8.
Final Considerations

"The original vision of ubiquitous computing, with an extensive real world web of networked objects and devices may take at least 10-15 years to come close to being realized. Indeed, it is unclear whether we will ever reach a situation where widespread intelligent, embedded technologies operate seamlessly in the environment around us. … These technologies are likely to develop rapidly over the next five years and will see a number of elements of ubiquitous computing being actively and usefully adopted"

David Ley Becta, "Emerging Technologies for Learning,"
ISBN 1 85379 467 8, Vol. 2, pp. 64-79, 2007.

User satisfaction, heterogeneous devices, autonomy, mobility, integration need, decentralized control, adaptability, context awareness, the balance between privacy and personalization as well as the balance between transparency and invisibility are all important challenges (requirements) that must be considered in the development of ubiquitous applications. Although there is a lot of interesting research focused on Ubiquitous Computing, there is a lack of approaches to provide engineering guidelines to support the incremental and systematic development of ubiquitous applications. In this thesis we proposed a step further. The systematic development of intentional-MAS-driven ubiquitous applications by promoting and facilitating software reuse to satisfy the set of requirements referred to above.

We believe the widespread need for developing new ubiquitous applications demands appropriate software engineering support. That was the focus of our research presented in this thesis. In this Chapter we present the main contributions of the proposed *Reuse-Oriented Approach for Incremental and Systematic Development of Intentional Ubiquitous Applications*, followed by a discussion about its limitation and we also outline open research problems suggested as future work.

8.1. Main Contributions

The work presented in this thesis represents a substantial step toward the advance of a new area which we called the incremental and systematic development of ubiquitous applications centered on the intentionality concept. The main contribution of our reuse-oriented approach is the formulation of support sets obtained from the *Domain Engineering of Ubiquitous Applications*. These support sets can be viewed as building blocks, which can be used directly or reused by instantiation or extension in the *Ubiquitous Application Engineering* process that guides and improves the development of intentional-MAS-driven ubiquitous applications. As follows we briefly provide an overview of those reuse-based building blocks by organizing them in five categories:

1. **Goal-Oriented Requirements Engineering (GORE)**

- The use of GORE support (e.g. i* Framework and NFR Framework) in our *Intentional Modeling Building Block* and *NFR Catalogue Building Block* focuses on activities that precede the requirements specification, and results in models that can be used during the design stage to drive and evaluate architectural decisions. It allows the evaluation of decisions impact centered on non-functional requirements that commonly influence the success of ubiquitous applications. Moreover, it facilitates the systematic refinement of models through the specification of all alternatives' contributions for the non-functional requirements under analysis.

- The use of a NFR Catalogue – from our *NFR Catalogue Building Block* – that offers *Softgoals Interdependency Graphs* for the main ubiquitous non-functional requirements by also specifying their interdependencies and operationalizations, mainly centered on the technological support provided by our reuse-oriented building blocks.

2. **Agent-Oriented Engineering**

- The use of cognitive entities centered on the support offered by our *Intentional Agents' Reasoning Building Block* to improve the development of ubiquitous applications. Software agents have interesting properties (e.g. autonomy, flexibility, mobility, adaptability, reactivity, proactivity and collaboration) that were combined to deal with challenges imposed by ever-changing contexts (e.g. their evolution over time and unpredictable behavior).

3. Intentionality-Driven Engineering

- The use of the intentionality concept to develop ubiquitous applications from the requirements to code. We propose to model ubiquitous applications requirements as suggested by our *Intentional Modeling Building Block* and to implement these applications as suggested by our *Intentional Agents' Reasoning Building Block* using the intentionality abstraction. According to (Bratman 1999; Gordon 2005; Dignum and Conte 1997), it represents an adequate way to deal with human practical reasoning (Bratman 1999) and to improve the like me recognition capacity (Gordon 2005), goals formation (Dignum and Conte 1997) and human-mental states interpretation/representation. In higher abstraction levels, we use the i* Framework to graphically represent the ubiquitous applications requirements by using, for example: (i) goals for functional requirements; (ii) softgoals for non-functional requirements; (iii) tasks for alternatives that will be chosen by intentional agents to adequately achieve the goals; and (iv) resources for anything that must be exchanged between agents (e.g. contents and messages). In the lower abstraction levels, we use the JADEX Framework to implement intentional agents centered on the BDI model and the concept of capability. A capability consists of a reuse-based module composed of beliefs, plans and goals that allows the agent to improve its knowledge by learning how to perform extra activities. In other words, the agents can acquire specific capabilities – at runtime – to learn how to deal with specific services, which can improve the agents' cognitive ability in ever-changing contexts. Furthermore, we have the *Fuzzy-Logic-Based Support*, which is combined with the *Intentional Agents' Reasoning Building Block* to improve the agents' reasoning engine and, consequently, its cognitive capacity in dealing with non-functional requirements at runtime by deciding – running fuzzy conditional rules – which actions must be performed to better attend these requirements.

4. Reuse-Based Engineering

- The use of a *Reuse-Oriented Architecture* centered on the support sets obtained from the *Domain Engineering of Ubiquitous Applications* by focusing on the development for reuse. The use of a *Reuse-Oriented Ubiquitous Application Engineering* to incrementally and systematically develop intentional-MAS-driven ubiquitous applications by focusing on the development with reuse. The support sets are building blocks centered on the main ubiquitous concerns to improve the development of intentional applications in ever-changing contexts from the requirements to test. Some pairs of concern and building block are:

    - Intentional Modeling: *Intentional Modeling Building Block* based on the i* Framework;
    - Non-Functional Requirements in Ubiquitous Applications: *NFR Catalogue Building Block* based on the NFR Framework;
    - Integration of Distributed Ubiquitous Environments and their intrinsic Heterogeneous Devices: *Integration Building Block* based on the JADE-LEAP Platform resources;
    - Content Adaptability Issue: *Ubiquity Issues Building Blocks*, more precisely the IFCAUC technological resources;
    - Dynamic Construction of Interfaces by considering Heterogeneous Devices: *Dynamic Interface Construction Building Block* based on the FIPA Standards Ontological Support;
    - Management of Ubiquitous Profiles at Runtime: *Dynamic Database Building Block* based on the Type-Square Architecture, the WURLF Repository and a Persistence Framework; and
    - Privacy versus Personalization & Transparency versus Invisibility: *Intentional Agents' Reasoning Building Block* centered on the *Capability* concept.

5. Product-Line-Based Engineering

- A possible view for our approach is the application of it as a product line (Dehlinger and Lutz 2008) for intentional-MAS-driven ubiquitous applications. In this view, our reuse-oriented approach offers software engineering conceptual models, frameworks and patterns for creating a collection of similar ubiquitous applications. Moreover, we can consider another activity in the SADT notation (Chapter 5 – Figure 5.1), called *Engineering of Cognitive-Domain-Based Ubiquitous Applications*. In this extended SADT:

- the *Domain Engineering of Ubiquitous Applications* represents the first activity and also the first layer in our feature-based process, briefly presented in Figure 8.1. This layer offers generic building blocks to facilitate the development of intentional-MAS-driven applications from different cognitive domains;
- the intermediary activity of the extended SADT represents the next layer in our feature-based process, the *Engineering of Cognitive-Domain-Based Ubiquitous Applications*, in which the generic building blocks are extended or instantiated to deal with concerns commonly found in intentional-MAS-driven applications from a specific cognitive domain (e.g. e-commerce, dental clinic and educational). Here, the main idea is to offer building blocks specialized in the desired cognitive domain. For example, in the dental clinic cognitive domain: (i) the *Dynamic Interface Construction Building Block* of this layer could provide an adequate ontological support to specifically deal with the dynamic construction of dental forms commonly used in different dental clinics; and (ii) the *Dynamic Database Building Block* of this layer could provide a specialized User Profile with Property-Type(s) to store information related to patients, dentists and attendants as well as a specialized Content Profile to store contents that are commonly found in this cognitive domain, such as x-rays, educational dental videos, and dental forms.
- the last activity of the extended SADT is the more specialized layer of the feature-based process, represented by the *Ubiquitous Application Engineering*. In this layer the cognitive-domain-based building blocks are instantiated in order to develop a specific ubiquitous application from this cognitive domain. The

idea is to facilitate this development and, consequently, the software engineers' work by reusing the building blocks provided by the intermediary layer (specialized building blocks for the cognitive domain under analysis) instead of the ones provided by the first layer (generic building blocks).

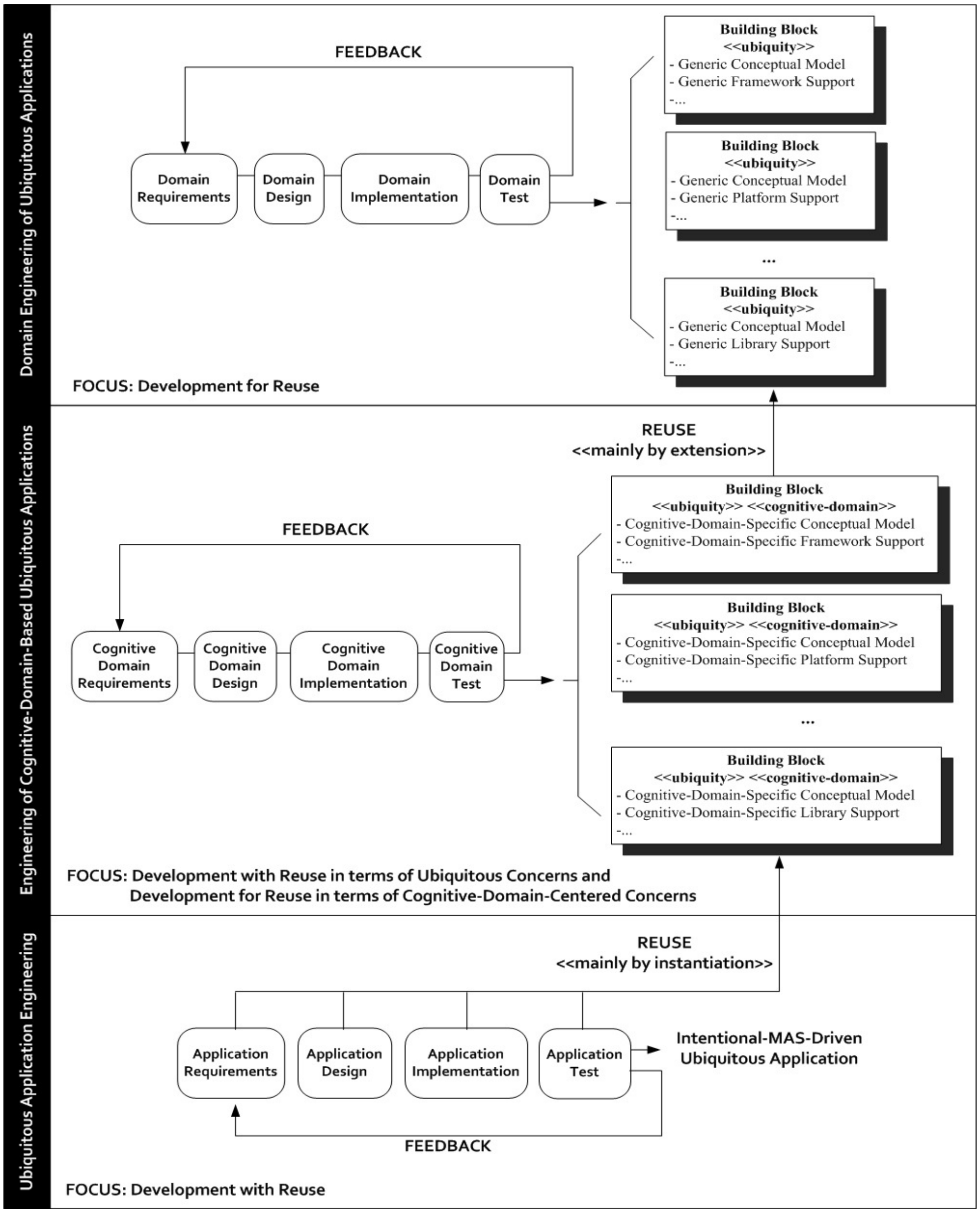
Domain Engineering of Ubiquitous Applications
FEEDBACK
Domain Requirements
Domain Design
Domain Implementation
Domain Test
Building Block
<<ubiquity>>
- Generic Conceptual Model
- Generic Framework Support
-...
Building Block
<<ubiquity>>
- Generic Conceptual Model
- Generic Platform Support
-...
...
Building Block
<<ubiquity>>
- Generic Conceptual Model
- Generic Library Support
-...
FOCUS: Development for Reuse
Engineering of Cognitive-Domain-Based Ubiquitous Applications
REUSE
<<mainly by extension>>
FEEDBACK
Cognitive Domain Requirements
Cognitive Domain Design
Cognitive Domain Implementation
Cognitive Domain Test
Building Block
<<ubiquity>> <<cognitive-domain>>
- Cognitive-Domain-Specific Conceptual Model
- Cognitive-Domain-Specific Framework Support
-...
Building Block
<<ubiquity>> <<cognitive-domain>>
- Cognitive-Domain-Specific Conceptual Model
- Cognitive-Domain-Specific Platform Support
-...
...
Building Block
<<ubiquity>> <<cognitive-domain>>
- Cognitive-Domain-Specific Conceptual Model
- Cognitive-Domain-Specific Library Support
-...
FOCUS: Development with Reuse in terms of Ubiquitous Concerns and
Development for Reuse in terms of Cognitive-Domain-Centered Concerns
Ubiquitous Application Engineering
REUSE
<<mainly by instantiation>>
Application Requirements
Application Design
Application Implementation
Application Test
Intentional-MAS-Driven Ubiquitous Application
FEEDBACK
FOCUS: Development with Reuse

Figure 8.1 - Product line view for our reuse-oriented approach

- Summarizing this view, we can consider that each layer adds functionalities to previously composed layers. Furthermore, different compositions of layers produce building blocks with specialized conceptual models, specialized frameworks, and other contributions.

Although this product line view was not the focus of our work, we consider our first effort in this research area – by providing reuse-oriented building blocks that can be specialized for a specific cognitive domain – as another interesting contribution of our proposal. It is possible to extend our approach's capacity by investigating this view in future work. Therefore, a relevant contribution in this context could be a configuration tool that assists the software engineers during the development process. It could allow, for example, the selection of features that would compose the desired intentional ubiquitous application. Here, it is important to consider that such decisions – which features must be selected or not selected – must respect the constraints specified by the feature model as well as the ubiquitous profiles information (e.g. users' preferences, devices features and other specifications).

8.2. Limitations

Given the ubiquitous challenges (requirements) described in this thesis as well as its proposed goals, it is relevant to consider the limitations of our approach. First, we understand that to provide a complete support to a complex, unpredictable and emergent domain such as the Ubiquitous Applications Domain demands efforts not only from Software Engineering in general but it also depends on many other multi-disciplinary contributions. Our intention with the proposed approach is to give the first steps towards this goal.

Considering our purpose of offering engineering guidelines for the construction of ubiquitous applications, we are aware that our approach is not able to guide the development of very specific ubiquitous applications. However, it was not our intention to provide a complete methodology or process. Instead, we meant to provide engineering guidelines – from the requirements to test – to help the software engineers' work in dealing with concerns commonly found throughout the development of ubiquitous applications, such as: (i) the necessity of integrating distributed smart-spaces; (ii) the ability to work with heterogeneous devices, which can be mobile or not and limited or not; (iii) the prerequisite of dealing with user satisfaction, privacy, personalization, invisibility, transparency and others non-functional issues; (iv) the necessity of adapting over time by following the ubiquitous context's evolution; (v) the capacity of representing the users by reasoning at runtime, achieving goals, performing tasks, dispatching plans and internal events; and (vi) the necessity of mechanisms that provide to the ubiquitous application the properties: autonomy, flexibility, mobility, reactivity, proactivity and collaboration. Furthermore, we intended to show how useful the proposed approach is in helping software engineers to incrementally and systematically develop intentional-MAS-driven ubiquitous applications by promoting and facilitating software reuse.

Moreover, it is relevant to emphasize that the proposed approach is in constant evolution, which ideally demands a collaborative work of different research groups to be effectively evaluated.

8.3. Future Work

In this Section we present some problems related to our research that can be investigated in future work.

As the Ubiquitous Computing is a grand vision for computing, improvements may be needed as different technological support sets that contribute to promote the software reuse – in a large scale – for the development of intentional-MAS-driven ubiquitous applications. It means both the construction of new frameworks, libraries, models, catalogues and platforms and the refinement of the suggested support to continually evolve and maintain the proposed approach.

Thinking about punctual contributions, interesting research may be conducted to improve the agents' reasoning engine to deal with non-functional requirements at runtime. An interesting work has been developed at PUC-Rio by the Requirements Engineering group (Serrano et al. 2011b). The group deeply investigates the use of a complex mechanism based on fuzzy logic libraries to improve the reasoning of software agents in dealing with non-functional requirements at runtime. The agents dynamically make decisions according to the context under analysis by using a rigorous analysis of different alternatives of action and the impact of them on the non-functional requirements of the application. This mechanism is an interesting improvement for our simple fuzzy-logic model.

Another relevant contribution would be to bring improvements to the agents' learning process. Human learning normally occurs as part of education, training and personal development. Moreover, it may occur as a result of our daily activities (i.e. tasks that we habitually perform in our life). It is important to say that this learning may be goal-oriented (as our approach considers). However, it also may be aided by motivation (not considered in our approach). Therefore, a complete human learning approach demands studies centered on several disciplines (e.g. educational psychology, learning theory and neuropsychology). The guidelines proposed in (Holyoak and Morrison 2005) and (Heit 2000) may integrate learning and reasoning into the ontological support by applying inductive reasoning to better detect the agent's learning evolution at runtime.

Considering the proposed NFR Catalogue, our approach also has drawbacks: (i) there is clearly overhead for the catalogue to be maintained and evolved; (ii) its effectiveness/efficiency depends on the collaboration and input of other groups; and (iii) it requires advanced mechanisms for knowledge management and version control for the baseline version and its extensions. In order to address such drawbacks, we have opened access to our catalogue, facilitating its use by our collaborators. On the basis of their feedback, it is possible to develop more precise methods for removing unused information and prioritizing softgoals.

Focusing our attention on our dynamic database model, an appropriate contribution would be the incorporation of a dynamic reorganization algorithm (Sockut and Iyer 2009), which can improve the database reorganization based on the agents' intuition and centered on different metrics, such as: user's intentions, different priorities, most accessed knowledge and data deterioration rate.

Furthermore, another opportunity of future work would be the development of experimental work to evaluate the **S**tandard **O**ntology for **U**biquitous and **P**ervasive **A**pplications (SOUPA) (Chen et al. 2004) in intentional-MAS-driven ubiquitous applications, and to compare SOUPA and FIPA Standard Ontology in terms of reuse facility and agents' communication and inter-operability.