Fraunhofer Institut
Experimentelles
Software Engineering

From Requirements Engineering to Knowledge Engineering: Challenges in Adaptive Systems

Authors:
Michael Eisenbarth
Mathias Grund
Klaus Schmid

Accepted for Publication in
SOCCER 05 Workshop,
Requirements Engineering Conference,
2005, Paris

Partially supported by the projects BelAmI (Bilateral German Hungarian Research Collaboration on Ambient Intelligence Systems) and ASG (FP6-IST-004617).

IESE-Report No. 118.05/E
Version 1.0
August 30, 2005

A publication by Fraunhofer IESE

Fraunhofer IESE is an institute of the Fraunhofer Gesellschaft.
The institute transfers innovative software development techniques, methods and tools into industrial practice, assists companies in building software competencies customized to their needs, and helps them to establish a competitive market position.

Fraunhofer IESE is directed by
Prof. Dr. Dieter Rombach (Executive Director)
Prof. Dr. Peter Liggesmeyer (Director)
Fraunhofer-Platz 1
67663 Kaiserslautern

Abstract

New technologies like service-based computing or ambient intelligence aim at systems that adapt their behavior at runtime and integrate functionalities dynamically to satisfy end-user needs. In this paper, we will argue that the way we do Requirements Engineering will need to change in order to support the necessary adaptability. We will describe the nature of the expected change and will illustrate our concepts with an example.

Keywords: ambient intelligence, requirements engineering, knowledge engineering, adaptive system, BelAmI, ASG

Table of Contents

1 Motivation

Many current SE paradigms like Service-Orientation or Ambient Intelligence focus on increasing levels of flexibility and adaptivity of the final system. So far, however, the impact this shift has on requirements engineering has not yet been sufficiently understood [1]. In this paper, we argue the increasing focus on adaptive systems will lead to a significant shift from traditional requirements engineering towards integration with knowledge engineering activities.

We define – for the purpose of our work – adaptive systems as systems that are able to adapt their behavior based on the availability of base functionality to optimally serve the goals of the user. Such adaptation can be triggered either by the absence or by the unexpected presence of functionality. Further, the behavior adaptation can be focused on providing the same problem-solving capabilities to the end-user with different qualities (e.g., response time) or by providing functionally different end-user services. In this paper we will focus on the provision of functionally different end-user services and the addition of unexpected services.

There are two very prominent technologies currently under discussion, which are most relevant to this: ambient intelligence and service-oriented systems. (Of course, both are related to each other as service-oriented technologies are seen as a key technology for implementing ambient intelligence systems.)

The ambient scenario is driven from the idea that coordinated assemblies of small devices would provide intelligent services to users in their vicinity. Of course, these devices must be regarded as characterized by the services they are able to provide and thus, we can see an ambient environment as a service-oriented infrastructure, in which services must be combined in a user-centric manner. We pursue these questions in the context of the BelAmI project [2].

We can directly compare this situation with a service grid infrastructure (except for location-awareness). As new services can be deployed in the grid infrastructure at any time, it is a key question, whether these services can be exploited by applications build on top of the infrastructure. This is the focus of the Adaptive Service Grid (ASG) project. This project addresses in particular the relationship between service customers and service providers. It is the explicit goal of this project to compose elementary services into complex processes, in order to create new application services on demand [3].

This paper is structured as follows. Chapter 2 gives on overview of the development context of adaptive, service-oriented systems. Chapter 3 discusses the

challenges for requirements engineering arising from adaptive systems. Chapter 4 exemplifies the challenges with a travel planner system. Finally, in Chapter 5, we give a conclusion and discuss future work.

2 Description of development context

Especially in the context of an adaptive service providing system with a dynamically changing set of resource providers and consumers, a service can be situated anywhere in the world and appear and disappear anytime during the usage of such a system. In order to be visible and accessible from everywhere at any time, service providers in a grid environment generally publish a service as a service interface providing service methods. Similarly, ambient devices need to make their services known to a service broker in an ambient environment. Web services are a typical form of this type of service provision, where web-based protocols and mechanisms are used by software elements to enable a standardized communication with these services.

This situation, in which systems need to dynamically identify relevant services in their environment, leads to different viewpoints in requirements engineering. The distinction of an end-user and a developer oriented view on requirements engineering for ambient systems is presented in [4]. In addition to these two views, a third perspective on requirement engineering is raised by the need to determine which existing services can fulfill an end-user request. Figure 1 illustrates the idea of a service-oriented requirements engineering methodology [5] that addresses these three views on requirements.

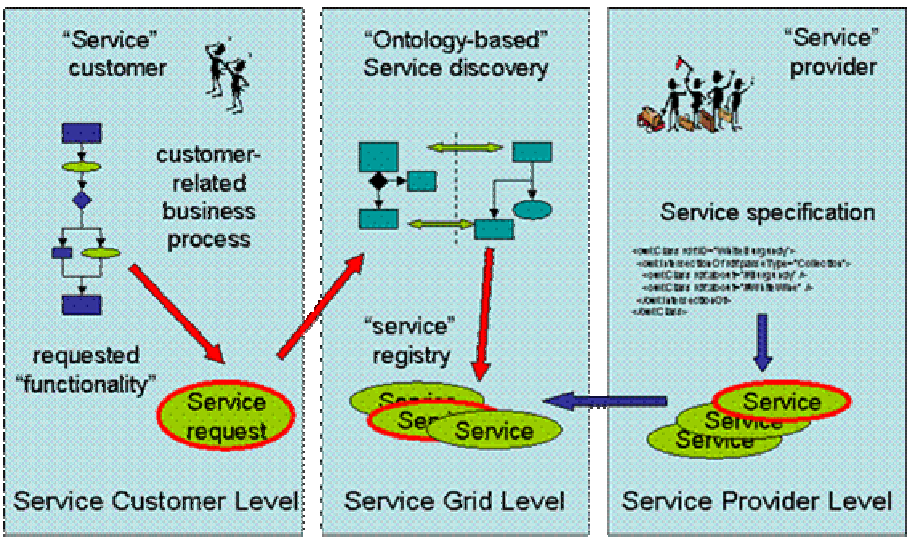

Figure 1 Service requirement levels

The service customer requires an explicit modeling of the current customer needs and constraints that affect the type and outcome of the requested services. At the service infrastructure level the need for an information model de-

scribing semantic information about the registered services is prominent. This semantic model can then be used to identify appropriate services for the specified service request. Finally, the service providers are required to provide service specifications of their services, describing functional and non-functional aspects as well as semantic usage knowledge.

One possible solution to achieve the systematic discovery of existing services and composition of new services by semantic modeling is the usage of ontologies or knowledge models. Based on the semantic specifications of requested services by service customers and the semantic specifications of the registered services by service providers, a service broker has to discover appropriate services, compose complex processes and – if required – generate software to create new application services on demand. Subsequently, application services will be provided through the underlying system infrastructure based on adaptive process enactment technology.

In the remainder of this paper, we will focus on the service customer, i.e. the end-user view on an adaptive system, to explain the impact of the capability to specify requirements for such kind of systems.

3 Requirements Engineering Challenges

A major characteristic of adaptive systems is the capability to provide new functionality during runtime of the system. This leads to new challenges for requirements engineering as current techniques are not sufficiently capable of supporting this degree of adaptivity of available functionality.

Use cases for example are used to describe how end-users will use the system. They describe a task or a series of tasks that users will accomplish using the software, and include the responses of the software to user actions. Use cases may be included in the Software Requirements Document (SRD) as a way of specifying the end-users' expected use of the software. They are used to validate understanding, and to identify normal and special use situations. For adaptive systems that can provide new functionality during runtime, it is obvious that a standard use case cannot specify these uncertain tasks or possible usage scenarios.

Generic use cases, which can be instantiated for specific situations, are capable of describing generic functionality of an adaptive system. But further context information and specific application domain knowledge is required to adapt these use cases to specific situations and therefore the system to a specific usage context and application scenario that provides new unprecedented services. This context information or domain knowledge about a specific application scenario must be specified and modelled in a systematic and extensible form.

Ontologies [6] are one way to define formal semantics for these additional kinds of information and knowledge, thus enabling automatic information processing. Ontologies are formal models of a specific domain that support the communication between human and computer based actors and therefore enable automatic adaptation of services or system components based on the current context.

The shift in the type of information contained in a requirements specification, which is needed for adaptive systems, is shown in the above figure 2.

In non-adaptive systems, the user-specified system requirements and hardware constraints are the most prominent part of a requirement specification as the systems functionality and usage context barely change during runtime. In the case of adaptive systems that can provide new functionality and services during runtime the above-mentioned specification of domain knowledge is required.

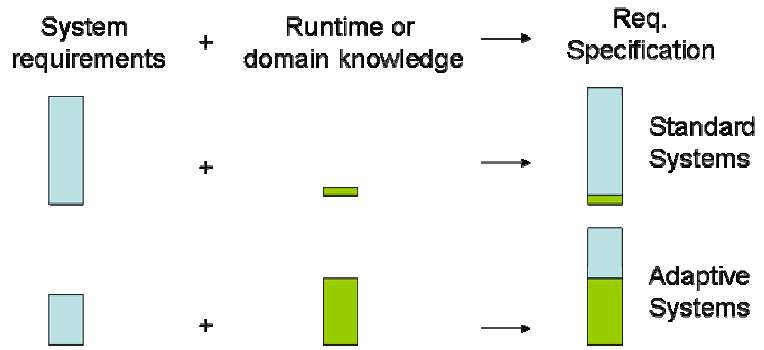

Figure 2 Relationship between development and runtime information

Supporting the evaluation of possible solutions is another important challenge for requirements engineering posed by adaptive systems. An adaptive system broker needs to explore alternative system proposals by automatically discover available services and to evaluate the appropriateness of a specific solution for a certain user request. Explicit goal modelling is one way to support the specification of user-oriented service requests. Goals provide the basis to evaluate the appropriateness and usefulness of newly available solutions to the end user. In [7][8], an approach for creating formalized goal-oriented specifications and evaluation of requirements is presented. The hierarchical decomposition of high-level goals into sub-goals helps to identify whether a service can fulfil a sub-goal and if the composition of multiple services is appropriate to fulfil the corresponding high-level goal.

4 Example for adaptive systems

We will now illustrate the envisioned shift towards knowledge representation with an example: a travel planner system (TPS).

The travel planner system allows a user to plan her travel. Let us first assume a rather traditional setting: all available services are known beforehand. There will be services available for: booking trains, planes, etc. Accommodation services will be available and so forth. Figure 3 shows an example of a use case that describes travel planning in such a situation.

On the other hand, in the case of an adaptive service framework, we will not be able to detail the use case in the same way, as we do not know whether the specific booking services will be available. Rather, the system will need to adapt its "flow of events" at runtime.

Use Case: Create Travel Plan (Traditional)
Actor: User
Goal: A travel plan is created, based on the services booking trains, planes and hotels.
Flow of Events:

1. The user enters point of departure and destination and the start and end time of the travel
2. The system displays the hotel screen
3. The user enters hotel rate
4. The system offers a list of available hotels
5. The user selects hotel
6. The system displays the travel screen
7. The user selects a means and class of transport
8. The system calculates valid routes which don't exceed maximum traveling time and costs
9. The system displays the list of valid travel plans

Figure 3 Use Case for traditional TPS

Figure 6 shows the resulting use case. In this case the details of the service interactions are decided at runtime. Thus, the necessary information must be available in a machine-processable form. A very basic ontology ([9] simplified) for this application is given in

Figure 4. As opposed to traditional ontologies we expect these to be built out of three parts:

- **Application-layer:** in our example, the information regarding the concept of a travel could be brought in by an application.
- **Infrastructure-layer:** The infrastructure (or grid) would need to make available base information that is relevant to services that can be deployed within the grid.
- **Service-layer:** Services can bring along some knowledge fragments which provide information specific to the service and sufficient to integrate it with the infrastructure layer (for example, a taxi service could provide the information that a taxi is a means of transportation). This is shown in
- Figure 4.

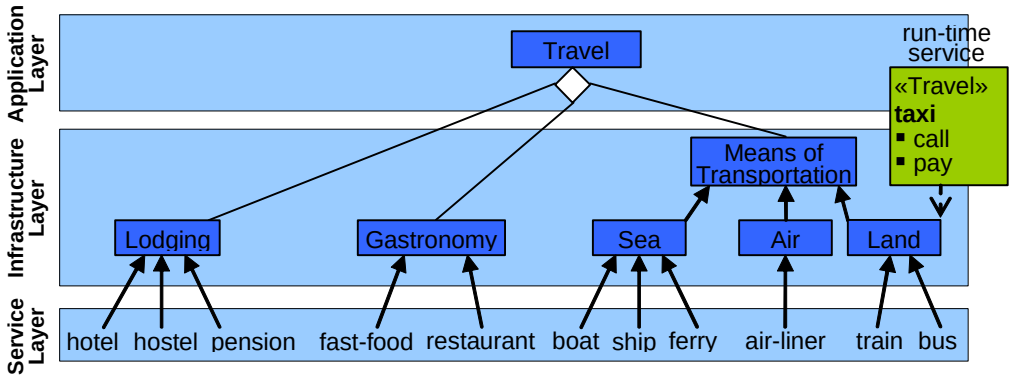

Figure 4 Domain model for adaptive TPS

In our example the final application will need to use the specific services available at runtime. Thus, the use case in Figure 6 will only capture the basic information like start/end date and destination. During runtime the system integrates available service, like train and bus as a transport service and hotel as accommodation service. That means, when describing the requirements for the system, it is not possible to say, which specific service will be available. During runtime there can be several services for transport and accommodation, services offering complete new functionality or not sufficient services. The system thus has to rely on the information available at runtime.

So far, we did not yet answer the question of how to use the existing knowledge to identify the necessary services at runtime: we bridge this gap by using approaches from the goal-oriented requirements engineering [7][8]. Ultimately, we need a high-level goal-model as part of the application. This - in combination with the underlying domain model - will provide the necessary information for the runtime assembly of the final application.

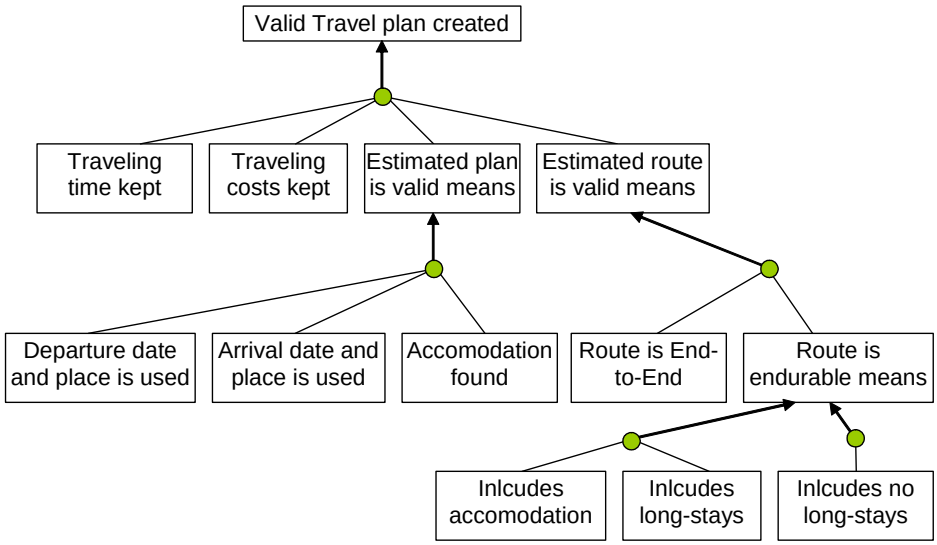

Figure 5 Goal Model for adaptive TPS

Figure 5 shows an example of such a goal-model. One should note a key difference between the initial development context of Lamsweerde's goal modeling and the context we are envisioning here: while the approach was initially developed as a construction-time activity, we envision here a use of this technique – based on further refinement and formalization as a (partial) runtime activity. This is of course closely related to approaches developed in the area of artificial intelligence as planning systems.

Use Case: Create Travel Plan (Adaptive)
Actor: User
Goal: A travel plan is created, based on the relevant services available at runtime.
Flow of Events:

1. The user enters point of departure and destination and the start and end time of the travel
2. The user enters maximum traveling time and costs
3. The system creates valid travel plans
   (Goal: Valid Travel plans created)v
4. The system presents the list of valid travel plans

Figure 6 Use Case for adaptive TPS

5 Conclusions and Further Work

Requirements Engineering for Adaptive Systems requires a serious shift in focus from collecting construction time knowledge to a combined approach that integrates construction time and runtime knowledge gathering. This is based on our assumption that adaptive systems will require a substantial amount of knowledge at runtime in order to perform the necessary adaptations in a manner deemed adequate by the end user.

We aim at further analyzing this relationship in order to derive an integrated methodology. We will do this in a demonstrator-driven manner based on our work on adaptive service grids (ASG) and ambient intelligence (BelAMI).

Acknowledgements

The work presented in this paper has been partially supported by the projects BelAmI and ASG. BelAmI (Bilateral German-Hungarian Research Collaboration on Ambient Intelligence Systems) is funded by German Federal Ministry of Education and Research (BMBF), Fraunhofer-Gesellschaft, and Ministry for Science, Education, Research and Culture (MWWFK) of Rheinland-Pfalz. ASG is funded by the European Union (FP6-IST-004617).

References

[1] Berry, D., et al., "The Four Levels of Requirements Engineering for and in Dynamic Adaptive Systems", Design and Evolution of Autonomic Application Software (DEAS'05), 2005.

[2] Project BelAmI, http://www.iese.fraunhofer.de/ belami/.

[3] Weske, M., et al. "Technical Annex 1 – Adaptive Service Grid, Description of Work", Proposal no. 004617, Integrated Project, Sixth Framework Programme, 2004.

[4] Schmid, K. "Requirements Engineering for Ambient Intelligence Systems: A Viewpoint", Workshop on Service-oriented Requirements Engineering, RE '04, 2004.

[5] Eisenbarth, M. et al., "Requirements Specification Survey", Adaptive Services Grid Deliverable D6.I-1, Integrated Project, Sixth Framework Programme, 2005.

[6] Gruber, T., "A Translation Approach to Portable Ontology Specifications", in: Knowledge Acquistion, Volume 5, pp. 199–220, 1993.

[7] Lamsweerde, A., "Goal-Oriented Requirements Engineering: Goal-Oriented Requirements Engineering: from System Objectives from System Objectives to UML Models to UML Models to Precise Software Specifications", ICSE '03 Tutorial, Portland, 2003.

[8] Lamsweerde, A., "Goal-Oriented Requirements Engineering: Goal-Oriented Requirements Engineering: A Roundtrip from Research to Practice", RE '04, 2004.

[9] Gordon, M, et. al., "Ontologies in a travel support system", in Proceedings of the Internet 2004 Conference, 2004.

Document Information

Title:	From Requirements Engineering to Knowledge Engineering: Challenges in Adaptive Systems
Date:	August 30, 2005
Report:	IESE-118.05/E
Status:	Final
Distribution:	Public