Scalable Migration for Mobile Agents

Martin Geier, Franz J. Hauck

März 1999 TR-I4-99-02

Technical Report

Computer
Science Department

Operating Systems — IMMD IV

Friedrich-Alexander-University
Erlangen-Nürnberg, Germany

This paper was accepted and presented at the
5th Mobile Object Systems Workshop at ECOOP‘99 in Lisbon, Portugal.

Scalable Migration for Mobile Agents

14. July 1999

Martin Geier, Franz J. Hauck

{geier,hauck}@informatik.uni-erlangen.de
IMMD IV, University of Erlangen-Nürnberg
Martensstr. 1, D-91058 Erlangen, Germany

1. Introduction

One of the most promising approaches to the design of distributed software is the technique of mobile agents [White96]. A mobile agent is an entity, which is capable to migrate around in a distributed system to fulfill its task instructed by the user. The typical migration entity of a mobile agent is an object of the object-based programming paradigm [Booch94] or a whole group of objects migrating together. The usage of mobile agents has not only the advantage of work delegation but has technical reasons as well: one advantage is *lower usage of bandwith* by moving the computation with the mobile agent to the data rather than the data to the computation. Especially if the agent has to deal with vast volumes of data, as it is typical in the area of information-retrieval, the paradigm of mobile objects seems to be feasible. But the use of mobile agents is still an area of academic research, i.e. typical implementations of mobile agents are small. With implementing “real world” applications like “Electronic Commerce” or “Personalized Server Behaviour” [HCK95] agents will become bigger. Then the question arises whether mobile agents implemented as a single migration entity are still the adequate programming paradigm. The consequence would be that mobility has to be confined or the praised advantage of bandwith limitation will be lost.

2. FOAM and the AspectIX architecture

We propose the programming model called **F**ragmented **O**bject **A**gent **M**odel (FOAM), which is based on the object model of the AspectIX architecture [HBG+98]. This object model adopts the *fragmented objects* paradigm [MGN+94], which evolves the monolithic object model introduced by Booch [Booch94] to the needs of distributed systems. The basic idea is to split a single object into smaller parts—the *fragments*—which can be distributed over different address spaces. All fragments implement the same interface and represent the so called *distributed object* in the accommodating address space. The fragments of a distributed object can communicate with each other using specialized mechanisms offered by the AspectIX architecture, e.g., multicasts and optimized stream sockets. AspectIX uses a CORBA-compliant communication layer so that distributed objects can be used by all applications which fulfill the OMG standard [OMG98], and vice versa, the fragments can use any CORBA-compliant objects as well.

How migration is modelled with distributed objects of the AspectIX architecture is presented in [GSH+98]: the distributed object is extended by creating a new fragment in the destination address space. Then the distributed object is shrinked away from the source address space by removing the fragment there. The old fragment can be replaced by a simple stub, which just acts as a forwarding entity. The implementation of conventional mobile agents with FOAM is straight forward and induces no additional overhead in comparison to frameworks using a monolithic object model, but FOAM is more powerful as it allows more fine grained migration entities for a mobile agent. The idea is to split off parts of the mobile agent into fragments which then define own migration entities. There are different possibilities to perform the decomposition into fragments which are currently evalutated in our research group. Our framework has the following advantages in comparison to frameworks using a monolithic object model:

Scalability. While in ordinary systems there is only a decision between no migration, i.e. the usage of RPC-semantics, and full migration, i.e. the usage of mobile agents, in FOAM there are more possibilities of migration in between by migrating the fragments of the object. What we get is the possiblitity of *partial migration* of the mobile object: not the whole object has to migrate but just the relevant parts of it. The two usual systems RPC and mobile agent are just special cases in FOAM: if no fragments are mobile the system is equivalent to the RPC-semantic; if all fragments of the agent are migrated together we get the conventional mobile agent model.

Dynamic adaption**.** The implementor of the object offers the finest granularity of migration with the size of the provided fragments. At runtime, the object or the runtime environment can decide, how many fragments are migrated and therefore which granularity the migration has. This decision on the distribution can be changed dynamically at runtime.

Omnipresence. In specific application domains for mobile agents, multiple feedback of different hosts is needed to be able to fulfill a task. Typical examples are applications with auction semantics as they are used in bargains. In FOAM, instead of pure migration we can use replication of the relevant fragments. Technically this is easy: after a fragment is copied to the new destination, the fragment on the source side is just not deleted [GSH+98]. The different fragments therefore spread over the application domain instead of migrating from host to host. An agent can be omnipresent and i.e. haggle with different shops at the same time.

3. Conclusion

Mobile agents implemented with a monolithic object model can get too big and therefore too sluggish for migration. FOAM offers scalability in migration by using more fine-grained migration entities than whole mobile agents. Further advantages of this approach are the possiblities of dynamic adaption and omnipresence of an agent. Our current research focus the evaluation of different decomposition techniques which are relevant for the feasiblity of this approach.

4. References

Booch94	G. Booch: “Object-oriented Analysis and Design with Applications”. Second Edition, Rational, Santa Clara, California, 1994.
GSH+98	M. Geier, M. Steckermeier, F.J. Hauck et. al.: “Support for mobility and replication in the AspectIX architecture”. In: Object-Oriented Technology, ECOOP’98 Workshop Reader, LNCS 1543, Springer, 1998; pp. 325-326.
HBG+98	F.J. Hauck, U. Becker, M. Geier, E. Meier, U. Rastofer, M. Steckermeier: “AspectIX: An aspect-oriented and CORBA-compliant ORB architecture”. Tech. Report TR-I4-98-08, IMMD IV, Univ. Erlangen-Nürnberg, Sep. 1998.
HCK95	C.G.Harrison, D.M.Chess, A.Kershenbaum: “Mobile Agents: Are they a good idea?”. IBM Research Divsion, Watson Research Center, 1995.
MGN+94	M. Makpangou, Y. Gourhant, J.-P. Le Narzul, M. Shapiro: “Fragmented Objects for distributed abstractions”. In: T. L. Casavant and M. Singhal (eds.), Readings in Distributed Computing Systems, IEEE Computer Society Press, 1994, pp. 170–186.
OMG98	Object Management Group: “The Common Object Request Broker Architecture”. Version 2.2. February, 1998.
White96	J. White: “Mobile Agents Whitepaper”. MIT Press, Menlo Park 1996, URL: http://www.genmagic.com/agents/Whitepaper/whitepaper.html.