Context-Dependent and Self-Responsible Migration of Software Agents in Heterogeneous Environments

Dirk Bade
University of Hamburg, Department of Informatics
Distributed Systems and Information Systems Group

bade@informatik.uni-hamburg.de

Abstract: Software agents are often employed in distributed environments to cope with the various dynamical dimensions of such systems. Furthermore, the paradigm of agent-oriented software engineering is of special interest in the course of ubiquitous computing and the emerging mobility of users. One reason for this are the constituting characteristics of agents, that perfectly suit the accompanying demands of these trends. Hence, this paper introduces a new kind of adaptive, mobile application supporting the ”anywhere and anytime”-ability of mobile computing in general as well as the mobility of users and their adaptation to new computing environments. Therefore, a generic and extensible environment model, facing the heterogeneity of the infrastructure and entities within the environment, is presented in this paper. Further on, a mobility model for safe and efficient migration of agents is introduced. Using these models, agents may adaptively choose appropriate migration strategies at runtime by taking their current context into account.

1 Introduction

Since the early eighties agent-oriented software engineering gains more and more interest. Due to the offered higher level abstractions compared to e.g. the object-orientation, this paradigm is especially suited for the development of complex and distributed systems [LMSW05]. Another reason for this are the constituting characteristics of software agents like their ability to perceive the environment, to autonomously and proactively act in this environment and thereby to dynamically adapt to changes [WJ95, RN03]. Additionally, the notion of mobility is included in some definitions of agents, allowing them to migrate between different execution platforms and perform tasks in a suitable environment [FG96].

The ability to sense the environment combined with the ability to change the place of execution allows for dynamically choosing appropriate execution platforms, characterized e.g. by offered local resources. This feature is especially useful in the context of mobile computing, where agents are executed on resource constrained devices and hence may wish to temporarily migrate onto other platforms in order to process their tasks. This also allows agents to autonomously accompany their user by monitoring her behavior and migrating on a (mobile) device close to the user. This way, the user has access to the agent's data and services anytime and anywhere. Think of a mail-agent for example, retrieving

mails once a network resource is available and following it's user so that she can read and respond to mails no matter whether she is at her desktop computer at home or at work or with her mobile device on the way.

But having a personal agent following it's user to offer data and services is just one aspect. Since only a single instance of an application agent is needed, multiple installations, configurations and subsequent updates as well as manually synchronizing two or more instances can be avoided. This results in a higher convenience for the user as she can work with her application in a familiar setting and does not have to get used to new application settings on each device she is working with. Other examples include self-organizing service networks, where context-aware, mobile service agents are responsible for load-balancing and failure-recovery or applications for mobile ad-hoc networks (MANETs), in which unreliable and slow connections demand safe and efficient migration protocols that take the agent's current context into account.

Therefore, in this paper an environment model is presented in section 2, allowing agents to exchange information about entities in their surrounding. Furthermore, section 3 introduces a mobility model, which is used to specify the details of a migration process like e.g. the kinds of environmental events triggering the migration, the kind of migration (weak or strong), safety provisions (e.g. agent backups, error handling) and so on.

2 Environmental Awareness

Sensing the environment plays a key role in an agent's internal reasoning process to choose an appropriate action towards reaching one of it's goals. Such an action is often directed to some kind of entity within the environment, e.g. another agent to coordinate with, a database to access or a service to engage. In order to perceive entities, an agent either needs some kind of sensor or it must exchange knowledge with other agents. The received percepts then need to be integrated into an internal model of the environment for further application-dependent processing.

This work therefore proposes a generic and flexible environment model, which is on the one hand responsible for gathering information about locally and remotely accessible entities and on the other hand for offering this information to the user and other agents in an easy, intuitive and yet expressive way. Furthermore, the model should not only allow for entity look-up or discovery, but should also support awareness [McG00], i.e. the continuous integration of state changes triggered by any environmental events. Due to the heterogeneity of the entities in the environment, it must allow for integrating information about any kind of entity (e.g. hardware entities, remote platforms, available services and other agents) and must be extensible to incorporate application-dependent information (e.g. documents, business objects, etc.). Finally, the model should make use of existing standards for exchanging and representing knowledge, but also provide simple proprietary mechanisms as a least common denominator.

In order to meet these requirements, the proposed model defines a set of generic interfaces for a) gathering information about local entities, b) exchanging information with other agents about remote entities and c) representing information. Local sensors are responsi-

ble for gathering information about e.g. the hardware, the operating system and execution environment as well as locally offered services and other local agents. Discovery protocols (e.g. Jini, Bluetooth SDP, JXTA, etc.) may be used to distribute information within the network and hence to collect information about remote entities. Different representation- and query languages as well as optional transformation services allow for expressing knowledge, ranging from simple string-based to expressive logic-based languages (e.g. RDF, OWL). Additional support for ontologies not only yields to a common conceptualization of entities, but also eases the transformation of representations into one another and hence supports the interoperability of different protocols and mechanisms.

In the context of this work a prototype for the *Jadex agent system* [BPL05] has been developed. This prototype has been implemented as a dedicated service agent (called *resource facilitator*, RF) responsible for managing and updating the environment model. Once started, this component may use different mechanisms to initially find other remote RFs and to subscribe for specific news channels (e.g. news about offered services, known contacts, resource load, etc.). In the following, information is exchanged using one or more standard discovery protocols, which are adaptively chosen, depending on the current context. This way, a virtual (multi-hop) network is spanned, in which environmental information is actively distributed among the RFs. Being aware of the environment is furthermore the basis for context-dependent, event-driven migration, as described in the following section.

3 Context-Dependent and Self-Responsible Migration

Considering the process of agent migration, there are three questions to answer: when, where and how to migrate an agent? In general, one might say, an agent needs to migrate once it's current platform lacks a required resource, which is available elsewhere. This answers the first two questions. But how to migrate strongly depends on the current conditions, because there are different kinds of migration techniques. One has to decide whether to migrate *weakly* (only transfer the object-state) or *strongly* (transfer the agent's object-state along with it's call stack and it's instruction pointer) and whether to push all or some of it's classes to the next destination or to pull them on demand. This decision is called a migration strategy [Bra03] and has a high impact on the efficiency of the migration process. For example, in a mobile ad-hoc network it might be better to push an agent, it's whole state and all of it's code, because it cannot be assured that the unreliable connection is still functional later on to reload some classes. On the other hand, with an expensive UMTS connection, one might want to push only the agent's object state and pull single classes once they are needed.

Answers to the questions raised above can often only be determined at runtime. And since an agent should be allowed to decide and act autonomously, user intervention should be avoided for convenience (surely not in all cases). Using the environment model introduced in the previous section an agent is able to perceive changes in the environment and may therefore decide at runtime when, where and how to migrate. But the process of migration is error-prone and hence reasonable precautions have to be taken, especially in case the user did not initiate the migration but the agent. As a consequence, the agent has to act

self-responsible, which means that the agent has to make sure, that the user's access to the agent is always granted. In unreliable networks (e.g. MANETs) the agent may for example create a backup of itself on the user's device, demand a transactional but less efficient migration and choose appropriate migration strategies in order not to get lost. In addition, sophisticated exception-handling for every single stage of the migration process needs to be applied in order to react to failures and to restart an agent once the migration failed. In order to meet these requirements, the proposed mobility model extends the *Kalong* model [Bra03], which focuses on the efficiency, in a way to also include safety aspects in a migration process. This way agents may autonomously find a trade-off between efficiency and safety of a migration process, depending on their current context.

4 Conclusion

This paper proposes a generic and flexible environment model as well as an adaptive mobility model. Although each of these models is useful in itself, the combination of these enables a new kind of adaptive mobile application. On the one hand, such an application may autonomously delegate subtasks to be executed on several nodes in a distributed system, thereby considering efficiency and safety aspects so that complete execution can be guaranteed. And on the other hand, this allows for an application to follow its user (or other mobile entities) in order to offer data and services anywhere and anytime [1].

References

[Bad07] Dirk Bade. Kontextabhängige und eigenverantwortliche Migration von Software-Agenten in heterogenen Umgebungen. Master's thesis, Uni Hamburg - Dept. Informatik - AB VSIS, May 2007.

[BPL05] Lars Braubach, Alexander Pokahr, and Winfried Lamersdorf. *Jadex: A BDI-Agent System Combining Middleware and Reasoning*, pages 143–167. Whitestein Series in Software Agent Technologies, Birkhäuser Verlag, 2005.

[Bra03] Peter Braun. *The Migration Process of Mobile Agents-Implementation, Classification, and Optimization*. PhD thesis, Uni Jena, CS Dept., May 2003.

[FG96] S. Franklin and A. Graesser. Is it an Agent, or just a Program? In *Intelligent Agents III. ATAL'96*, volume 1193. Springer-Verlag, 1996.

[LMSW05] M. Luck, P. McBurney, O. Shehory, and S. Willmott. *Agent Technology: Computing as Interaction (A Roadmap for Agent Based Computing)*. AgentLink, 2005.

[McG00] R.E. McGrath. Discovery and Its Discontents: Discovery Protocols for Ubiquitous Computing. Technical report, Uni. of Illinois, CS Dept., 2000.

[RN03] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition, 2003.

[WJ95] Michael Wooldridge and Nicholas R. Jennings. Intelligent Agents: Theory and Practice. *Knowledge Engineering Review*, 10(2):115–152, 1995.

[1] Further information can be found in [Bad07]