Modeling the monitoring and adaptation of context-sensitive systems

Jéssyka Vilela, Jaelson Castro

Centro de Informática
Universidade Federal de Pernambuco
Recife, Brazil
{jffv, jbc}@cin.ufpe.br

Abstract. [Context] Context-sensitive systems (CSS) must detect variations in their operating context and adapt their behavior in response to such variations. Hence, their development requires the support of appropriate methods of software engineering. [Objective] This paper describes the activities of the GO2S systematic process to specify the adaptation and monitoring as well as the flow expressions of CSS. [Results] This process guides the software engineer to model the adaptive behavior through contextual design goal models and contextual refinements. [Conclusion] These models explicitly capture what changes in the environment and in the system to be monitored, what to adapt, when to adapt and how to adapt. We illustrate our proposal by applying it to the smart home exemplar.

Keywords: Adaptation, Context, Design Goal Model, Monitoring, Behavior.

1 Introduction

Context-Sensitive Systems use context to provide services and relevant information to their users. They are flexible, able to act autonomously on behalf of users and dynamically adapt their behavior. Hence, these systems must have the following characteristics: monitoring, awareness and adaptability [1]. Considering the inherent complexity and variability of context-sensitive applications, their development requires the support of appropriate methods of software engineering.

The specification of adaptive behavior is an issue addressed with different perspectives. The contexts are used to represent and analyze the variations in i* models resulting from the domain variability in [10]. The work of [8] supports the design and runtime execution of adaptive software systems both at a requirements and architectural level. Another work [2] describes a systematic methodology to design adaptive software systems. Finally, a method to derive the adaptive behavior of Dynamically Adaptive Systems (DAS) from a set of i*models is presented in [7].

In previous works [4][5], we proposed the GO2S (GOals to Statecharts) process, a systematic approach for deriving the behavior (expressed in statecharts) of context-sensitive systems, from requirements models (described as goal models).

In this paper, we detail two critical sub-process of the GO2S process: the modeling of the monitoring and adaptation as well as the specification of the flow expressions. These sub-process define systematic methods to model the system's adaptation and monitoring through the elements of an extended (contextual) design goal model (DGM) and flow expressions [6]. We illustrate our proposal by applying it to the smart home exemplar.

The remainder of this paper is organized as follows. In Section 2, we present our approach to perform the specification of the adaptation and monitoring as well as the behavior of CSS following the GO2S process. Section 3 discusses the contributions of this work and present the venues for future works.

2 Our proposal

The GO2S is an iterative process centered on the incremental refinement of a goal model, obtaining different views of the system (design, contextual, behavioral). The GO2S process consists of six sub-processes: 1) *Construction of design goal model; 2) Specification of contextual variation points; 3) Specification of monitoring and adaptation; 4) Specification of flow expressions, 5) Statechart derivation and refinement* and *6) Prioritization of variants*. In the next subsections, we detail how to specify the adaptation and monitoring (sub-process 3) as well as the flow expressions (sub-process 4) of CSS. The GO2S process assumes that the requirements elicitation and analysis activities were previously performed and a goal model was generated. It is out of scope of this paper to present and discuss all activities of the GO2S process.

2.1 Modeling the adaptation and monitoring of context-sensitive systems

We propose that the specification of the adaptation and monitoring of context-sensitive systems (sub-process 3 of the GO2S process) is performed through refinements in the design goal model [2] extended with contextual annotations [3] which we call contextual design goal model.

Accordingly, in this sub-process we add adaptation design tasks in the contextual DGM. These tasks are required for the adaptation of each requirement that needed to be monitored. Then, they are refined through tasks in AND/OR decompositions that represent the adaptation strategies. The activities required for the modeling the adaptation and monitoring are presented in Fig. 1.

The input of this sub-process is the contextual design goal model that is used by the software engineer to define the critical requirements that requires adaptation. The next activity is the representation of the adaptation management, which we propose to perform through the following activities:

1. **Add a new design task in the root node for adaptation management** (This activity is necessary when the system requires more than one adaptation).
2. **Add design tasks in the parent node previously created for the management of each requirement that must be monitored and adapted** (*ex: Manage gas leak (t3)* in Fig. 2).

3. **Add design tasks to represent the adaptation strategies for each monitored requirement** (ex: *Turn off the oven (t1)* and *Call fire department (t2)* for the task *Manage gas leak (t3)* in Fig. 2).

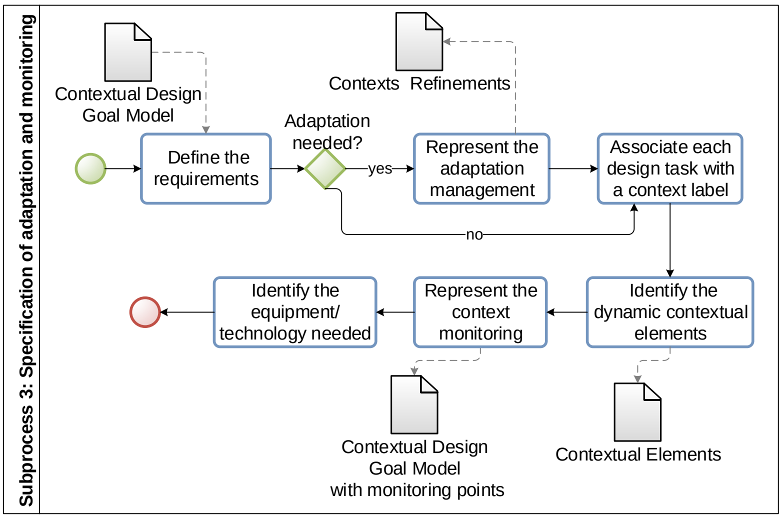

Fig. 1. Activities of specification of adaptation and monitoring sub-process.

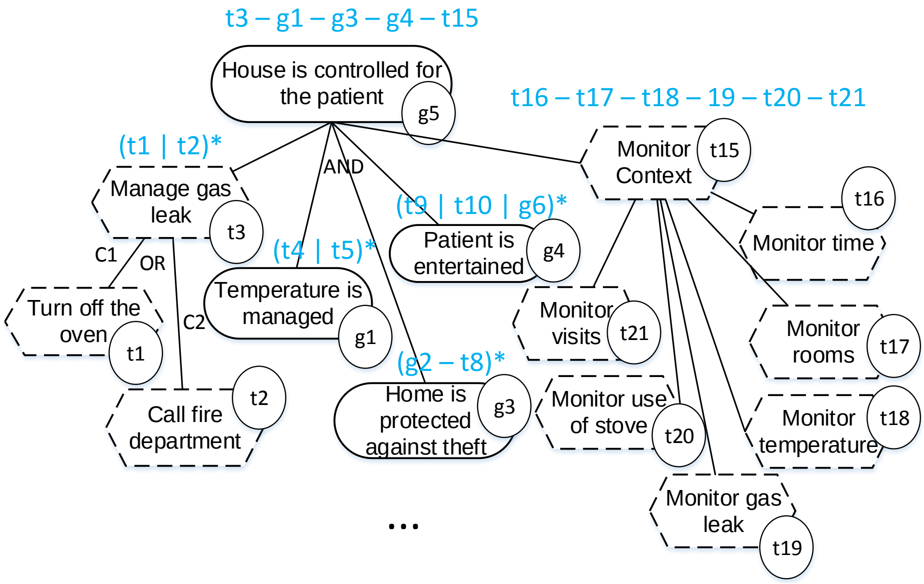

Fig. 2. Excerpt of the behavioral contextual DGM of smart home example.

Note that we should add at least two adaptation design tasks since the variants are the cornerstone for adaptability, a system with only one variant cannot be adaptable.

After the identification of the tasks necessary for the system adaptation, the next activity is to associate each adaptation design task with a context label since these tasks will executed only in certain contexts. In the smart home example, the *Turn off the oven (t1)* design task will be executed when the context C1 holds (*the patient finished using the oven*) and the *Call fire department (t2)* design task will be executed when the context C2 holds (*a gas leak is detected*).

The next step is the identification of the dynamic contextual elements. The dynamic contextual elements are the properties of real-world presented in the facts of context refinements that change their values dynamically. Therefore, the changes in the contextual elements imply in changes in the system context. In our running example, the dynamic contextual elements are the time, rooms, temperature, gas leak, use of the stove, and visits for the patient.

The next activity corresponds to the representation of the context monitoring. Accordingly, we propose the following activities in order to achieve this:

1. **Add a new design task in the root node** (*ex: Monitor Context (t15)* in Fig. 2).
2. **Add design tasks to monitor each dynamic contextual element** *(ex: Monitor time (t16), Monitor rooms (t17), Monitor temperature (t18), Monitor gas leak (t19), Monitor use of the stove (t20)* and *Monitor visits (t21)* in Fig. 2).

We propose to add the adaptation and monitoring activities in the root node since we want to improve the system's modularity and separation of concerns. Accordingly, the related design tasks will be executed concurrently with the system's requirements.

The last activity of this sub-process is the specification of the equipments/technology necessary for monitoring the contexts. In the smart home, the technologies needed are some mechanism to information storage and different types of sensors (presence, temperature, gas leak, stove and luminosity sensors). The outputs of the sub-process 3 of GO2S are the contextual design goal model refined and the contexts refinements.

Having defined the adaptation strategies and the contextual elements that need to be monitored, we can now move on to specify the order of execution of tasks and goals. For this, we rely on flow expressions. This sub-process is described in the next section.

2.2 Modeling the behavior of context-sensitive systems

The goal of the sub-process 4 (Specification of flow expressions) of the GO2S process is to refine the contextual DGM with flow expressions. Flow expressions are a set of enrichments to a goal model that allow specification of the runtime behavior through the execution order of its elements [2]. These expressions are used in the GO2S process as an intermediary model in order to derive the statechart [4-5].

The input of this sub-process is the contextual DGM previously obtained. The first activity is to assign an identification (ID) to each goal and task in the model. Their identification is necessary for reference in the flow expression later. *Gi* was used as ID for goals and *Ti* for tasks and design tasks where *i* is the number of the task.

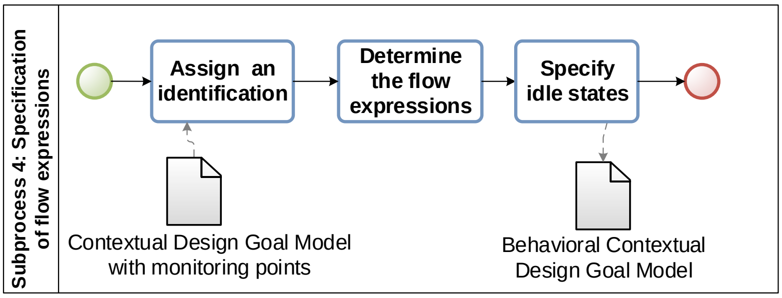

Fig. 3.Activities of Specification of flow expressions sub-process.

After the IDs assignment, the next activity is to define the flow expression for each parent node which describes the behavior of its children elements using the symbols proposed by [2]. The strategy of specifying the children behavior of a parent node can be bottom-up or top-down, the result will be the same. Thereafter, when we reach the root goal, we have the flow expression from the entire system. The resulting flow expressions should be annotated in the contextual DGM as demonstrated in Fig. 2.

A common practice when creating statecharts is to use intermediate states as a point where the system is idle, waiting for some input, such as input selection by the user or for a context to hold. Considering how frequently these states appear, and aiming to reduce visual pollution in the behavioral contextual DGM, such states must be inserted directly in the flow expressions identified as iX, where X is an integer.

The output of this activity is the behavioral contextual DGM. It is the contextual design goal model annotated with flow expressions. This model can represent in unified way all the views developed in the GO2S (contextual, design and behavioral).

3 Ongoing and Future Work

We conducted a controlled experiment in order to evaluate our process. This study was performed using 18 undergraduate and graduate students enrolled in a requirements engineering course divided into two groups with nine subjects each. Each subject of the first group constructed a statechart of the smart home system following the GO2S process (the GO2S group) and each subject of the second group built/developed a statechart without guidance (the control group).

The experiment results are encouraging since the structural complexity of the experimental group was lower and the mean of behavioral similarity was higher than control group. Besides, the subjects agreed that the GO2S process is easy to use indicating that it is understandable. The results of this experiment can be found at [12].

While statecharts are the industry standard and provide an intuitive representation of behavior models, formal analysis is limited and difficult. Hence, we are currently working on an approach to analyze properties of the statecharts generated with the GO2S process. Moreover, we are also investigating the contributions of using ontologies to the verification of statecharts considering their empirical benefits for requirements engineering identified in a previous systematic literature review [9].

We also expect to develop a case tool to support the process. It should be used to generate the different views (design, contextual and behavioral) of our process. Besides, it is important to derive systematically the other architectural views of CSS. The structural view of context-sensitive systems was already addressed in the work of [11] and the GO2S process addressed the behavioral view. The other views can be incorporated in our process in order to obtain a complete architecture specification.

References

1. Klein, C. et al. A Survey of Context Adaptation in Autonomic Computing. In: 4th International Conference on Autonomic and autonomous Systems (ICAS), 2008. pp. 106–111.
2. Pimentel, J. et al. From requirements to statecharts via design refinement. In: proceedings of the 29th SAC, 2014, pp. 995–1000.
3. Ali, R.; Dalpiaz, F.; Giorgini, P. A goal-based framework for contextual requirements modeling and analysis. Requirements Engineering, v. 15, n. 4, 2010, pp. 439-458.
4. Vilela, J.; Castro, J.; Pimentel; J.; Soares, M.; Lima, P.; Lucena, M. Deriving the behavior of context-sensitive systems from contextual goal models. In: Proceedings of the 30th SAC, 2015.
5. Vilela, J.; Castro, J.; Pimentel; J.; Lima. On the behavior of context-sensitive systems. In proceeding of 18th Workshop on Requirements Engineering (WER), 2015.
6. Boudaa, B.; Hammoudi, S.; Chikh, M.A. ODM-based modeling for user-centered context-aware mobile applications. IN: 3rd International Conference on Information Technology and e-Services (ICITeS), 2013, pp. 1-7.
7. Welsh, K. and Sawyer, P. Deriving Adaptive Behaviour from i* models. In: proceedings of the 4th International i* Workshop (istar 2010). pp. 98-102.
8. Pimentel, J.; Angelopoulos, K.; Souza, V. E. S.; Mylopoulos, J.; Castro J. From Requirements to Architectures for Better Adaptive Software Systems. In: proceedings of the 6th International i* Workshop (iStar 2013). pp. 91-96.
9. Dermeval, D.; Vilela, J.; Bittencourt, Ig.; Castro, J. Isotani, S.; Brito, P.; Silva, A. Applications of ontologies in requirements engineering: a systematic review of the literature. Requirements Engineering, 2015, pp. 1-33.
10. Lapouchnian A. and Mylopoulos, J. In: proceedings of the 5th International i* Workshop (iStar 2011). pp. 96-101.
11. Pimentel. J. et al. Deriving software architectural models from requirements models for adaptive systems: the stream-a approach. Requirements Engineering, v.17, n.4, 2012, pp.259–281.
12. J. F. F. Vilela. A systematic process to derive the behavior of context-sensitive systems from requirements models. MSC Dissertation, Universidade Federal de Pernambuco, Centro de Informática, 2015. Available at: http://www.cin.ufpe.br/~ler/supplement/istar2015/.