# Experience using Processes for Pervasive Applications

Stephan Urbanski, Marcus Handte, Gregor Schiele and Kurt Rothermel

Distributed Systems Department
Institute of Parallel and Distributed Systems, Universität Stuttgart
Universitätsstr. 38, 70569 Stuttgart
{vorname.nachname}@ipvs.uni-stuttgart.de

**Abstract:** Pervasive Computing envisions seamless computing support for users while performing their everyday tasks. Many of these tasks are hard to capture by the notion of a single application as their requirements on computing support vary over time. To ease the development of applications for such tasks, we have designed an application model for process-oriented pervasive applications, called Sentient Processes, and a supporting middleware system implementing the flow control for processes. In this paper, we show the capabilities of the Sentient Process model to capture everyday user tasks by the example of a process supporting a student in managing his classes during a term.

## 1 Introduction

Pervasive Computing aims at supporting users while performing their everyday tasks. Regarding the temporal aspect of tasks, we can distinguish two different kinds. The first kind is short-lived tasks that usually entail a single supporting application. An example for such a task is conducting a presentation in a smart environment [HE02]. The second kind is long-lived tasks that last over a long period and contain several sub-tasks. As an example, consider a student that takes a certain set of courses. During the term, he visits lectures and schedules meetings where he displays the lecture slides at need or he prepares the next lecture by reading relevant research papers. Such a task lasts a whole term, includes several repetitive sub-tasks, and follows a characteristic flow.

In the past, researchers have built a number of pervasive computing infrastructures that focus on various aspects such as providing transparent support for mobile devices or bridging device heterogeneity. However, these infrastructures typically focus on short-lived tasks [SO02, RO02, GR04, BE04] and do not provide abstractions that ease the development of long-lived ones. As a result, developers must face the complexity of combining applications for short-lived tasks to integrated applications for long-lived tasks without specialized tool support.

With Sentient Processes (SP) [SU06], our process-based pervasive application model, we concentrate specifically on providing an adequate tool for developing pervasive applications that support long-lived tasks. To do so, we build upon existing infrastructures for short-lived tasks. In essence, SP combines concepts from workflow systems for capturing business processes [LE00] and concepts of context-sensitive and context-adaptive applications. In the following, we discuss how SP can be utilized to capture a student's task of taking courses in a process. We then describe the experience gained from modelling this task and discuss the benefits and possible improvements.

We structure the remainder of the paper as follows. Next, we present the SP application model and the supporting infrastructure. In Chapter 3, we introduce the exemplary task and its SP representation. The lessons we learned from designing and executing this process using our infrastructure are discussed in Chapter 4. The final Chapter 5 summarizes the findings and concludes the paper.

## 2 Sentient Processes

We developed SP to simplify the development of long-lived, process-oriented pervasive applications. The SP project consists of an application model based on a high-level process description and a runtime environment for controlling the process execution.

The SP model represents user tasks as a process. The process is subdivided into a flow of steps. The granularity of the steps of a process is governed by their resource requirements. Each step should be atomic in respect to its resource requirements, allowing the system to allocate all necessary resources at the beginning of a step and releasing them once the step finishes. Typically, a step maps on a unit of execution, e.g., a single service or an ordinary application. We use abstract service descriptions for binding steps to units of execution dynamically. The long-lividness, the ever-changing environments and the everyday characteristics of user tasks require considering the user's situation while executing a task. The SP infrastructure enables context sensitive task execution by modelling context situations and specifying actions to be taken in these situations. Possible actions are for example to suspend the execution of a step and to resume it later. Figure 1 shows the states of a step (Exec, Paused, Error) and actions (abort, activation, finish, pause, resume, retry).

Once the developer has specified the individual steps of a process, the flow of control between these steps can be modelled using so-called dependencies. A dependency describes the conditions that must be fulfilled for a step to become active. This includes other steps that need to be finished and optionally a context situation. Each step may have multiple dependencies, thus a directed graph can represent the process. The start of a process is given by one specific start step designated by the application developer. The process is finished once a step of the group of end steps has been reached or once the process is aborted due to an error.

The SP runtime infrastructure controls the concurrent flow of all currently executed processes, keeps track of their dependencies and maps steps to applications available in the environment. To execute these applications, the infrastructure uses existing resource platforms available in the environment. Possible examples for such resource platforms are Operating Systems, e.g., Windows, but also Pervasive Computing infrastructures for short-lived applications like PCOM. Furthermore, the runtime infrastructure interacts with external context services, e.g., [GR05], for obtaining context information.

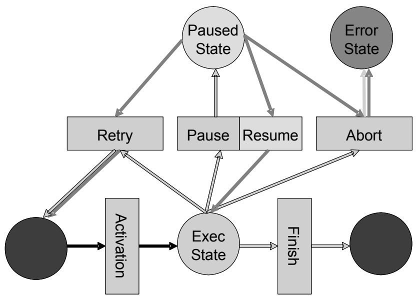

**Figure 1** - States of a step

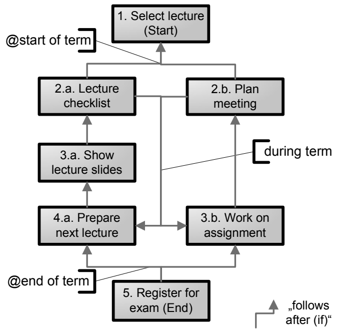

**Figure 2** - Lecture Support Task

The externalization of the context service and resource platform allows us to bridge heterogeneous environments. At runtime, the infrastructure is able to access newly available services and change the mapping of running processes, thus adapting them to the new environment.

We implemented the SP runtime infrastructure on top of the BASE Middleware, our event-based communication infrastructure for Pervasive Computing.

## 3 Exemplary Task

A typical everyday's task for a student is taking courses. For each lecture of a course, the student typically takes lecture material along, attends the event, meets with fellow students to work on assignments, and prepares the next lecture by reviewing the lecture slides and reading the course book. These sub-tasks are repeated during a term for each lecture.

This task apparently exhibits flow characteristics and falls into the class of long-lived tasks. To apply a process-oriented application model we must first identify the individual steps of the task. As we use SP, we apply a resource-governed granularity to isolate steps. In our example, we want to support the student in a number of steps, each time making best use of resources available in the user environment.

During the lectures, we provide an application for viewing lecture slides. It uses the largest available display accessible by the user in his current environment. When students meet to work on their assignments, the collaborating students are provided with an ad-hoc connecting shared white-board application. At home, the student again takes advantage of the viewing application for pre-reading slides. Along with the support for those individual steps, the task of the user involves keeping track of his schedule, reminding him of appointments and things to take along for the lectures and finally arranging the meetings with his fellows for working on assignments.

The steps of the lecture support task exhibit a distinct ordering with dependencies to time and the location of the user. The viewing application for lecture slides should only be executed during lectures. Similarly, the white-board application should only during the corresponding meetings. Re-reading slides is bound first on a suitable location like home or the library and secondly should happen after the class and before the next class. Figure 2 depicts the complete task structure and shows several other, rather obvious dependencies. The described conditions and connections between the steps can be modelled using the SP notion of dependency. For completing the task description, we introduce two additional steps, the starting, and endpoint of the task.

The process presents itself to the student as follows. For each course, the student wants to attend he instantiates one process. In the first step of the process, the student picks the course he wants to attend (Step 1). Then the actual process cycle starts. The application reminds the student of the things to take along before each lecture (Step 2.a.). During the lecture, the slides are presented to him (Step 3.a.). After class and being in an appropriate location, a re-reading and pre-reading of lecture slides is suggested to the user (Step 4.a.). In parallel to the class track, the user is supported first by scheduling a meeting with fellow students (Step 2.b.). At the scheduled meeting, the user can start the white-board application (Step 3.b.). If both tracks are finished, the cycle starts over unless the term is finished. In that case, the process concludes with helping the student to register for his exam in that course (Step 5).

## 4 Experiences

In the previous chapter, we have captured the course task in a process-oriented application model. Aside from demonstrating the feasibility of the approach, the modelling extends the capabilities for supporting the user. Due to the available information on spatial and temporal dependencies as well as execution order of the steps in a task, we can now propose steps the user should or could take in his current situation. It also allows analyzing the interconnections between various tasks of a user. Given a set of utility values, for example execution time, we could compute an overall utility for each possible step at runtime and show it to the user.

In contrast to business processes, our example of SP shows that the student may omit steps without breaking the process. We achieve this flexibility by two properties of SP. First, the step to be executed is always selected by the user and not by the system. Secondly, we can model omitting steps using context conditions, declaring for example the step of showing lecture slides as completed once the class period is over.

Another capability to support the user is allowing him to define his own processes. While typical application development exceeds the knowledge of an average user, defining a process might be feasible. This was indicated by the study of Knoll et. al. [KN06]. In our example, this would enable the user to create variants of the task adapted to individual courses. Currently we are developing a visual tool for designing processes that combine existing applications.

Our initial experiences with SP indicate that they are a valuable extension to our existing Pervasive Computing infrastructure that focuses on short-lived applications. The architecture and extensibility of the runtime engine towards different infrastructures enables SP to extend them with better support for long-lived applications.

## 5 Conclusion

Many user tasks in Pervasive Computing environments are long-lived and exhibit flow characteristics as shown by the exemplary task. We have presented and discussed the model of Sentient Processes to capture the properties of such tasks. The transformation of a flow-oriented user task into a Sentient Process is shown by the example of the course task resulting in the user taking advantage of guidance during task execution. The separation of flow and steps allows separating the design of task flow from implementing applications. We hope that this, in turn, will eventually enable the user to design his own tasks building on third-party applications using simple visual tools.

## References

[BE04] Becker, C.; Handte, M.; Schiele, G.: PCOM – A Component System for Pervasive Computing. In International Conference on Pervasive Computing and Communications (PERCOM) 2004, Orlando, USA, 2004

[GR04] Grimm, R.; Davis, J.; Lemar, E.; MacBeth, A.; Swanson, S.; Anderson, T.; Bershad, B.; Borriello, G.; Gribble, S.; Wetherall, D.: System support for pervasive applications. In ACM Transactions on Computer Systems, Novemer 2004; pp. 421-486

[GR05] Grossmann, M.; Bauer, M.; Hönle, N.; Käppeler, U.-P.; Nicklas, D.; Schwarz, T.: Efficiently Managing Context Information for Large-scale Scenarios. In Proceedings of the 3rd IEEE Conference on Pervasive Computing and Communications (PERCOM) 2005, Hawaii, USA, 2005

[HE02] Hess, C. K.; Román, M.; Campbell, R. H.: Building Applications for Ubiquitous Computing Environments. In International Conference on Pervasive Computing, Zurich, Switzerland, August 26-28, 2002; pp. 16-29.

[HO99] Hohl, F.; Kubach, U.; Leonhardi, A.; Rothermel, K.; Schwehm, M.: Next Century Challenges: Nexus - An Open Global Infrastructure for Spatial-Aware Applications. In Proceedings of the Fifth Annual International Conference on Mobile Computing and Networking (MobiCom '99)

[KN06] Knoll, M.; Weis, T.; Ulbrich, A.; Brändle, A.: Scripting your Home. In Proceedings of the 2nd International Workshop on Location- and Context-Awareness (LoCA 2006), Dublin, Ireland, May 2006

[LE00] Leymann, F.; Roller, D.: Production Workflow. Prentice Hall, Inc., NJ (2000).

[RO02] Román, M.; Hess, C. K.; Cerqueira, R.; Ranganathan, A.; Campbell, R. H.; Nahrstedt, K.: Gaia: A Middleware Infrastructure to Enable Active Spaces. In IEEE Pervasive Computing, 2002; pp. 74-83.

[SO02] Sousa, J. P.; Garlan, D.: Aura: an Architectural Framework for User Mobility in Ubiquitous Computing Environments. In Proceedings of the 3rd Working IEEE/IFIP Conference on Software Architecture, August 2002; pp. 29-43.

[SU06] Urbanski, S.; Becker, C.; Rothermel, K.: Sentient Processes – Process-based Applications in Pervasive Computing. In Proceedings of the 4th International Conference on Pervasive Computing, Pisa, Italy, 2006