PUC

ISSN 0103-9741

Monografias em Ciência da Computação
n° 08/10

# Experiences with Model Reuse: Non-Functional Requirements Catalogues for Ubiquitous Systems

**Milene Serrano**

**Carlos José Pereira de Lucena**

**John Mylopoulos**

**Eric Yu**

Departamento de Informática

**PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO**

**RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22451-900**

**RIO DE JANEIRO - BRASIL**

Monografias em Ciência da Computação, No. 08/10 ISSN: 0103-9741
Editor: Prof. Carlos José Pereira de Lucena May, 2010

# Experiences with Model Reuse: Non-Functional Requirements Catalogues for Ubiquitous Systems*

Milene Serrano[1], Carlos José Pereira de Lucena[1], John Mylopoulos[2], and Eric Yu[2]

[1]Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro
[2]Department of Computer Science, University of Toronto, Toronto, ON, Canada

milene@les.inf.puc-rio.br, lucena@inf.puc-rio.br, jm@cs.toronto.edu, and eric@cs.toronto.edu

**Abstract.** Requirements engineering activities demand adequate conceptual modeling support to be effectively and efficiently performed, especially in complex domains such as ubiquitous computing. Ubiquitous systems are embedded in ever-changing environments, in which the device heterogeneity, user mobility, content server distribution, adaptability, context awareness and other issues are intrinsic. Unfortunately, there is a lack of conceptual and tool support that guides developers from requirements to code while respecting the non-functional requirements that arise from the special nature of ubiquitous systems. In this paper, we present a catalogue of non-functional requirements for ubiquitous systems and a method for using it to guide systematic system development. We report on experiences in the construction of this catalogue, drawing on state-of-the-art literature as well as our own experimental research. The catalogue is constructed using the NFR Framework. Operationalizations of the NFRs are illustrated through the use of multi-agent-systems, goal orientation and distributed intentionality.

**Keywords**: Ubiquitous Computing, Systematic Development, NFRs Catalogues, Intentional MAS, and Goal-Oriented Requirements Engineering.

**Resumo**. As atividades de Engenharia de Requisitos demandam uma modelagem conceitual adequada para que as mesmas sejam efetivamente e eficientemente desempenhadas, especialmente em domínios complexos como o da computação ubíqua. Sistemas ubíquos estão inseridos em ambientes marcados pelas contantes mudanças, nos quais a diversidade de dispositivos, a mobilidade do usuário, a distribuição dos servidores de conteúdo, a adaptabilidade, a sensibilidade ao contexto e outras características são intrínsicas. Infelizmente, existem poucos suportes conceituais e ferramentas que guiem os desenvolvedores dos requisitos ao código, enquanto respeitando os requisitos não-funcionais originários da própia natureza dos sistemas ubíquos. Nesse artigo, apresentamos um catálogo de requisitos não-funcionais para sistemas ubíquos e um método para guiar o desenvolvimento systemático dos mesmos. Reportamos nossas experiências na construção do catálogo proposto, apresentando o estado-da-arte bem como nossa pesquisa experimental. O catálogo é construido usando o *NFR Framework*. Operacionalizações dos requisitos não-funcionais são ilustradas através do uso de sistemas multi-agentes, orientação à meta e intencionalidade distribuída.

**Palavras-chave**: Selecione até cinco palavras-chave que definam precisamente o conteúdo do seu trabalho.

---

* This work has been sponsored by CAPES and CNPq, Brazil.

**In charge of publications**:

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC-Rio Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22451-900 Rio de Janeiro RJ Brasil
Tel. +55 21 3114-1516 Fax: +55 21 3114-1530
E-mail: bib-di@inf.puc-rio.br

# Table of Contents

## 1 Introduction

Ubiquitous Computing is a post-desktop model of computing where multiple, heterogeneous hardware devices are thoroughly integrated into everyday objects and activities. In this new computing paradigm, the user is engaging in everyday activities – such as walking, working, or listening to music – while "using" ubiquitous systems (ubisystems), and may even be unaware that he is actually doing so.

As expected, ubisystems development poses new challenges for software engineers because of the special nature of the non-functional requirements (NFRs) that apply to such systems. Weiser [1] offers a thorough account of these NFRs: (i) **S**atisfaction, ubisystems must guarantee high levels of user satisfaction; (ii) **A**daptability, device- and context-aware adaptations are necessary to ensure that a ubisystem functions in multiple contexts; (iii) **H**eterogeneity, involving different devices, platforms, networks, spaces and people, is intrinsic and needs to be accommodated; (iv) **M**obility, users can be anywhere and everywhere, performing their daily activities while using a ubisystem; (v) **O**mnipresence of ubisystem, they are available wherever the users are; and (vi) **A**wareness support to capture user, device, feature, and context information.

The development of ubisystems thus demands careful accommodation of a common set of NFRs. The main objective of this paper is to report on our experiences by presenting our catalogue-based technique, which consists of reusable requirements models for ubisystem NFRs, as well as a systematic process for using the catalogue during the design process. The proposed technique has been applied to a number of case studies in order to accumulate evidence that the technique actually works.

The proposed framework is centered on a repository of reusable models that aggregates the results of our efforts in capturing ubiquitous NFR issues in models to be shared in a common baseline with the Ubiquitous Computing community. The offered models can be reused by simply picking them up from this baseline and/or instantiating them to better attend to ubisystem specific needs. Our catalogue is founded on the NFR Framework [2][3] and consists of softgoal hierarchies for ubisystem NFRs. The softgoal hierarchies represent positive/negative contributions among different softgoals, as well as suggestions for their operationalizations, inspired by Multi-Agent System (MAS) and goal-oriented ideas. Although our catalogue's operationalizations are illustrated through the application of MAS, the developers can use the software hierarchies to guide their ubiquitous projects in different paradigms (e.g. object-oriented and component) as the softgoals' decompositions and the interdependencies are independent of the programming language. We have experimented with the proposed framework in the Software Engineering Laboratories at PUC-Rio and UofT for a number of years and report on our experiences. Since 2007, we have developed various ubiquitous applications in different cognitive domains (e.g. eHealth and eCommerce,) and frameworks to deal with the main ubiquitous issues (e.g. user's satisfaction, adaptability, mobility, and context awareness). We applied the proposed NFRs catalogue reusable models in the systematic development of these applications and frameworks. Thus, we performed several tests to evaluate our proposal. The tests were mainly centered on content's adequacy, user's satisfaction, system's usability, system's security, and other quality criteria of ubiquitous systems. The acquired experiences suggest that the proposed framework offers suitable support for guiding developers in answering the following questions: What are the NFR issues to be addressed for the ubisystems development process? When must these issues be considered during the development? Where can we get information on these issues? Why do we need to consider these issues? Who is impacted by them? How can we deal with them? In other words, our framework guides the development process from early requirements to implementation in dealing with ubisystem NFRs, facilitating the adaptation of our support to specific projects.

The rest of the paper is organized as follows: Section 2 introduces details about the NFR Framework. Section 3 offers an overview of the catalogue construction process and contents. Section 4 presents two case studies, while Section 5 details the application of our framework to these case studies. Section 6 presents a comparative evaluation of the framework. Section 7 reviews related work, while Section 8 concludes and suggests directions for further research.

## 2 NFR Framework Overview

The NFR Framework constitutes a Goal-Oriented Requirements Engineering (hereafter GORE) approach [4] for capturing NFRs in the domain of interest, and defining their interdependencies and operationalizations. Nowadays, the interest in this kind of approach increases in the RE community as the elaboration process of GORE approaches ends where the traditional ones (e.g. Rational Unified Process (RUP) and other object-oriented approaches) start. Thus, the NFR Framework, as well as all GORE approaches, focuses on activities that precede the requirements specification, and that are performed during the architecture design stage to drive and validate architectural decisions. We chose the NFR Framework because it allows: designing alternatives for different NFRs; dealing with conflicts, tradeoffs, and priorities; evaluating the decisions impact centered on NFRs that commonly influence the success of ubisystems; and systematically refining the models through the contributions specification for all alternatives on the NFRs. The NFR Framework provides graphs – Softgoals Interdependency Graphs (SIGs) – for NFRs modeling. The SIGs graphically represent NFRs as nodes; their refinements using AND/OR decompositions links; their positive/negative interdependencies as *some*+(*hurt*), *some*-(*help*), *some*++(*make*), *some*--(*break*) contribution links; their operationalizations as leaf nodes; and claims as annotations in natural language. Figure 1 illustrates a very simple SIG that models the Software Ubiquity, by considering its decompositions – AND links – in Software Pervasiveness, Software Mobility, and User's Satisfaction; an interdependency between Software Mobility and User's Satisfaction – Mobility[Software] positively impacts (*help*) on Satisfaction[User]; an operationalization: "Mobile Agents using special capabilities" *help* Mobility[Software]; and a claim "The Software delegates the complex device's configuration to the users" *hurt* the decomposition between Ubiquity[Software] and Satisfaction[User].

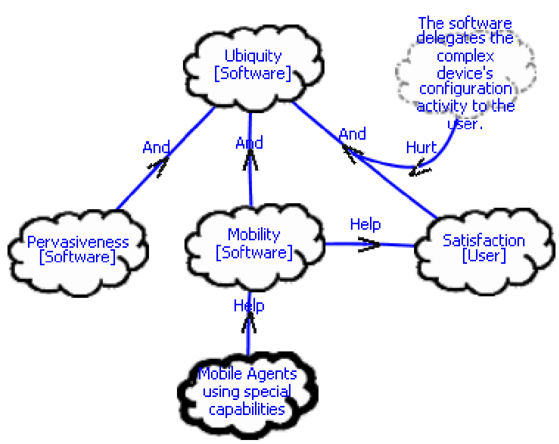



**Figure 1: SIG Notation Example**

## 3 The NFR Catalogue for Ubiquitous Computing

In order to develop the proposed catalogue by composing a reusable and conceptual NFRs models base for ubisystems, we concentrated our efforts – respecting the NFR Framework notation – on three activities: NFRs elicitation; NFRs decomposition; and NFRs Interdependencies identification. These activities started from ubiquitous scenarios and the quality criteria identification obtained in the State-Of-The-Art, experts consultation, and during our experimental research. The elicited NFRs were evaluated with the user's participation, and then evolved. These activities were iteratively performed, which allowed us to incrementally construct our knowledge base through the following phases:

**(i) State-Of-The-Art Investigation** - We started our work investigating the literature to compile an adequate initial understanding of ubiquitous concerns focusing on Ubiquitous Computing (e.g. [1][5][6]) and experimentation-oriented papers (e.g. [7][8][9].) This investigation, conducted from different viewpoints (e.g. requirements engineers and software engineers), allowed us to obtain a first version of the catalogue. This version consisted of top-level ubiquitous requirements as well as their direct refinements. Specifically, the catalogue included three top-level NFRs (Ubiquity, Pervasiveness, and Mobility), and four refinement NFRs (Content Adaptability, Context Awareness, Device Heterogeneity, Software Processes Complexity Invisibility).

**(ii) Experimental Research** - Based on the initial version, we performed our first experimental research [10] at the PUC-Rio Software Engineering Laboratory. Our main goal consisted of applying the first version of the catalogue's reusable models to the systematic development of ubisystems. We obtained some interdependencies, and operationalizations for each specified NFRs. Moreover, the research suggested some refinements for the first proposed catalogue. Among other improvements, we incorporated User Satisfaction as a seminal ubiquitous issue, and also defined other important NFRs as well as their refinements. Notable among them were Usability, Content/Service Accessibility, and Ubiquitous Profile Awareness. As a result, the evolved catalogue constituted of 21 NFRs.

**(iii) Iterative Evolution** - During the last three years, from 2007 (beginning) to 2009 (end), we performed several iterations to evolve the catalogue. Basically, the catalogue's iterative evolution involved: (a) literature investigation; (b) catalogue content exploration; (c) catalogue content identification, considering ever-changing contexts, simulated by our case studies [10][11][12][13][14][15][16][17][18]; (d) application of the catalogue on these case studies; (e) incremental refinement of the catalogue according to the newly discovered ubiquitous concerns; (f) comparative evaluation of refined and reusable models obtained from the catalogue to validate the refinements; and (g) catalogue evolution based on successful refinements. Our approach included developing tools to guide the software engineer's work in building ubisystems, by dealing systematically with key ubiquitous issues. In order to facilitate the NFRs specification and graphical representation, we centered our models on the NFR Framework notation (SIGs and Frame-Like-Notation). Basically, during this phase we iteratively created novel catalogue content, eliminated replications redundancies and ambiguous specifications. We also combined NFRs in one SIG to obtain a structured reusable model, while at other times we refined one NFR in different SIGs to improve our reusable models.

**(iv) Evolution and Maintenance** – Throughout the process, collaborators could submit new SIGs and review the catalogue. This phase includes novel experimental research to incrementally refine the actual reusable models version, considering the use of our reusable models in different Ubiquitous Computing groups' projects.

As the result of this process, the catalogue's last version is composed of almost 700 interdependent softgoals. We organized them according to their prioritizations in ubi-systems, which were obtained in our experimental research throughout the last three years. The main softgoals - the most commonly found in the ubisystem development process as well as the most generic ones - received highest priority. The catalogue is actually organized into four main softgoals (Ubiquity, Pervasiveness, Mobility, and User Satisfaction) at the top level. Moreover, there are 17 softgoals in the second level, including: Content Adaptability, Context Awareness, Device Heterogeneity, Transparency, and Process Complexity Invisibility. Furthermore, there are almost 200 NFRs at the third level, such as: Self-Regulation, Autonomy, Reactivity, and Controllability. This categorization - driven by the capturing of ubiquitous NFRs issues in several different ubisystems - was applied to the entire catalogue, improving its applicability. It is important to notice that as the catalogue is in constant evolution, the refinements involve refactoring in the prioritizations and, consequently, they reflect on the catalogue's organization. In this research line, we are also proposing a further way-of-working to organize our catalogue based on different criteria such as: most used NFRs or ones that address greater number of issues receive higher priorities. Table 1 summarizes the main ubiquitous NFRs issues addressed by our catalogue of reusable models.

**Table 1: Summary of the main ubiquitous NFRs issues addressed by our reusable-models**

| NFR | Meaning | First Identification | Priority | Top Down View Category |
|---|---|---|---|---|
| Software Ubiquity | ... | Phase 1 | Extremely High | 1 |
| Software Pervasiveness | ... | Phase 1 | Extremely High | 1 |
| Software Mobility | ... | Phase 1 | Extremely High | 1 |
| Content Adaptability | ... | Phase 1 | Very High | 2 |
| Context Awareness | ... | Phase 1 | Very High | 2 |
| Device Heterogeneity | ... | Phase 1 | Very High | 2 |
| Software Process Complexity Invisibility | ... | Phase 1 | Very High | 2 |
| Software Distribution | ... | Phase 1 | Very High | 2 |
| User's Satisfaction | ... | Phase 2 | Extremely High | 1 |
| Software Usability | ... | Phase 2 | High | 3 |
| Content/Service Accessibility | ... | Phase 2 | Very High | 2 |
| Ubiquitous Profiles Awareness | ... | Phase 2 | Very High | 2 |
| User's Privacy | ... | Phase 3 – First Iteration | High | 3 |
| Software Traceability | ... | Phase 3 – First Iteration | Very High | 2 |
| Software Recoverability | ... | Phase 3 – First Iteration | Very High | 2 |
| Software Portability | ... | Phase 3 – First Iteration | Very High | 2 |
| Software Self-Regulation | ... | Phase 3 – Second Iteration | High | 3 |
| Software Autonomy | ... | Phase 3 – Second Iteration | High | 3 |
| Software Flexibility | ... | Phase 3 – Second Iteration | High | 3 |
| Software Reactivity | ... | Phase 3 – Second Iteration | High | 3 |
| ... | ... | ... | ... | ... |
| Software Accuracy | ... | Phase 3 – Last Iteration | High | 3 |
| Software Controllability | ... | Phase 3 – Last Iteration | High | 3 |
| Software Transparency | ... | Phase 3 – Last Iteration | Very High | 2 |
| New One | ... | Phase 4 | ... | ... |

Due to the huge number of NFRs and reusable models shared in our baseline, we also developed a Web-application to facilitate their access and to help in the presentation and browsing of our catalogue's contents. This application offers an exploration tree to navigate and choose the desired NFR, the NFRs meaning, and links to their SIGs and Frame-Like Notations. In addition, according to our experimental research, the NFRs' elicitation has been a good starting point for capitalizing knowledge in Ubiquitous Computing, since they do not vary much from one ubisystem to another. This makes our reusable models as well as their decompositions, interdependencies, and operationalizations applicable to a broad class of ubisystems.

## 4 Case Studies Description

In order to report on our experiences with the proposed catalogue – centered on its application, evaluation, and evolution – we present two case studies:

- **E-Commerce Case Study (Media Shop Project)** – an intentional-MAS-oriented case study, developed at PUC-Rio to investigate ubiquitous concerns and how intentional MAS can support them. This project involved two Ph.D. students, several media servers, heterogeneous devices, and distributed users. It is a MIDP-application that provides different kinds of media contents, which are dynamically and periodically updated to compose a suitable media set for different users, allowing on-line customers' consultations and download, which could come from everywhere/anytime. The users were not necessarily experts on using software, or dealing with devices configuration. However, they wanted to know about the download, its cost, security and status. The Media Shop addressed interactions that must be performed from mobile or not and limited or not devices by respecting the user's preferences, device features and network specifications, such as: media registration, processing and download; users, devices and network profiles consulting; adaptability; appropriate media set availability; and security, download spent time and other issues analyses. Software agents supported these interactions, combining Intentional MAS and Ubiquitous Computing transversal domains. The following is a typical situation from the user's point of view: "*Find a media content that is interesting, whose price is according to my budget, … - thus, the user's interests must be considered - through the device I am using now - thus, the device heterogeneity must be considered. The download can be requested from everywhere/anytime - thus, the service omnipresence and user's mobility must be considered. I am not concerned about accessibility, connectivity, or adaptability - thus the invisibility of the software processes must be considered.*" In order to construct the system, we investigated the literature, and used some building blocks suggested by our catalogue. However, our catalogue was only in a beginning stage at that time. We had defined only some building blocks. The Media Shop and other various case studies contributed to refine them.

- **E-Health Case Study (Smart Dental Project)** – an extensive Dental case study, involving two Ph.D. students and a dental clinic's stakeholders. This clinic is an academic dental clinic, which belongs to a dental association in São Paulo State, Brazil. Its members perform social activities, attending the community; and it contributes to the dentists' academic life by training them in different dental branches. Thus, we had the dental domain and transversal ones: academic with social issues, Ubiquitous Computing, and Intentional MAS. The clinic's partners wanted to perform some activities using ubisystems, mainly desiring that the patients were able to register and schedule treatment using their hand-held devices. The clinic had special privacy policies – e.g. stored data could not be shared with other competing dental clinics. The patients also desired to protect their personal data. The stakeholders wanted to know what was going on during the performed activities. In this context, the main stakeholders were: *Patient* – user of the available dental clinic services to take care of her/his dental problem; *Dentist* – active position at the dental clinic that performs several tasks – e.g. triage process and patient's treatment; *Professor* – active position at the dental clinic that performs academic tasks – e.g. dentist's supervision and dentist's evaluation; *Attendant* – active position at the dental clinic that performs several tasks – e.g. patient's registration and registration payment; and *President*, *1º Vice-President*, *2º Vice-President*, *Secretary*, and *Bursar* as administrative positions that manage/control the dental clinic way-of-working. The following is a typical situation from the patient's point of view: "*Find a dentist adequate to my dental problem, close to my actual location, who respects my privacy policies, whose prices are according to my social conditions, … - thus, the patient's interests must be considered - through the device I am using now - thus, the device heterogeneity must be considered. The request can be placed everywhere/anytime - thus, the service omnipresence and user's mobility must be considered. I* *am a layperson in software technologies but I want to know what is going on - thus the software invisibility must be considered in complex processes but the transparency software is required to keep the user aware of what is going on.*" In order to help in the system construction, we followed our ubiquitous profile [14]. We conducted meetings with the stakeholders using brainstorming, open/close questionnaires, observations and interviews as elicitation techniques, in which we obtained different documents – e.g. dental doctor's statement, anamnesis dental form and specializations dental forms. We also described the clinic's activities through scenarios - a semi-structured language, which is close to our natural language. It simplified our interaction with the stakeholders. In a short time, the stakeholders were able to understand the scenarios, leading to their validation. We also performed activities in different disciplines (Early and Late Requirements, Architectural and Detailed Design, Implementation, and Test) following the catalogue. We do not describe them here, as it is not our focus in this paper.

Some lessons learned from these case studies and how we dealt or intend to deal with them are mainly reported on Sections 6 and 8. Only to illustrate, we can argue that our catalogue: improved the NFR elicitation efficiency; reduced elicitation time and team efforts in dealing with NFR issues; improved the quality of NFRs; and enhanced the traceability of Requirements Engineering activities. However, there were some discovered limitations, such as: the catalogue demands time to be maintained and evolve; and it needs knowledge management mechanisms.

## 5 NFR Catalogue Use Method

In this section we focus on the catalogue use method, considering two views: (i) as an SADT; and (ii) as a detailed activity-based representation. We applied this method as a building block in the systematic development of our Smart Dental Project.

- **SADT Representation:** Based on our ubiquitous profile, the catalogue can be consulted from the Early Requirements to Code. We defined some activities to apply it as shown on Figure 2 using SADT [19].

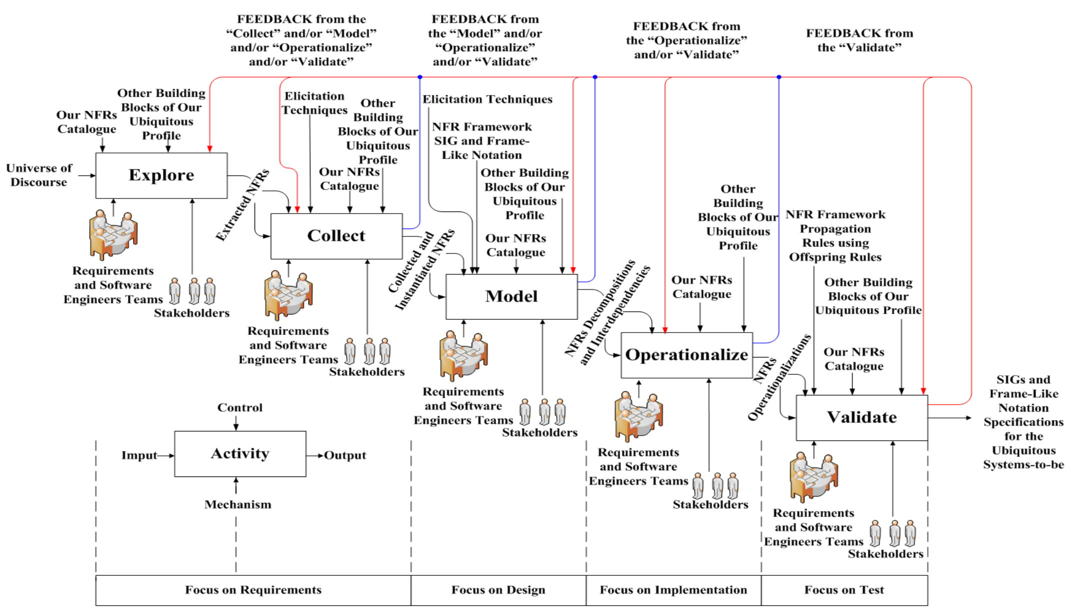



**Figure 2: NFR Catalogue Use Method in SADT Notation**

The activities' set comprises our reuse-oriented method to facilitate the use of our baseline. Among other contributions, it can: (i) assist in knowledge exploration and extraction; (ii) help in the requirements elicitation, picking them up from our baseline; and/or instantiating and evolving them; (iii) assist in the design, guiding the NFRs modeling; (iv) assist in the implementation, offering pre-defined operationalizations; and (v) help in the NFRs evaluation using propagation rules and offspring labels – denied (***D***), weakly denied (***WD***), undecided (***U***), weakly satisficed (***WS***), satisficed (***S***), and conflict (***C***) – offered by the NFRs Framework. Moreover, conflicts of interests can be solved with the stakeholders. We also use the notion of feedback proposed by Lehman [20]. The SIGs and frame-like notation can be evolved. Mistakes identified regarding further activities can demand reviews on previous ones. Basically, our catalogue focuses on: Requirements in the "Explore" and "Collect"; Design in the "Model"; Implementation in the "Operationalize"; and Test in the "Validate". However, all the specified activities are performed through the Requirements and Design Disciplines of the systematic development.

- **Detailed Activity-based Representation:** for each ubiquitous concern found in our case study – identified as NFR – we conducted the catalogue method activities. To elucidate the process performed for all softgoals elicited on our dental ubisystem, consider how we dealt with Mobility [Software] (Figure 3,) their decomposition, interdependencies and operationalizations. It is relevant to consider that Mobility means: "*the state of being in motion, using untethered technology to access data/services from occasionally-connected, portable, networked computer devices.*"

  **1. Performing the Explore Activity:** divided into Consult and Extract sub-activities. The Consult sub-activity consists of the catalogue knowledge investigation to understand ubiquitous concerns. The Extract sub-activity consists of the deduction of what knowledge is pertinent for the ubisystem. The success of this sub-activity depends on whether the Consult is satisfactorily accomplished.

  *1.1. Performing the Consult Sub-Activity:* we consulted the baseline to know about mobility in ubiquitous scenarios. This concern is decomposed on Software Distribution, Software Versatility, Software Portability, Software Traceability, Software Connectivity, Software Reliability, Software Recoverability and Software Location Awareness, which means that these eight softgoals and their sub-decomposed ones demand special attention to deal with mobility in ubiquitous contexts. Traceability [Software] is decomposed on Traceability [Device Location] and Traceability [Requests] to maintain the traces for the device location and users' requests in mobile applications. The catalogue mainly helped us with regard to knowledge capitalization and ubiquitous terms familiarization. Contributing to this field, the catalogue presents the meaning of all baseline terms and information sources for further and deeper investigations. After this exploratory searching, we were able to identify the main mobility-related issues; determine their impacts in the Mobility concern; and capitalize sufficient knowledge to put together a comprehensive view of Ubiquitous Computing's main concerns.

  *1.2. Performing the Extract Sub-Activity:* we extracted mobility-related knowledge based on our Dental case: portability (the catalogue suggests correlation between portability and devices heterogeneity, and our Dental Project involved different devices); connectivity (same correlation); traceability (to maintain the traces among requests, users, and devices in the dental clinic); and recoverability (in the stakeholders' activities the data recoverability is intrinsic).

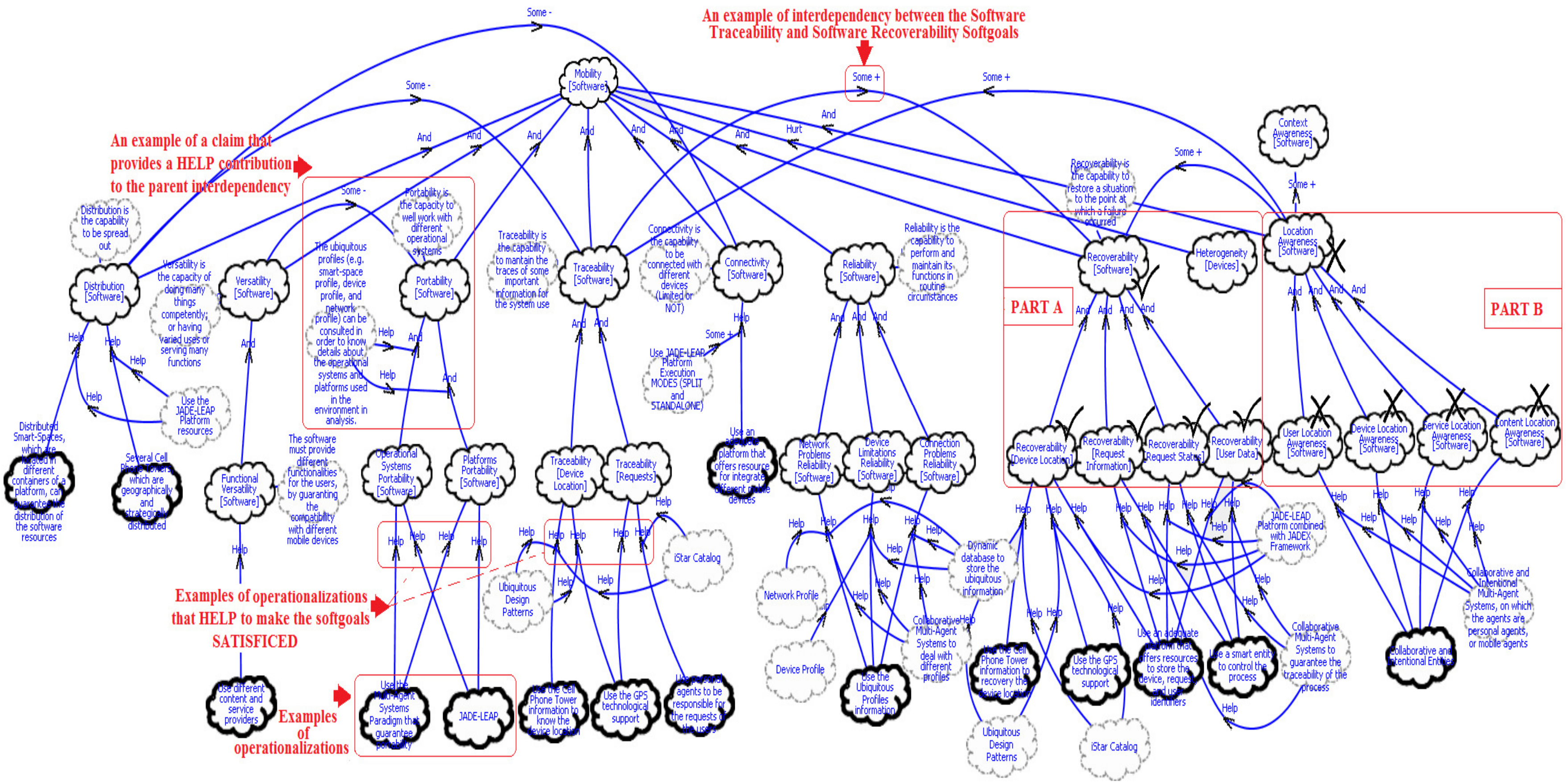



**Figure 3: Software Mobility SIG from our NFR Catalogue**

2. **Performing the Collect Activity:** composed of Pick-up and Instantiate/Evolve sub-activities. The Pick-up activity occurs if the extracted knowledge matches with the ubisystem's needs. If adjustments are necessary, the Instantiate/Evolve sub-activity is performed. Thus, the knowledge in SIG and Frame-Like Notation is instantiated and evolved

   *2.1. Performing the Pick-up Sub-Activity:* the Mobility SIG knowledge fully satisfied the concerns of our Dental Project. Thus, basically we picked up the Mobility SIG from the baseline. However, in some specific points, we evolved the SIG, instantiating it to better represent and satisfy our needs.

   *2.2. Performing the Instantiate/Evolve Sub-Activity:* we performed evolutions in the Mobility SIG – Traceability (Figure 4.) We instantiated this SIG, extending the Software Traceability decomposition, in order to maintain traces to directly associate the users' requests and the responsible agent for dealing with them.

3. **Performing the Model Activity:** based on Decompose/Determine Interdependencies in SIG; and Specify Decomposition, Claim, Correlation Rule in Frame-Like Notation – depending on the chosen notation. It is also possible to use both. The first is a graphical view whereas the second is a semi-structured specification centered on parent/offspring/contribution/constraint/condition.

   For SIG Notation:

   *3.1. Performing the Decompose Sub-Activity:* as we picked up the Mobility SIG from the baseline with few modifications, we only decomposed the Traceability concern. As shown in Figure 4, we incorporated an AND decomposition link between Traceability [Software] and Traceability [Responsible]. In our evolved SIG, the Traceability [Software] was decomposed on Traceability [Device Location], AND Traceability [Requests], AND Traceability [Responsible].

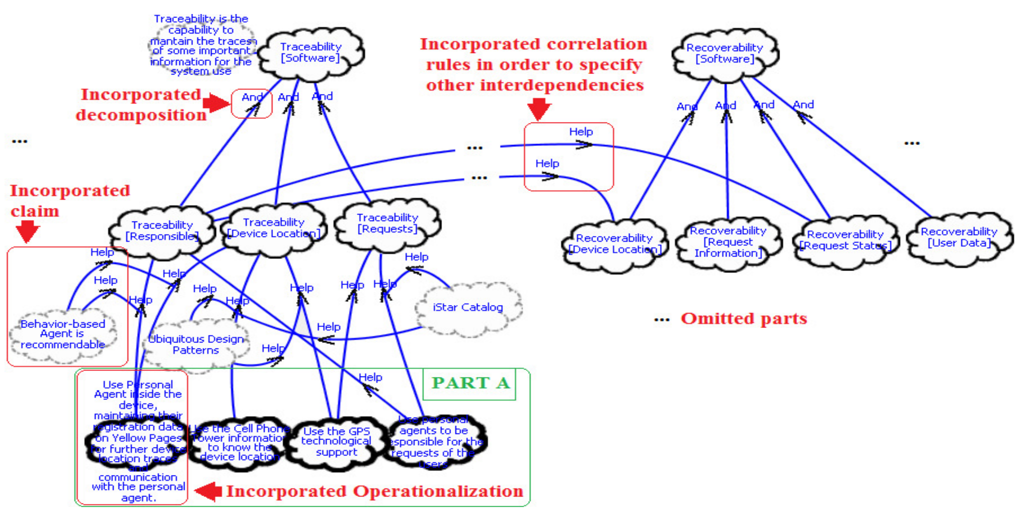



**Figure 4: Evolved Part of the Software Mobility SIG**

   *3.2. Performing the Determine_Interdependencies Sub-Activity:* Mobility SIG pre-defined some interdependencies – e.g. interdependency between Traceability [Software] and Recoverability [Software], specified as s*ome*+ contribution, which means Traceability [Software] positively contributes to Recoverability [Software]. As we instantiated this SIG, we determined interdependencies between Traceability [Responsible] and: (i) Recoverability [Device Location]; and (ii) Recoverability [Request Status] using *help* contribution links.

For Frame-Like-Notation:

*- For Traceability[Software] Decomposition*

*3.1. Performing Specify Parent Sub-Activity:* **parent:** Traceability [Software]

*3.2. Performing Specify Offspring Sub-Activity:*

**offspring:** Traceability [Device Location]; Traceability [Requests]; Traceability [Responsible]

*3.3. Performing Specify Contribution Sub-Activity:* **contribution:** AND

*- For Claim in the Traceability[Software] SIG*

*3.1. Performing Specify Parent Sub-Activity:*

**parent:** Traceability [Device Location] AND Traceability [Requests] AND Traceability [Responsible] SATISFICE Traceability [Software]

*3.2. Performing Specify Offspring Sub-Activity:* **offspring:** Claim[argument]

*3.3. Performing Specify Contribution Sub-Activity:* **contribution:** HELP

*3.4. Performing Specify Constraint Sub-Activity*

**constraint:** /*argument is specific about the traceability concern*/

*- For Correlation Rule between Recoverability and Traceability*

*3.1. Performing Specify Parent Sub-Activity:* **parent:** Recoverability[Request Status]

*3.2. Performing Specify Offspring Sub-Activity:* **offspring:** Traceability[Responsible]

*3.3. Performing Specify Contribution Sub-Activity:* **contribution:** HELP

*3.4. Performing Specify Condition Sub-Activity:* **condition:** true

4. **Performing the Operationalize Activity:** to define an adequate operationalizations' set to further implementations. Our catalogue already offers some operationalizations to be reused and attend needs faster. It is also possible to establish new support, using the developers' expertise. For our Dental Project, we normally used the offered support set. However, sometimes the offered support did not address our needs – e.g. the GPS use suggested to deal with the device traceability. We preferred to use a personal agent inside the device. This agent was registered on an MAS platform (Yellow Pages [21]). If we wanted to trace the device, we simply consulted the Yellow Pages, and recovered the agent identification to establish contact.

   *- With the NFR Catalogue Operationalizations Support*

   4.1. *Performing the Specify_Operationalizations_Based_On_NFRsCatalogue Sub-Activity:* we simply selected and picked up operationalizations from the baseline.

   *- Without the NFR Catalogue Operationalizations Support*

   4.1. *Performing the Specify_Operationalizations_NOT_Based_On_NFRsCatalogue Sub-Activity:* we specified an operationalization for the Traceability [Responsible] softgoal: "*Use Personal Agent inside the device, maintaining their registration data on Yellow Pages for further device location traces and communication with the personal agent.*" We also specified the contribution of it for the Traceability [Responsible] as a *help* contribution (Figure 4 - Part A).

5. **Performing Validate Activity:** divided into Evaluate and Solve Conflicts sub-activities. The Evaluate sub-activity checks interdependencies using correlation

rules and stakeholders' meetings. The Solve-Conflicts sub-activity deals with conflicts and open states (Figure 3 – Parts A and B).

*5.1. Performing the Evaluate Sub-Activity:* we scheduled meetings with the clinic's stakeholders to evaluate the contributions specified on the evolved Mobility SIG independencies. As suggested in the catalogue method, we explained the SIG and Frame-like notations for them to facilitate our interaction. During the evaluation, using propagation rules, some conflicts were identified: *the dentists concluded that the heterogeneity was not a concern, as they proposed the same mobile device for all the dental clinic's dentists.* However, the attendant mentioned she used a desktop in her tasks, and some patients said it would be a good idea to perform registration/ treatment scheduling using their own devices.

*5.2. Performing the Solve_Conflicts Sub-Activity:* we decided to use different devices to improve the patients' and attendant's level of satisfaction. We also standardized the devices for the dentists, establishing smartphones with large memory and processing capacities. This decision contributed to the dentists' satisfaction and the content adaptability exchanged between the dentists' devices.

Furthermore, the catalogue method contemplates the feedback notion, allowing refinements when misconception/misunderstanding occurs from faulty judgment, deficient knowledge or lack of forethought. In our case study, we constantly returned to previous activities to review details. Finally, we obtained the SIGs and Frame-like notation. Figure 5 summarizes the entire process performed in the Dental case study.

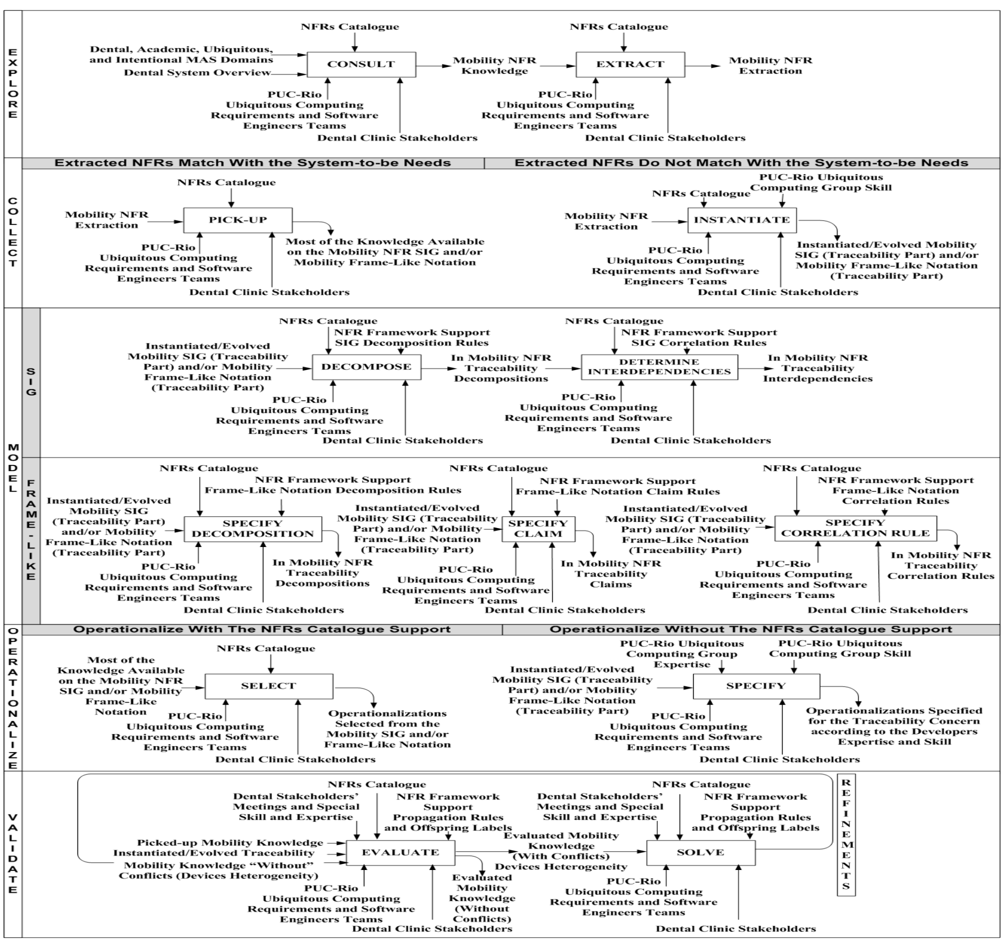
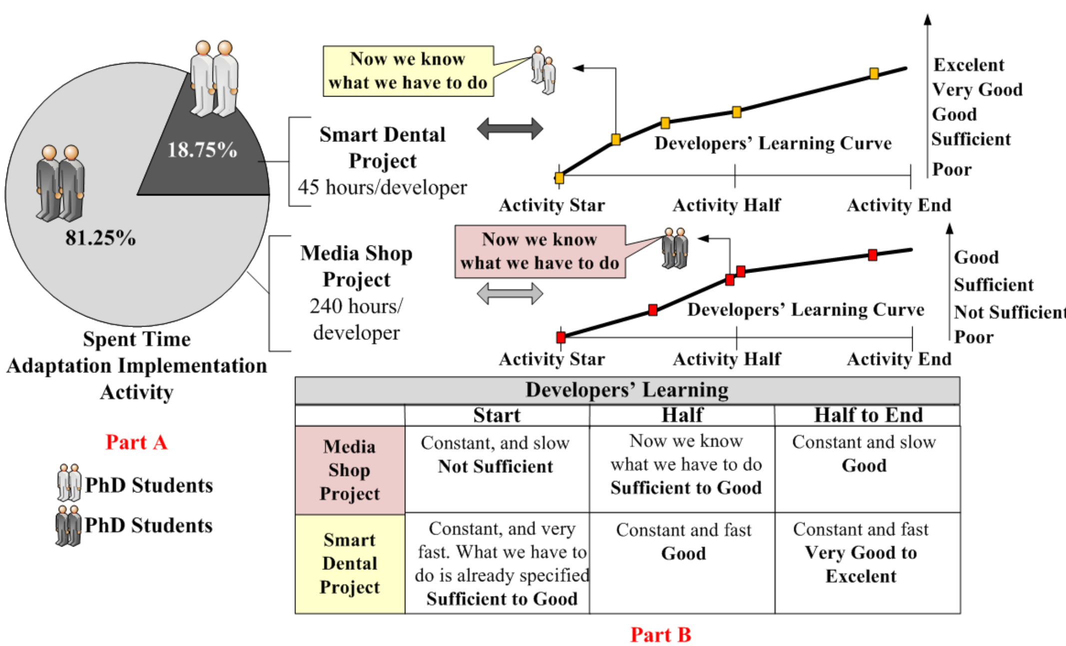



**Figure 5:** NFRs Catalogue Use Method instantiated by our Dental Case Study

## 6 A Comparative Evaluation of the NFR Catalogue

The catalogue operationalizations guided the dental ubisystem implementation. Following them, we used the JADE-LEAP [22] to integrate the platform and powerful PJava devices using the *standalone* execution mode; and the platform and limited MIDP devices using the *split* execution mode. Another example was in the content adaptability using an intentional-MAS-oriented framework (IFCAUC [12][14],) suggested on the adaptability softgoal operationalizations. We developed this framework as an API to be offered as a building block to support adaptability.

Moreover, the catalogue operationalizations reduced the spent time and efforts to deal with some ubiquitous issues. For the adaptability issue, we basically incorporated the API in our ubisystem project; extended it to adequately address our needs; and the adaptability involving distributed smart-spaces, content providers, and heterogeneous devices was implemented. We compared the spent time with content adaptation in our Media Shop case study – developed following the catalogue first version (without IFCAUC) – and the spent time in the Dental case study – developed following the catalogue last version (with IFCAUC). Moreover, the Media Shop involved less content providers and requests than the Dental case. Figure 6 (Part A) graphically presents the results. The adaptability concern in Media Shop took almost four months to be performed, considering two dedicated developers working six hours/day and five days/week (proximally 480 hours). It took through weeks in the Dental Project, considering two dedicated developers working six hours/day and five days/week (proximally 90 hours). We also evaluated what we learned throughout the adaptation implementation activity, considering our knowledge in the beginning (Start); middle (Half); and from middle to end (Half to End) of the process. Furthermore, we determined the exact moment we became aware of what we had to do in each project. We graphically linked these punctual measurements (Figure 6 – Part B) to proximally obtain the learning curves.




| Developers' Learning | | | |
|---|---|---|---|
| | Start | Half | Half to End |
| Media Shop Project | Constant, and slow **Not Sufficient** | Now we know what we have to do **Sufficient to Good** | Constant and slow **Good** |
| Smart Dental Project | Constant, and very fast. What we have to do is already specified **Sufficient to Good** | Constant and fast **Good** | Constant and fast **Very Good to Excelent** |

**Figure 6: Comparing Spent Time and Effort for Content Adaptation - Part A and B**

Moreover, we analyzed the users' satisfaction in both projects to obtain the stakeholders' feedback regarding the systems' usability and content adequacy (Figure 7.) In the Media Shop, the users' satisfaction varied from good to excellent for the usability issue (Part 1a); and from regular to excellent for the content adequacy issue (Part 1b). In the Dental Project, the variation was from very good to excellent for usability (Part 2a); and excellent for the majority of the stakeholders in the content adequacy issue (Part 2b). Regarding usability – it was better evaluated on the Dental case – the differences between the analyzed systems were not so evident. Observing the adaptability, we can conclude that the use of the IFCAUC significantly contributed to the users' satisfaction. In the Dental case, the content adequacy was recurrent, respecting the user's preferences, the device's features and other ubiprofiles. In the Media Shop, the variation with regard to users' satisfaction was a concern for our group, which sought to solve it by developing the IFCAUC. In order to facilitate the comparison, we considered the worst 16 evaluations for each issue and project.

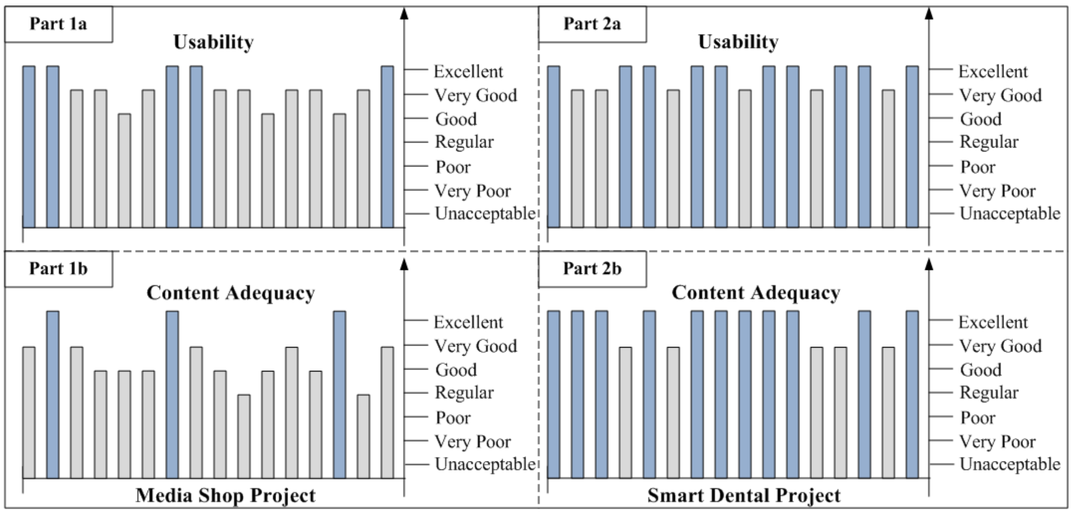



**Figure 7: Evaluating the Users' Satisfaction based on Usability and Content Adequacy Issues**

## 7 Related Work focused on NFRs in Ubiquitous Contexts

There are interesting NFRs-driven approaches: In [23], the authors describe an RE methodology (PriS) to incorporate privacy requirements into the design process. PriS supports the privacy-related requirements modeling and their issues by matching them with implementation techniques. Online service providers that ensure their clients' privacy are the potential users for PriS. In [24], Lamsal proposes a requirements collection for modeling trust in ubisystems and ad-hoc networks, emphasizing how difficult it is to deal with abstract concepts such as trust. His approach is based on a pre-defined model, organized in five steps: from the scenario analysis and conceptual requirements to the model breaks investigation. Finally, the author presents the main aspects for modeling trust: configurability; access control; connection; negotiation protocol; quantification; security; and context dependency. In [25], the authors argue that Web Services demand capacity to dynamically respond to different NFRs. In this field, the authors focus on quality of service provision for Web Services consumed in ubiquitous business environments. Basically, they support existing generic UDDI compatible catalogues through an architecture for Web Service discovering, which dynamically deals with QoS categories and metrics. In [26], the authors describe their context-driven infrastructure to create pervasive service instances, dealing with non-functional issues. The pervasive service instantiation is created according to the user's preferences and each device's features. Based on several NFRs identified in pervasive contexts, they developed an infrastructure to perform activities considering changeable working conditions. They intend to improve it providing support for cross-cutting non-functional issues (e.g. security).

Unfortunately, most of these approaches are issue-specific- and application-specific-oriented. It is difficult to reuse their support in other ubisystems that do not match with these issues or application profiles. Moreover, they focus on design and implementation disciplines, normally leaving the RE activities outside their approaches. Furthermore, they normally do not apply intentional MAS. The Distributed Intentionality and MAS paradigms are the top priority of investigations into different computer areas. Thus, it would be interesting to investigate this field for Ubiquitous Computing. Taking into consideration these timely observations, a guideline necessity to orient the engineers in the ubisystems systematic development requires approaches based on reusable models. This technological gap motivated us to develop a catalogue driven by NFRs reusable models for ubisystems, which pinpoints the gap between ubiquitous-issue-specific and software-engineering-generic supports. The former are inadequate as they normally consider a very specific ubiquitous concern, and the latter do not appropriately deal with the ubiquitous concerns.

## 8 Final Considerations

This paper reports experiences on the construction and use of a reusable model-based catalogue centered on GORE for ubisystems. The catalogue aggregates our efforts in the development of a common baseline of ubiquitous concerns, their interdependencies and operationalizations centered on emergent technologies obtained from the literature. The catalogue has been thoroughly investigated, carefully implemented and tested in our Software Engineering Laboratories at PUC-Rio and UofT over the past three years. Moreover, according to our evaluation of the framework in ubiquitous contexts, the reuse of existing and shared models demonstrated some benefits: (i) it improved the efficiency of the NFR elicitation process, by offering a suitable and reusable body of knowledge to deal with ubiquitous quality criteria; (ii) it significantly reduced elicitation time and team efforts in dealing with NFR issues; and (iii) it improved the quality of NFRs. The results suggest that our catalogue constitutes suitable support to guide the software engineers - from early requirements to code - in the systematic treatment of NFR issues. Moreover, thanks to its GORE nature, our framework also maintains traceability for RE activities, commonly left out during the development process.

However, we are conscious about our proposal also has weaknesses: (i) it demands time for the catalogue to be maintained and evolve; (ii) its effectiveness/efficiency depends on the collaboration and input of other groups; and (iii) it needs advanced mechanisms for knowledge management and version control for the baseline version and its extensions. In order to address such weaknesses, we have opened access to our catalogue, facilitating its use by our collaborators. On the basis of their feedback, we hope to develop more precise methods for removing unused information and prioritizing softgoals. We also envision further work on enhancing the catalogue with scenarios [27].

## References


1. Weiser, M. Some Computer Science Issues in Ubiquitous Computing. Communications of the ACM, Vol. 36, No. 7, pp. 75-84, July (1993)

2.Mylopoulos, J.; Chung, L.; Nixon, B. Representing and Using Non-Functional Requirements: A Process-Oriented Approach. IEEE Trans. on Soft. Eng., 18(6), June (1992)

3.Chung, L.; Nixon, B.; Yu, E.; Mylopoulos, J. Non-Functional Requirements in Software Engineering. Kluwer Acad. Pub., (2000)

4.Lamsweerde, A. van. Goal-Oriented Requirements Engineering: A Guided Tour. 5th IEEE International Symposium on RE'01, ISBN: 0-7695-1125-2, pp. 249-262, August (2001)

5.Weiser, M. The Computer for the 21st Century. ISSN 0036-8733, Vol. 265(3,) p. 94 (1991)

6.Abowd, D.; Dey, A. K.; Orr, R.; Brotherton, J. Context-Awareness in Wearable and Ubiquitous Computing. Journal Virtual Reality, Vol. 3, No. 3, pp. 200-211 (1998)

7.Ravindran, G.; Jaseemudin, M.; Rayhan, A. A management framework for service personalization. 5° IFIP/IEEE International Conference on MMNS'02, USA (2002)

8.Estrin, D.; Culler, D.; Pister, K.; Sukhatme, G. Connecting the Physical World with Pervasive Networks. IEEE Pervasive Computing, pp. 59-69 (2002)

9.Scholtz, J.; Consolvo, S. Toward a Framework for Evaluating Ubiquitous Computing Applications. IEEE CS and IEEE ComSoc, Pervasive Computing, pp. 82-88, June (2004)

10.Serrano, Milene. Media Shop Framework. Technical Report. DI/PUC-Rio, RJ, July (2007)

11.Serrano, Milene; Serrano, Maurício; Lucena, C. J. P. Engineering Multi-Agent Systems for Design and Implementation of Ubiquitous Computing. PUC-Rio, Monograph on Computer Science, No. 10/08, ISSN 0103-9741, 23 pages, March (2008)

12.Serrano, Milene; Serrano, Maurício; Lucena, C. J. P. Framework for Content Adaptation in Ubiquitous Computing Centered on Agents Intentionality and Collaborative MAS. Fourth Workshop on Software Engineering for Agent-oriented Systems (SEAS), 12 pages (2008)

13.Serrano, Milene; Serrano, Maurício; Lucena, C. J. P.; Leite, J. C. S. P. The Modeling and Implementation of Ubiquitous Applications based on Agents' Intentionality. PUC-Rio, Monograph on Computer Science, No. 52/08, ISSN 0103-9741, 20 pages, December (2008)

14.Serrano, Milene; Serrano, Maurício; Lucena, C. J. P. Ubiquitous Software Development Driven by Agents' Intentionality. 11th International Conference on Enterprise Information Systems (ICEIS'09), Vol. SAIC, pp. 25-34, Milan, Italy, May (2009)

15.Serrano, Milene; Serrano, Maurício; Lucena, C. J. P. Ubiquitous Software Development Driven By Agent's Intentionality (An Extended Version). PUC-Rio, Monograph on Computer Science, No. 53/08, ISSN 0103-9741, 21 pages, December (2008)

16.Serrano, Milene; Lucena, C. J. P. Systematic Development driven by Intentional MAS for Ubiquitous Computing. 92 pages (to be published as a Monograph at PUC-Rio) (2010)

17. Serrano, Milene; Lucena, C. J. P. Agent-Oriented Dynamic Database for Intentional Development of Ubiquitous Systems. 20 pages. (submitted to IDEAS'10) (2010)

18. Serrano, Milene; Lucena, C. J. P. "Applying FIPA Standards Ontological Support Intentional-MAS-Oriented Ubiquitous System," 8 pages (to appear in the Proceedings of 12th International Conference on Enterprise Information Systems (ICEIS'10), Funchal, Madeira, Portugal, June 2010.

19. Marca, D. A.; McGowan, C. L. SADT: Structured Analysis and Design Technique, McGraw-Hill, ISBN 0-07-040235-3, 392 pages, New York, USA (1987)

20. Lehman, M. M. Feedback in the Software Evolution Process. Information and Soft. Technology, Centre for Soft. Reliability Annual Conf., Vol. 38, Issue 11, pp. 68l-686 (1996)

21. Braubach, L., Pokahr, A. and Lamersdorf, W. Jadex: A Short Overview. NetObjectDays'04: AgentExpo (2004)

22. Caire, G. LEAP User Guide. TILAB, last update: December (2003)

23. Kavakli, E.; Kalloniatis, C.; Loucopoulos, P.; Gritzalis, S. Incorporating Privacy Requirements into the System Design Process. Internet Research Journal, Vol. 16(2), pp. 140-158 (2006)

24. Lamsal, P. Requirements for Modeling Trust in Ubiquitous Computing and Ad Hoc Networks. Ad-Hoc Mobile Wireless Networks, ISBN 951-22-6309-2 (2002)

25. Makripoulias, Y.; Makris, C.; Panagis, Y.; Sakkopoulos, E.; Adamopoulou, P.; Pontikaki, M.; Tsakalidis. A. Towards Ubiquitous Computing with Quality of Web Service Support. In The Fifth Anniversary of UPGRADE, Vol. VI, Issue 5, pp. 29-34, October (2005)

26. Preuveneers, D.; Berbers, Y. Automated context-driven composition of pervasive services to alleviate non-functional concerns. Int. Journal of Comp. and Information Sciences, Vol. 3, Issue 2, pp. 19-28, August (2005)

27. Leite, J. C. S. P. et.al. Enhancing a Requirements Baseline with Scenarios. Requirements Engineering Journal, 2(4):184-198 (1997)