Situation-Driven Development: a Methodology for the Development of Context-Aware Systems

David Martín[1], Diego López de Ipiña[2], Carlos Lamsfus[1] and Aurkene Alzua[1]

[1]CICtourGUNE – Competence Research Centre in Tourism, Donostia - San Sebastián
[2]Deusto Institute of Technology - DeustoTech, University of Deusto, Bilbao

{davidmartin, carloslamsfus, aurkenealzua}@tourgune.org;
dipina@deusto.es

Abstract. Several toolkits have been proposed in order to ease the development of context-aware systems, providing high-level programming interfaces to manage context data. One of the main tasks in the development of such systems is the definition of user situations that have to be identified by the system in order to adapt its behaviour. These situations are best defined by domain experts, but usually they do not have programming skills. Apart from that, there is a lack of methodologies to guide the development process. This paper presents a methodology based on the definition of situations that is designed to involve domain experts in the development process. This way, they can support programmers in the definition of the required situations. Also, a web-based platform has been implemented in order to manage context data without any programming skills. This way, domain experts can also configure the situations to be detected by the system.

Keywords: Context-Awareness, End-User Programming, Mobile Services.

1 Introduction

Context-aware systems provide personalized services to users at any time and place, based on the identification of the situation of the user. Nevertheless, the development of these systems is difficult for programmers. There are several technical challenges that have to be faced: context sources have to be identified, information has to be obtained from these distributed sources, data has to be modelled in order to be processed by computers, the situation of the user has to be identified, and finally, the system has to be adapted to the identified situation.

Apart from that, it can be difficult for a programmer to identify the needed situations and the desired behaviours of the system to be developed once a situation is detected, because they are usually dependent on the application domain. Domain experts, that is, people that are experts in the domain where the application is going to be deployed, can provide the needed expertise regarding the above requirements.

There are several research works that propose toolkits in order to ease the development of context-aware systems [1]. Most of them offer high-level application pro-

gramming interfaces for skilled programmers, so domain experts cannot be involved in the development process. The participation of users that do not have programming skills but are experts in the application domain can speed up and improve the development process of context-aware services.

This paper proposes a development methodology, which is called Situation-Driven Development, designed to involve domain experts in the development of context-aware systems in collaboration with programmers. The aim of the methodology is to guide the identification and parameterization of the relevant situations that have to be detected by the system to be developed.

Moreover, a web based platform has been designed where all the features involved in the development of context-aware services can be easily configured without writing any line of code. For instance, the context data model, the context information gathering process and the detection of users' situations can be configured using a web interface. The configuration of the platform is guided by the methodology and it makes the involvement of domain experts in the development process possible. Both the methodology and the platform have been successfully validated.

The paper is organized as follows. In section 2, the related work is presented. In section 3, the development methodology is explained. In section 4, the implementation of the platform is detailed. Section 5 presents the evaluation of the methodology and the platform. Finally, section 6 concludes the paper with brief concluding remarks.

2 Related Work

There are several software development methodologies that can be used in order to implement context-aware systems [2] (e.g. waterfall, iterative) but all these methodologies are designed to guide the development process of general software systems, so they do not consider the specific issues that are related to the development of context-aware systems.

Several authors have proposed development methodologies to be used in the implementation of context-aware systems. Henricksen and Indulska [3] propose a methodology and a modelling language called Context Modelling Language (CML). This language was developed for conceptual modelling of databases that store context data. It has different constructs for capturing the needed classes and context sources, the quality of context data and the dependencies and constraints between context fact types. This way, system designers can specify the requirements of context data needed by the system. The methodology is divided in five different stages: analysis, design, implementation, configuration of the system and validation. Context-oriented Programming (COP) [4] is a programming model that offers mechanisms to adapt the system to be implemented according to the gathered context data.

The above mentioned approaches are focused on system designers and programmers and do not involve non-technical users. In that way, domain experts cannot take part in the development process.

On the other hand, there are several context-aware toolkits to support the development of such systems. Most of the toolkits [5][6] can only be configured using high-

level application programming interfaces, and therefore only people with technical skills can use them in order to develop context-aware systems.

Other authors propose visual approaches where domain experts can take part in the development process. The iCAP toolkit [7] is a visual editor where domain experts can prototype context-aware systems using rules. DiaSuite toolkit [8] comprises a domain-specific design language, a compiler for this language and an editor to define simulation scenarios. The OPEN framework [9] is a programming environment for rapid prototyping of context-aware applications. It is based on the configuration of semantic rules in order to trigger predefined actions. All these toolkits have been designed in order to involve non-technical users in the development process, but they have some drawbacks. For instance, the iCAP toolkit does not have functionalities in order to define a context data model and it cannot be extended. DiaSuite toolkit provides a modelling language that can be quite difficult for a non-technical user because programming skills are required to define artefacts. The OPEN framework uses a semantic model to represent context data that can only be extended by skilled ontology experts.

3 Situation-Driven Development

In order to guide the development process of context-aware systems and involve non-technical domain-experts in collaboration with programmers, a methodology has been designed. This methodology is based on the definition of situations and it is based on the following premise: the programmers have context-aware toolkits that provide the needed functionalities in order to detect user situations and these toolkits can be configured without programming skills.

3.1 Definition of a situation

There are several definitions for the concept of "situation", and all of them have something in common: they consider context [10] as low-level data, while a situation is high-level data. This way, a situation is dependent on the context information and it can be considered as an abstraction of it [11].

A situation is defined as *the state of the current and past context at a certain region in space and a concrete interval in time that are relevant to identify that situation*. There are two main principles that have been taken into account for this definition. The first one is the notion of time. A situation can have temporal boundaries. It also can be related to past or current situations according to Allen's temporal logic [12]. The other one is the notion of space, that is, where the situation can be identified. For instance, the situation "waiting for the bus number 28" could be detected when the user is located at a certain bus stop (space) between 1 p.m. and 9 p.m. (time).

3.2 Description of the methodology

The aim of the methodology is to promote the collaboration among domain experts and programmers in the development of context-aware systems. This methodology

considers the different users' situations that are relevant to adapt the behaviour of the system to be developed. Like that, the used toolkits have to be able to provide mechanisms to detect users' situations and produce outputs based on these situations.

The methodology is divided into four different stages: analysis, configuration, development and validation. There are some stages where only the programmer can participate because they require some kind of development.

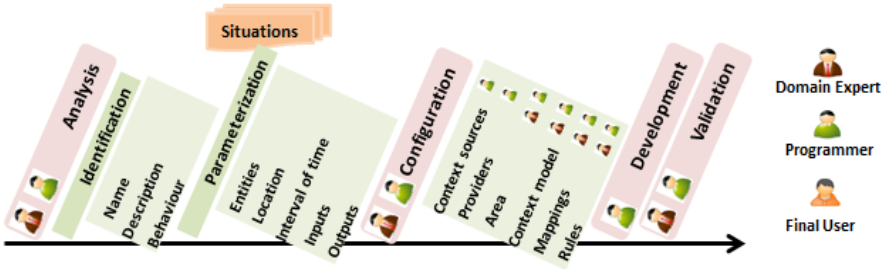

Fig. 1. Situation-Driven Development

- **Analysis.** In the analysis stage, domain experts and programmers have to identify all the situations that can be relevant for the system to be developed, specifying a name, a description and the desired behaviour of the system once the situation is detected. Also, each of the identified situations have to be parameterized with the entities that are involved in the situation, the location where the situation can be identified and the interval of time when the situation can be detected. The needed inputs of context data in order to detect the situation have to be specified as well, providing the objective, the conditions and the restrictions for each data type. Finally, the needed outputs once a situation is detected have to be specified. These outputs will be used by programmers in order to adapt the system's behaviour according to the previous specifications. In order to support the analysis stage, an excel sheet has been designed, where domain experts and programmers can discuss about the needed parameters of each of the identified situations.
- **Configuration.** Once the analysis stage is finished, the toolkit has to be configured with the specified parameters. In this stage, the programmer has to identify and configure the context sources that can provide the defined inputs of data, and configure or implement the providers that are going to obtain these data from the identified sources. The next step is the configuration of the areas where the situations can be detected, the context data model that will store context data, the mappings between the obtained data and the model, and the inference mechanisms in order to detect the needed situations. These configurations should be done by domain experts with the collaboration of programmers, so the toolkit has to provide configuration mechanisms in order to avoid the usage of programming languages.
- **Development.** The programmer has to implement the defined behaviours of the system to be developed, processing the outputs generated by the toolkit.
- **Validation.** Finally, the service has to be tested and validated by domain experts and programmers. Also, the final user could be involved in this stage.

4 Context Cloud: a Web Platform for the Development of Context-Aware Systems

In order to ease the implementation of context-aware systems a web platform has been developed. The aim of the platform is to promote the involvement of non-technical users in the development life-cycle according to the specified methodology. It provides a web environment where context data can be managed using a graphical interface without having to code any programming line. It also provides geospatial functionalities to manage location context data and all the configurations can be extended at runtime. This way, users without programming skills can actively participate in the development life-cycle of context-aware systems.

Context Cloud[1] can be considered as a black box that receives inputs from the identified context sources and produces outputs triggered by the defined context rules. The interactions between the platform and the third party services are made in a RESTful way. Figure 2 shows the system architecture.

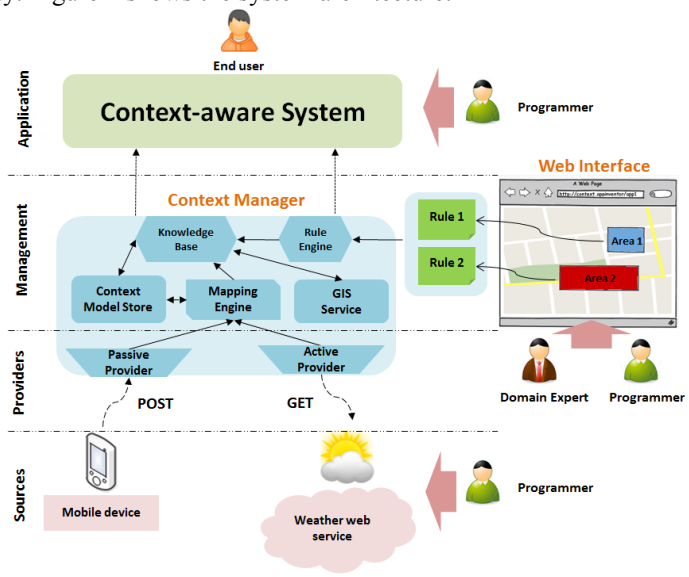

Fig. 2. Architecture of Context Cloud

The first layer is where all the context sources are (e.g. mobile devices, sensors, web services). The next layer contains the providers of the platform. These modules obtain data from the identified sources. There are two types of providers, the passive ones and the active ones. Passive providers wait until any of the registered sources send data to the platform in order to process it. On the other hand, active providers obtain data from the source making periodical GET requests. Context data has to be provided by sources using XML.

[1] https://vimeo.com/contextcloud

The third layer is where context data is managed and configured using the interface of the platform. This layer exposes a web front-end where the user can create areas, rules, context entities, mappings between context sources and entities and the context data gathering process from the identified context sources by the use of dialogs. The main component of the layer is the Context Manager which is responsible for the management of the context data life-cycle. Here, the defined context data model is transformed into Java Bean classes and it is stored in the Context Model Store. This way, the platform allows the creation of a context model defining entities with different typed properties. Situations are modelled as Event Condition Action (ECA) rules [13] that are validated and inserted into the rule engine of the Knowledge Base. The platform provides dialogs to create areas and rules assigned to these areas, which are defined by polygons that are created over a Google Maps layer. The Mapping Engine does the mappings between data coming from the configured context sources and the defined context entities, according to the user's defined mappings. It also saves or updates context entities instances into the Knowledge Base. The GIS Service translates the coordinates of context entities into a registered area name. This way, rules will only have into account context entities that are in their associated area. Finally, the rule engine is the responsible of firing all the defined rules. This component can also POST data in XML format to external services as a consequence of the defined rules.

Finally, the upper layer is where the context-aware system resides. This system will adapt its behaviour based on the identified situations and the outputs generated by the platform.

5 User Evaluation

The platform has been validated with 20 participants. They carried out the evaluation in pairs composed by a domain expert and a programmer. The evaluation was inspired in a tourism scenario, so the non-technical users were experts in the tourism domain.

5.1 Methodology

First of all, the users were introduced to the Context Cloud platform and to the experiment's objectives. They were instructed on how to configure the platform and they were given an example on how to identify a situation using the methodology. The participants were given a text document where four different situations were described.

- Waiting for the bus (S1): the visitor waits for the bus at Bus Stop A and she receives an SMS with the estimated time of arrival for the next bus.
- Sunbathing (S2): the visitor is at the beach when she receives an SMS advising her that she should use sun cream because the temperature is higher than 30ºC
- Waiting for the bus (S3): the visitor waits for the bus at Bus Stop B and she receives an SMS with the estimated time of arrival for the next bus.
- Arriving to the hotel room (S4): the air conditioning is activated when the visitor goes into the room.

The situations were simulated using the Siafu Context Simulator[2]. It was configured to send context data about the visitor on the move to the platform. The simulator also provided some web services in order to obtain the weather information and to access the air conditioning system of the Hotel. The participants had to configure the platform in order to detect the above mentioned situations. During the test, an external observer annotated all the problems that the participants found using the platform. Also, once a situation was detected, the time spent in its configuration was annotated.

After having completed the user experience, each participant had to fill out a questionnaire based on a six-level likert scale, with values from 1 (totally disagree) to 6 (totally agree). The used survey was designed on the Technology Acceptance Model (TAM) literature, and in particular, it was adapted from David's studies [14]. This way, three constructs were considered: Perceived Ease of Use (PEOU), the Perceived Usefulness (PU), and the Behavioral Intention (BI).

5.2 Results

The 95% of the participants find that learning the methodology is easy and the 100% of them state that it eases the collaborative work. Also, the 100% of the participants find that the methodology is useful to work with the platform, and that it is useful to develop context-aware systems.

The 95% of the participants find that learning how to use the platform is easy. The 75% of the participants find it easy to get Context Cloud to do what they want to do. However, the other 25% disagree on that. The reason is that the participants were not used to work with these kinds of toolkits. The 95% of the participants also find that the interaction with Context Cloud is clear and understandable. The 90% of the non-programmers state that it would be easy for them to become skillful at using the platform.

The perceived utility of the platform is also highly supported by domain experts. The 100% of the domain experts state that using Context Cloud in their jobs would enable them to develop context-aware systems more quickly and that it would make it easier to develop context-aware systems. Also, all of the participants would recommend other users to use the platform and they would use it in future developments. In addition to this, the 80% of them would pay for the system.

The average time spent by each of the pairs to solve the evaluation test was 89 minutes. It is relevant that the they spent an average time of 37 minutes in order to solve the situation number one, while for the rest of the situations, the average time was 17 minutes. This means that once they know how to configure the platform in order to identify the first situation, it is easier for them to configure it for the rest of the situations. This way, the learning curve is steep, that is, the participants learn in a very short period of time how to use the platform successfully.

[2] http://siafusimulator.sourceforge.net/

6 Conclusions

This article presents a situation-driven development methodology in order to promote the collaboration between programmers and domain experts in the implementation of context-aware systems. Thanks to the designed platform, users without any programming skills can actively participate in the development life-cycle of context-aware services. This way, programmers can focus on the interactions between the platform and the context sources, and the management of all the outputs that are triggered by the rules in order to develop all the business logic of the service to be implemented. The evaluation results show that the involvement of non-technical users is possible and that it is beneficial in order to improve the development process.

7 References

1. Baldauf, M., Dustdar, S. and Rosenberg, F.: A survey on context-aware systems. International Journal of Ad Hoc and Ubiquitous Computing, vol. 2, p. 263 (2007)
2. Green, D., and DiCaterino, A.: A Survey of System Development Process Models. Center for Technology in Government University, Albany (1998)
3. Henricksen, K., and Indulska, J.: Developing Context-Aware Pervasive Computing Applications : Models and Approach. Pervasive and Mobile Computing, 2(1), 37-64 (2006)
4. Hirschfeld, R., and Costanza, P.: Context-oriented Programming. Journal of Object Technology, 7(3), 125-151 (2008)
5. Bardram, J.E.: The Java Context Awareness Framework (JCAF) – A Service Infrastructure and Programming Framework for Context-Aware Applications. Proceedings of the 3rd International Conference on Pervasive Computing, pp. 98-115 (2005)
6. Gu, T., Pung, H. and Zhang, D.: A service-oriented middleware for building context-aware services. Journal of Network and Computer Applications, vol. 28, pp. 1-18 (2005)
7. Sohn, T. and Dey, A.: iCAP: An Informal Tool for Interactive Prototyping of Context-Aware Applications, Extended abstracts of CHI, pp.974-975 (2003)
8. Cassou, D., Bruneau, J., & Consel, C.: A tool suite to prototype pervasive computing applications. 2010 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), 820-822 (2010)
9. Guo, B., Zhang, D. and Imai, M.: Toward a cooperative programming framework for context-aware applications, Personal and Ubiquitous Computing, 15(3), 221-233 (2012)
10. Dey, A., Abowd, G., and Salber, D.: A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. Human-Computer Interaction, vol. 16, pp. 97-166 (2001)
11. Yau, S. S., and Huang, D.: Mobile Middleware for Situation-Aware Service Discovery and Coordination. In P. B. and A. Corradi (Ed.), Handbook of Mobile Middleware (2006)
12. Allen, J.: Maintaining knowledge about temporal intervals, Communications of the ACM, 26(11), 832-843 (1983)
13. Ipina, D., and Katsiri, E.: An ECA Rule-Matching Service for Simpler Development of Reactive Applications. Published as a supplement to the Proc. of Middleware 2001 at IEEE Distributed Systems Online, Vol. 2, No. 7 (2001)
14. Davis, F.: Perceived Usefulness, Perceived Ease of Use, and User Acceptance of lnformation Technology, MIS Quarterly, 13(3), pp. 318-340 (1989)