi-agent system approach. By doing so, a complete system to create, perform and monitor therapies for physical rehabilitation of people was developed. Hence, a tool was provided for being used by experts in the field of rehabilitation to define a set of customized therapies and the rules that determine the behavior of the system at runtime. In this way, activities could be adapted to people with ABI, and especially older people, while performing a particular therapy. The creation of therapies was driven by a metamodel which followed a concrete schema: a Therapy was composed of Activities, an Activity was composed of Tasks, and a Task was a set of Steps, which could be either Gestures or Postures that the older person had to perform.

Relationships among elements of the same hierarchical level could be established in order to define a sequence of Therapies, Activities or Tasks, respectively. Therefore, the therapy model could be described as a composite state diagram. The therapist could easily instantiate the meta-model by using a user interface developed for Microsoft PixelSense and specify the movements that the person had to perform by using Microsoft Kinect. The resulting model could be interpreted by the system, which automatically generated therapies that were performed by people with motor impairment problems. In this sense, our system could be adapted according to the patients' needs. These therapies were also supervised and monitored by the system using MS Kinect and other kinds of sensors that provided information about the physical conditions of the people in a transparent way. Furthermore, in the context of rehabilitation, a patient only needed to make the movements according to the specified therapy without direct manipulation of any device, taking advantage of the AmI paradigm. Moreover, the therapist may define a set of FISs that enabled our system to adapt the therapies and to decide the performance order of the tasks, activities and other parameters of the therapies at runtime. A context metamodel was also presented for modelling those patients who were expected to use our therapy system, as well the available devices and any type of ambient information relevant for the adaptation process. Many works performed during the whole process of this thesis were looking for these outcomes, such as [56] (Chapter 7), [52][54] (Chapter 8), [58][59] (Chapter 9), and [53] (Chapter 10).

More specifically, a user model was also proposed for modelling the body parts, the different movements every part can perform, as well as some other features related to ABI, e.g. the ABI
group or the different deficits the patient could present at physical, cognitive or emotional level. Apart from this user model, an interaction model was proposed too. It facilitated the specification of Interaction Modalities (IMs) the user and the available devices of the system could present. A feature model was presented as well for creating adequate and bespoke cognitive activities for the rehabilitation of people with ABI. This proposal aims at controlling the high context variability usual in people with ABI while performing rehabilitation tasks. According to the characteristics of the patient, the system will be able to provide a configuration of features, including the most appropriate association activity for handling cognitive deficiencies in people with ABI. All these metamodels and models presented in [51] (Chapter 13) will facilitate adaptation at HCI level following a model-based approach. Therefore, appealing and encouraging future lines are derived from this work.

With regard to the venues of the articles written during this thesis, 3 articles have been published in JCR journals (2 Q2 and 1 Q1), and 3 additional articles are currently under review or will be in the near future. Moreover, 10 articles have been published in specialized international conferences, including 2 CORE A venues. 4 articles have been published in national venues (1 journal and 3 conferences). Finally, 2 additional works have been published as technical reports. Up to now, 11 articles have been indexed by DBLP (http://dblp.uni-trier.de/pers/hd/r/Roda:Cristina), namely 3 JCR journals and 8 conference papers.

However, despite of the results reached during this thesis, it is obvious that there are some limitations which will guide our future work. These limitations and future proposals will be outlined in the next section.

### 14.2 Ongoing and Future Work

As aforementioned, the main purpose of this thesis has been to provide best practices and useful guidelines for assisting in the task of adapting a system at both architectural and HCI level. However, there is still work to be done for improving such adaptations at architectural and HCI level, as well as for attempting to cover the different gaps encountered in SAs for context-aware systems, as seen in Chapter 11.

For the aim of enhancing evolution at architectural level, some tasks can be carried out. For example, the full integration of the generic proposal that uses AK as an evolution driver (Chapter 3) into a specific approach, such as ATRIUM [36], that defines and manages SAs from the requirements phase. This incorporation of evolution styles into ATRIUM will be just a matter of extending the AK network to include the new kinds of relationships (those related to evolution) and to provide an additional cycle in its model-driven development process. Further work is related to the evaluation of AK-driven evolution styles from an empirical point of view. In addition, in order to answer the RQ 2 completely, we want to prove whether AK can be also considered as a driver for the dynamic evolution of SA, i.e. when adapting the system at runtime, so necessary when dealing with context-aware systems. Regarding the representation of AK (Chapter 4 and Chapter 12), future work will focus on 3D visualization techniques for capturing AK and will try to determine which technique is the most appropriate for each type of AK management activity. It is also interesting to investigate what visualization techniques would facilitate a more effective collaborative work between software architects, as well as to confirm the present results with further experiments including practicing architects where possible.

For the purpose of improving HCI adaptation tasks, some works can be performed as well. For instance, regarding the work presented in Chapter 5 and Chapter 6, future research should examine
a wider range of people, both knowledgeable and not knowledgeable in interaction use, to describe quality. It is shown that there is a gap between the views that both groups express about the notion of interaction quality. Therefore, the wider range of people giving their opinion about quality, the better understanding of quality to provide meaningful information about QoA. It is also interesting to analyze the design features that make certain kinds of adaptations more accepted than others, i.e. those adaptations with better QoA.

With respect to the interactive FIS that provides bespoke therapies to older people (Chapter 9), future plans will be oriented to carry out new evaluations, involving more therapists and patients, to validate the system in a clinical environment. In this sense, although the system design has been user-centred, new evaluations will be designed for involving both therapists and patients during a clinical study. One of these evaluations will focus on the usability of the systems and will provide us with useful insights about several aspects, such as to what extent therapists need to be trained to use the system, or to detect any potential issues that might arise while using the system. It would also be interesting to perform evaluations of the system with different types of users to establish a statistical model for the automatic assessment of certain variables. This could lead to better fuzzy rules, for instance, relating fatigue and time-out between Steps, Tasks or Activities. Moreover, these evaluations will enable us to draw more in-depth conclusions about the usefulness of this technology.

Currently, no learning mechanism is supported in the MAS proposed for ABI tele-rehabilitation (Chapter 10), and it has not been validated in a real environment. Thus, future work will be focused on the evaluation of such system in a real set up, relying on the collaboration of some associations that assist older people and people with ABI [1][11]. Besides, another challenge to be addressed in the near future is the integration of other AI techniques, beyond FISs, such as Neural Networks, to support learning from therapy executions in order to improve the adaptation process.

Finally, the mapping study carried out about SAs for context-aware systems (Chapter 11) has revealed many gaps regarding the deficiencies and weaknesses of those systems. In this sense, these gaps ought to be covered in the near future, and the guidelines proposed should be taken into account for dealing with context and its different dimensions, as well as for knowing how the SA may consider such context. By doing so, we will be able to develop our own adaptation framework based on model-driven techniques that will exploit the different proposals, models and metamodels presented in this thesis.

With regard to the ongoing work, now we are focused on managing variability in the context of ABI domain and tele-rehabilitation, by exploiting our user model, interaction model and feature model previously defined (Chapter 13). For this purpose, we are considering several approaches, such as genetic algorithms and ant colony optimization.
References


References


