Towards Model-Driven Requirements Analysis for Context-Aware Well-Being Systems

Steven Bosems

University of Twente,
Department of Electrical Engineering, Mathematics and Computer Science,
Enschede, The Netherlands
s.bosems@utwente.nl

Abstract. Over the years, the interest in the field of pervasive computing has increased. A specific class of applications in this domain is that of context-aware applications. These programs utilize context information to adapt to their current environment. This quality can be used, among others, when dealing with health care and well-being situations. However, as the user requirements for these specific applications are almost never well-specified, there is a real risk that the resulting application does not offer the right set of features to the user. In order to mitigate this risk, we propose a model-driven method of requirements engineering for systems in the domain of context-aware well-being applications. This method will result in an explicit specification of requirements, and an improved alignment of user requirements and system features. Furthermore, due to the model-driven character of the method, the artifacts created during the requirements engineering phase of the development process can directly be incorporated in the subsequent development steps.

Keywords: requirements, architecture, model-driven development, pervasive, context-aware.

1 Introduction

According to Scopus [2], the field of pervasive computing has increased drastically over the last 15 years, the number of published research papers increasing from less than 500 to over 3,000 per year. In order to make pervasive systems adaptive to their environment, we add context information to them. In these context-aware applications, sensors are used to monitor the user's environment such that the experience the user has when using the application can be improved. By analyzing the data collected by the sensors and reasoning about this information, the program can adapt itself in order to better suit the current situation the user is in, provide relevant information to the user, or offer services that are deemed useful. One domain in which this type of application can be particularly useful, is that of health care and well-being. In this PhD research, we will be focusing on this specific field of application.

This research will be performed as part of the COMMIT SWELL project [1]. The focus of this project is the well-being of knowledge workers. Over the years,

P. Herrero et al. (Eds.): OTM 2012 Workshops, LNCS 7567, pp. 43–50, 2012.

this group has seen a decline in their well-being [9]. SWELL aims to improve both the physical and the mental well-being of knowledge workers by providing the users with a context-aware well-being system. However, the features offered by this system have to be aligned with the demands of the user. If this is not the case, the system will be disregarded. As such, the process of requirements engineering and alignment in context-aware applications is of key importance. During this PhD research, we aim to improve this process.

Our contribution to the field of requirements engineering is a model-driven method of requirements elicitation in the domain of context-aware well-being applications, which can be used to increase user involvement in the software development process, resulting in higher user satisfaction in the resulting application. It is to be noted that the planned method is to be used while designing and developing a context-aware well-being system. We do not intend to utilize it for runtime alteration of the system's behavior; we deem a model-driven method using model transformations unsuitable for runtime adaptation of these systems.

The rest of this paper is structured as follows: Section 2 discusses the work done in the domain of context-aware applications and model-driven development, Section 3 introduces the hypothesis for our research, Section 4 outlines the planning for the rest of this thesis, Section 5 looks at some possible drawbacks of the proposed method and Section 6 provides concluding remarks.

2 Related Work

In 1991, [19] explored the idea of ubiquitous computing systems, interconnected by wired and wireless technology, that would be invisible to the users. The author predicted the rise of small computing devices that could be used and combined in order to provide the best possible user experience. The inclusion of context information and the term context-awareness, however, were not used until 1994. [16] was the first to define the term context-aware computing. The authors identify the location, people, hosts, accessible devices, and changes of these over time as relevant contexts for mobile distributed computing systems.

[6] defines context information in a broader way as "any information that can be used to characterize the situation of an entity" and defines an entity as "a person, place or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves." The authors deem a system context-aware, if this context is used in order to "provide relevant information and/or services to the user." They note that this relevancy is dependent on the task of the user. Context-aware applications can be categorized according to three features: presentation of information, execution of services and tagging of context to information. Four context types are recognized: activity, identity, location and time. It is noted that "activity" describes the environment and everything that happens in it at the given time. The authors further describe that these context types are primary context types. Using this information, secondary context types can be derived from other sources.

The authors of [7] distinguish between intrinsic and relational context, the former inhering in a single entity and the latter being defined as a property of an entity in relationship to one or more other entities. They also introduce the concept of situation, which represents a state of affairs of interest to the application, defined in terms of temporal constraints on context value.

In the field of requirements engineering, multiple methods exist in order to model goals and requirements. Modeling goals of the users and stakeholders is part of the requirements engineering process. KAOS [11], use case models [15], and i* [20] are some examples of such methods and languages. When looking at these, however, we find that the level of abstraction at which the goals are specified is too high to be directly usable for model transformations into system architectures; for this additional information is required. [4] uses a Model-Driven Architecture-based approach to transform business rules, located at the Computational Independent Model (CIM) level, into Platform Independent Models (PSMs), however, they do not supply us with the additional step of generating Platform Specific Models (PSMs), i.e. system architectures.

Introduced in 2001 by the Object Management Group (OMG), the Model-Driven Architecture (MDA) [12] explains how OMG standards may be used together. MDA focuses on the relation between the concepts of Platform Independent Models (PIMs) and Platform Specific Models (PSMs). Model-Driven Engineering (MDE) [10] incorporates these techniques, and adds tool support to aid developers in maintaining models, the primary artifacts in this process, at different levels of abstraction. Model transformations are used to translate between these levels. We shall use the term "model-driven" for any process using these transformation techniques, having models as primary development artifact. [3] argues that model-driven development should not be structured as the OMG proposes it in MDA. The author calls this "Generative MDD." Rather, he suggests that MDD techniques are to be used in an agile fashion, "Agile MDD", iteratively creating models and writing code. MDD techniques are then used to keep these artifacts synchronized.

[8] defines a way to model the context of pervasive systems. The method proposed allows for the description of the concept of a 'Person', who is authorized to use, or is located near a certain 'Device'. The Person also has a (communication) 'Channel', which in turn requires a Device. 'Location Coordinates' are used to locate the Person and the Device. The authors allow for the use of different association types in the models, including 'Static associations', 'Derived associations', and 'Sensed associations'. Dependencies can be added between associations in order to guarantee a stable system. Furthermore, associations can be annotated with quality parameters, indicating accuracy and certainty of these connections between entities.

In [13], the authors introduce the Pervasive Modeling Language (PervML). This Platform Independent Modeling (PIM) language allows the user to define the pervasive system in an abstract way that does not rely on the underlying implementation technology. The authors propose to use model transformations in order to obtain Open Services Gateway initiative (OSGi) models. These

Platform Specific Models (PSMs) can then be used to generate OSGi-based Java programming code. A PervML model specifies a *ServiceModel*, which in turn has zero or more *Service*s. These model the way different parts of the system inter-operate, defining triggers, pre- and postconditions, and service aggregation. According to [14], a model that will facilitate the description of requirements for a context-aware system, is to capture the characteristics of the physical environment, a description of tasks and a description of the system behavior. Using this information, a PervML model can be generated. [17] uses PervML and the PervML Generative Tool (PervGT) [5] to develop a context-aware system in a model-driven way. Their conclusions are that MDD of these systems decreases development time, increases reusability and increases software quality. For our purpose, however, PervML is too low level, as the language requires training in systems architecture for a user of the final product to understand it, thus hindering direct user involvement in the development process.

3 Research Hypothesis

As experience shows, a discrepancy often exists between user demands and features offered by the final version of the system that is being developed. It is the task of the requirements engineer, the systems architect and the system developer to minimize this gap between what is expected, and what is offered. In pervasive computing, even more so than in regular software applications, the system must do exactly as the user anticipates it will do. If this is not the case, the system becomes a hindrance and is no longer pervasive.

In current requirements elicitation practices, the link between the user requirements and the technical implementation is often neglected. As such, the requirements have to be reinterpreted and re-elicited in subsequent development steps. Furthermore, current tools and practices do not deal with the specific challenges the well-being domain poses.

Because of the importance of proper requirements engineering in the field of pervasive systems, our research results in the definition of a domain specific modeling language for the specification of requirements of context-aware applications, along with model transformations that will allow us to transform between user requirements and system architectures. Through the use of this technique, the requirements and the architecture of the system that is to satisfy them can be kept synchronized.

The method envisioned is structured as follows: a user provides requirements, which are captured in a model. The requirements engineer can use this model to continue the requirements elicitation process. The completed model is then to be transferred to the system architect, who makes sure the architecture fits the requirements. This step is to be performed in a model-driven fashion, using model transformations. The completed architectural model can then be used by the developer of the system. However, during this process, feedback is continuously provided to people working on the project: alterations in the architecture will have to be reflected in the requirements. For this flow to be supported, model transformations can also be used. Figure 1 illustrates this process.

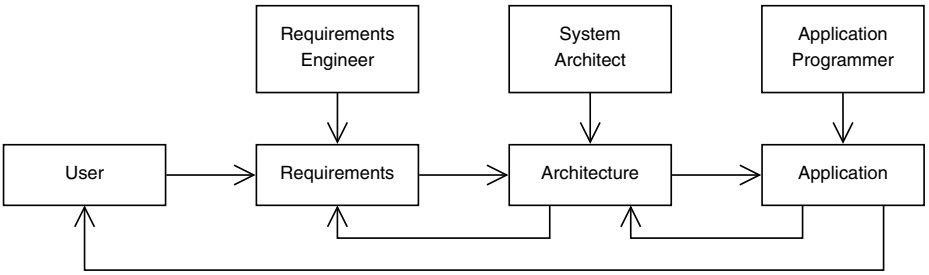

Fig. 1. Conceptual framework

In this PhD project, we will be researching the validity of the following hypothesis:

> *Through the use of model-driven engineering techniques, the process of creating user requirements for context-aware well-being applications can be improved, resulting in programs that better suit the demands of the users, than those that were created using general purpose requirements engineering techniques.*

4 Work Plan

In order to prove that the use of model-driven techniques involving the user requirements of context-aware applications is beneficiary to the process of the creation of well-being applications, several steps will have to be taken.

Firstly, we will have to identify the methods currently used in the field of requirements engineering for pervasive systems. A distinction shall be made between general purpose software requirements engineering methods, and methods tailored specifically to pervasive systems. Both model-driven and non-model-driven techniques shall be evaluated. This research shall be performed as a structured literary survey.

Secondly, research will be conducted to get a full overview of the domain in order to create meta-models for the model-driven process. These meta-models will be used to define models of the user requirements and system architectures, and should contain all elements that could be present in the field of pervasive systems: the requirements of the user toward the system itself, the specification and requirements of the context, and a profile of the user. The validity of these models shall be tested through interviews with domain experts.

Thirdly, a model-driven method will be created that uses the meta-models described in the previous step. This method will aim to involve the intended users of the system under development in the process of creating requirements. By using model-driven techniques, we will allow these users to express themselves in a non-technical way, while still creating artifacts that are used throughout the rest of the design process. Also, we aim to keep these resulting artifacts and

the used requirements synchronized by utilizing bi-directional model transformations. This method will differ from [13], [14], and [17], in that it will focus on the elicitation of requirements and generation of a system architecture, whereas the mentioned sources aim at the actual development of the systems themselves.

Fourthly, tool support for the creation, maintenance, and transformation of the models mentioned in the previous step will have to be provided. In order to do so, plug-ins for the Eclipse platform [18] will be created. Through this process, we will be able to transform user requirements into PervML models, which in turn can further be transformed as described in [17].

Finally, we will verify that the proposed method, supported by the tool, does indeed result in an improvement of the developed applications. For this, case studies and interviews with the intended product users shall be conducted. One of these case studies will be in the COMMIT SWELL project, researching whether direct transformation from requirements into an architecture yields better results, than when requiring interpretation of requirements by domain experts and manual translation of these into an overall system architecture, as is currently the case.

5 Discussion

As the proposed model-driven process is likely to involve additional work from developers and architects when compared to traditional methods of working, it is likely that the overhead of this process might be counterproductive for small-scale projects. Due to this possible problem, it is important to gain knowledge on the estimated size of the project at hand before choosing to use our method.

As we are involving users in the design process of software, we have to create a tool to aid us with this. The design of the tool is to be adapted with regard to the intended audience: traditional design tools are used by software developers that work with these tools on a regular basis. As such, they gain knowledge into the workings of the tool and structure their work flow around them. However, as the users of these tools will be non-experts, the workings of the tool will have to be clear to them when using it for the first time. Experts in the field of human-computer interaction and interface design might have to be consulted.

6 Conclusion

Over the years, the well-being of knowledge workers has been degrading. The COMMIT SWELL project aims to improve this using pervasive, context-aware applications. In order to improve the alignment of user requirements and features offered by the system envisioned, we propose to use model-driven techniques during the process of requirements engineering of these well-being applications. We hypothesized that this will result in programs that better suit the user's requirements, then when using general purpose requirements engineering techniques. A work plan has been illustrated, and a discussion was provided with regard to potential problems when implementing our methodology.

Acknowledgments. We would like to thank Marten van Sinderen for his insights and comments.

This publication was supported by the Dutch national program COMMIT (project P7 SWELL).

References

[1] Commit: A public-private research community (2011), http://commit-nl.nl/
[2] Scopus (March 2012), http://www.scopus.com
[3] Ambler, S.: Agile model driven development is good enough. IEEE Software 20(5), 71–73 (2003)
[4] Castro, V.D., Marcos, E., Vara, J.M.: Applying CIM-to-PIM model transformations for the service-oriented development of information systems. Information and Software Technology 53(1), 87–105 (2011), http://www.sciencedirect.com/science/article/pii/S0950584910001588
[5] Cetina, C., Serral, E., Muñoz, J., Pelechano, V.: Tool support for model driven development of pervasive systems. In: Fernandes, J.M., Machado, R.J., Khedri, R., Clarke, S. (eds.) Proceedings of 4th International Workshop on Model-based Methodologies for Pervasive and Embedded Software, pp. 33–41. IEEE Computer Society Press (March 2007)
[6] Dey, A.K., Abowd, G.D.: Towards a better understanding of context and context-awareness. In: Computer Human Intraction 2000 Workshop on the What, Who, Where, When, Why and How of Context-Awareness (2000), ftp://ftp.cc.gatech.edu/pub/gvu/tr/1999/99-22.pdf
[7] Dockhorn Costa, P., Almeida, J.P.A., Ferreira Pires, L., van Sinderen, M.: Situation Specification and Realization in Rule-Based Context-Aware Applications. In: Indulska, J., Raymond, K. (eds.) DAIS 2007. LNCS, vol. 4531, pp. 32–47. Springer, Heidelberg (2007), http://doc.utwente.nl/61762/
[8] Henricksen, K., Indulska, J., Rakotonirainy, A.: Modeling Context Information in Pervasive Computing Systems. In: Mattern, F., Naghshineh, M. (eds.) PERVASIVE 2002. LNCS, vol. 2414, pp. 167–180. Springer, Heidelberg (2002), http://dx.doi.org/10.1007/3-540-45866-2_14
[9] Hooftman, W., Hesselink, J.K., van Genabook, J., Wiezer, N., Willems, D.: Arbobalans 2010: Kwaliteit van de arbeid, effecten en maatregelen in nederland. Tech. rep., TNO (2011)
[10] Kent, S.: Model Driven Engineering. In: Butler, M., Petre, L., Sere, K. (eds.) IFM 2002. LNCS, vol. 2335, pp. 286–298. Springer, Heidelberg (2002)
[11] Lamsweerde, A., Dardenne, A., Belcourt, F.D.: The KAOS project: knowledgeacquisition in automated specification of software. In: Design of Composite Systems. Proceedings of AAAI Spring Symposium Series, pp. 59–62 (1991)
[12] Miller, J., Mukerji, J.: MDA guide version 1.0.1. OMG document. Object Management Group (2001), http://www.omg.org/cgi-bin/doc?omg/03-06-01
[13] Muñoz, J., Pelechano, V., Fons, J.: Model driven development of pervasive systems. In: Intl. Workshop on Model-Based Methodologies for Pervasive and Embedded Software (MOMPES), pp. 3–14 (2004)
[14] Muñoz, J., Valderas, P., Pelechano, V., Pastor, O.: Requirements engineering for pervasive systems. a transformational approach. In: Requirements Engineering, 14th IEEE International Conference, pp. 351–352 (September 2006)

[15] Object Management Group: OMG Unified Modeling Language (OMG UML), infrastructure, v2.1.2 (2007), http://www.omg.org/spec/UML/2.1.2/Infrastructure/PDF
[16] Schilit, B., Adams, N., Want, R.: Context-aware computing applications. In: Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications, WMCSA 1994, pp. 85–90. IEEE Computer Society, Washington, DC (1994), http://dx.doi.org/10.1109/WMCSA.1994.16
[17] Serral, E., Valderas, P., Pelechano, V.: Towards the model driven development of context-aware pervasive systems. Pervasive and Mobile Computing 6, 254–280 (2009)
[18] The Eclipse Foundation: Eclipse (2012), http://eclipse.org/
[19] Weiser, M.: The computer for the 21st century. Scientific American 265(3), 94–104 (September 1991), http://doi.acm.org/10.1145/329124.329126
[20] Yu, E.: Towards modelling and reasoning support for early-phase requirements engineering. In: Proceedings of the Third IEEE International Symposium on Requirements Engineering, pp. 226–235 (January 1997)