

Java-based Intelligent Mobile Agents for Open System Management

Stefan Covaci, Tianning Zhang, Ingo Busse

GMD FOKUS

Kaiserin-Augusta-Allee 31, 10589 Berlin, Germany

Abstract

Traditional approaches for network and service management are not suitable for heterogeneous, rapidly changing and extending telecommunications environments. This paper discusses the use of Java-based intelligent mobile agents in the field of open system management. It first presents a Java-based mobile agent platform supporting the mobile management agents. Based on the mobility framework, the paper then describes an intelligence platform which supports the specification and processing of management solution knowledge implemented as intelligent mobile agents. The management solutions are viewed as java-based rules, which are organized into a hierarchy by the specialization/generalization relationships among the network situations which the management solutions are developed for. The proposed intelligence paradigm and the associated knowledge processing mechanism offer the advantages of Object-Oriented development in a widely distributed telecommunications environment. The paper introduces an appealing alternative or extension to the current network management solutions.

Key-words: Network management, service management, intelligent agents, mobile agents, Java

1 Introduction

In the traditional approaches of network and service management, i.e. in the field of SNMP[13] and TMN[15]-based management, an agent has a standardized or proprietary interface, providing a fixed functionality, and is running in a specific place. It provides a limited view of the local resources at a remotely accessible management interface. This is used by management applications to monitor the state of the resources or to perform configuration operations via the SNMP or the CMIP protocol.

For several reasons such approaches seem to be insufficient in a dynamically changing, heterogeneous environment which we are facing in the network and service management area today. Here we have to deal with a multi-technology, multi-vendor equipment based environment that is rapidly developing due to technology improvements. This is quite obvious when considering the growing extensions

that each vendor defines and offers in addition to the existing management interface standards.

Furthermore, network and service management do not only include the technical supervision of the hardware but it should cover also service and business layer aspects, especially in a multi-domain networked environment, where management systems of different authorities should interact within the open service market. In such a competitive environment a provider must be able to adapt/introduce its services quickly to the demand of the market or of even individual customers.

The current management standards are not flexible enough to build dynamically extensible management solutions.

Another key issue that is not sufficiently addressed in the traditional management frameworks is the support for the acquisition and processing of management expertise. The complexity of modern telecommunications environment means that network and service management have to deal with a large amount of complicated and highly inter-related information, and have to derive management decisions based on very sophisticated expertise. Traditional management approaches focus on the syntactical aspects of the network management information, and rely on the human managers to make management decisions. Lack of enough qualified experts is now becoming a major obstacle in realizing efficient and effective management of global networks and services.

A software design paradigm with support the acquisition and deployment of management knowledge will play a key in the future of telecommunications network management.

Moreover, the management expertise and the related management solutions for specific problems can be acquired or developed anywhere within the global telecommunication market. The same network problems, however, can occur

in any remote sites lacking the necessary expertise for maintaining the solutions, and which are also far away from the source of the solutions. There must be therefore a solution for moving the management expertise over the network from its provider to the consumer sites.

To reduce requirement on human expertise and network resources in installing, maintaining and deploying the management expertise, management solutions in the form of autonomous, intelligent mobile entities will play an important role.

The promising approach in this context is the usage of intelligent mobile agents (called IAs hereafter).

An IA is defined as an *autonomous* software entity which, with certain degree of *intelligence*, can *migrate* to and operate in different network nodes to achieve its objectives.

A solution based on IAs has several advantages compared to the traditional management approaches:

- reducing the number of remote interactions and the associated traffic load for realizing the management functionality, via autonomous IA operations,
- reducing the requirement on human intelligence or personnel qualifications during the installation and operation of the management solution, by utilizing the IA intelligence,
- supporting dynamic and flexible management functionality by delegating management solutions *on demand*,
- increasing the fault tolerance and robustness of the management solutions by reducing the dependency of the management operations on the remote network connections,
- enabling, via IA intelligence, mobility and autonomy, a global open service market for network and service management solutions and expertise, where management solutions and expertise can be produced, traded, transported and consumed.

Generally, the computing and communication overhead for management activities can be reduced by putting the agent in place when a specific management action has to be per-

formed, and destroying it when there is no longer a need for it.

Certain management tasks, like gathering usage data and computing/transmission charges, can be located close to the real resources. Then we have only the reduced accounting data that need to be transported. This would reduce the communication overhead and thus the network load.

Such aspects are of major importance in a multi-domain environment where a management solution provider operates the service from a remote site over a wide-area network, which is usually of low bandwidth and has long round-trip time.

Each piece of management software solutions in the managed environment can frequently be outdated, e.g., because a bug was discovered and fixed, or because a new environment requires a different or extended functionality. Using IAs the functionality within an agent can be easily replaced by a new version of the software from a remote site.

The ability to move functionality to several places for execution allows for load balancing and results in a fault tolerant system. IAs can be directed to places with free computing power. In case an agent fails, the functionality can be loaded in a different place on another host. Each host running the agent execution environment can act as a back-up system.

A provider can react flexibly on the changing environment in a customer domain within a Customer Premises Network (CPN) and especially in an intranet management outsourcing scenario [5]. A roaming IA can collect configuration data about the customer domain and return to the provider. Further IAs can be configured and sent to the customer domain according to the equipment installed and with management functionality required, without a manual interaction between the customer and the provider. In case of special fault conditions additional agents for fault analysis/localisation can be sent to the customer domain performing analysis tasks.

2 The Agent Infrastructure

The application of the agent technology has already a history and spreads over many areas [1], including user interface/personal assistance, mobile computing, information retrieval & filtering, data mining, smart messaging, the electronic marketplace, telecommunication services and service / network management. The last area represents a still new application field for agent technology, now rapidly gaining importance in the emerging global Information Infrastructure [11].

The developments of agent technology are currently divided into two separated worlds: static agents and mobile agents.

In the context of static agents, people are focusing on the intelligence aspects of non-mobile agent technology, developing knowledge-based mechanisms for representing and processing application specific expertise and solutions, and for agent co-operations ([6] [8] [23] etc.). The typically used knowledge-based language paradigms have advantages in terms of flexibility, extensibility, reusability, user friendliness and enhanced service support over conventional languages which are typically used in the mobile agent context.

Research and developments in the area of mobile agents focus on the platform for supporting agent migration and management. Agents in this context are usually any program codes in some interpreted imperative languages with strong Internet association. Most important languages in this context are Java, Tcl/Tk or Telescript. The Internet flavour of such languages and the demand for mobile solutions in a distributed global telecommunications environment enable such developments to gain quickly academical and commercial importance. Examples of important platforms developed in this context include the IBM's Aglet [12], Crystaliz's MuBot [3], General Magic's Telescript [10] and Odyssey[9], the Open Group's MOA [21], and GMD FOKUS's JMAF [14].

The separation of static and mobile agents is in fact very *artificial* and was the result of the different backgrounds in

which the two technologies emerged. With the distribution and complexity of the telecommunications environment, both mobility and the intelligence based on knowledge processing will be needed for IA-based network and service management solutions.

Basically, the infrastructure supporting the operation of the IAs can be divided into two sub-platforms:

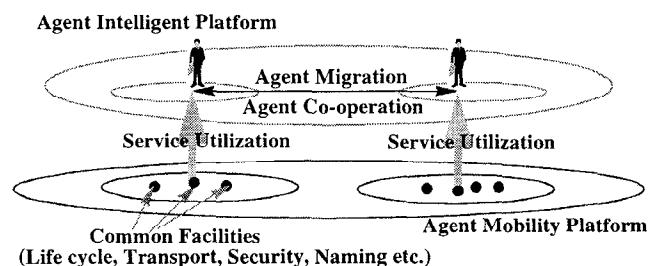
- The *agent intelligence platform* supports the representation and processing of the application specific mobile knowledge. Such mobile knowledge includes the representation of the IAs and the co-operation messages among the IAs.
- The *agent mobility platform* provides the common facilities which will be needed for the management, migration of IAs and the transportation of IA co-operation messages. Such common facilities can include:

- IA life cycle management (creation, deletion, suspension and activation),
- IA naming and location services,
- IA transportation and agent message transportation,
- security

The agent intelligent platform is in fact the focus of current developments in the area of static agents, while the agent mobility platform falls into the domain of mobile agents. Current standardization in OMG [4], e.g. only addresses the agent mobility platform functionalities.

The agent intelligence platform can be regarded as a client layer of the agent mobility platform, which utilizes the facilities offered by the underlying mobility platform components (figure 1).

Figure 1: The IA Infrastructure



3.3 Name Server

The *name server* provides a directory or naming service. It implements a mapping of URLs to remote object references. The *factory* of an agency is registered at a *name server*. This allows us to use a URL to identify the destination in a *move* operation on a mobile agent.

4 The Intelligence Platform for Agent-based Network and Service Management

To deal with the complexity of managing the emerging telecommunications environment [11], just like in any other areas which provide real challenges to human intelligence, we have to introduce some machinery with needed machine intelligence. Such intelligence can assist or partially replace the human experts in fulfilling their tasks.

It is in this context that the *intelligence* based on management expertise, and its combination with the mobility of agent technology become promising factors for solving network and service management problems.

As discussed above, the intelligence platform is regarded as a client of the mobile agent platform, which makes use of the facilities provided. Within the agent intelligence platform an IA is a subclass of the generic agent class supported by the mobility platform.

Each IA hosts a specific *run()* method which realizes the inference procedure on the management knowledge of the IA. This inference procedure finds out and initiates the appropriate management actions.

4.1 The IA Knowledge Representation

In an open environment, numerous players can contribute to the management solutions with their own experiences and views. A suitable framework supporting such a global and co-operative acquisition of knowledge will therefore determine the success or failure of the IA-based approach. The following characteristics of such a knowledge acquisition framework will play a key role:

- Reusability and Interoperability

The global telecommunications environment encompasses a tremendous number of heterogeneous sub-environments. The success of any solutions in this world depends on their capability of integrating themselves into the operation environments, i.e. the ability to be applied in a wide range of contexts.

Being important in all the scientific areas, *reusability* and *interoperability* provide the dominating requirements in the case of distributed resource management.

Reusability and interoperability in the distributed environment are usually achieved through standardization at different syntactical or semantic levels, and through Object-Oriented constructions.

- Accumulative Construction

To cope with the complexity of the management scenarios, an enormous amount of very complicated knowledge has to be used. Such knowledge can only be derived from profound experiences which have to be collected gradually during the evolution of the resources.

Management however has to be carried out during the whole life cycle of the resources. At the beginning it has to be based on some very primitive management knowledge and operations. As time passes by, new knowledge concerning the deeper inter-relationships between the components of the environment and the detailed behaviour of the components will be obtained, and new management operations can be added due to new requirements and new facilities. All such new knowledge and functionalities have to be gradually integrated into the management stations.

In this sense, accumulative construction of the intelligent management systems is also an important factor in realizing effective management of future networks. The most important technology in this context is the Object-Oriented style of development, which is also the major philosophy of the our IA-based approach for network and service management.

The knowledge for the management IA is the set of *management solutions* that address the problems that can occur

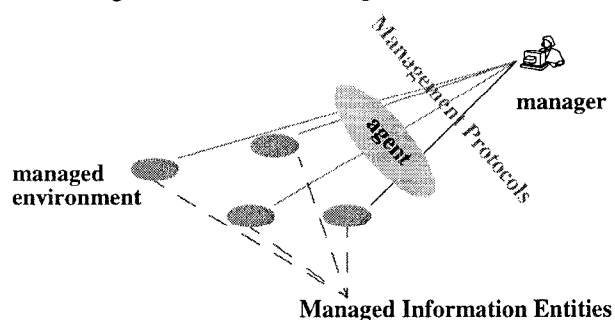
in the network. Each management solution can be constructed from two key components:

- the *pre-condition* for the solution, in terms of the network *situations* that require or support the application/invoke of the solution, and
- the *management plan*, defined as a pre-defined sequence of management actions, including the generation of new Event Reports.

4.2 The Situation Theory for IA-based Network and System Management

Most of the currently dominating network management frameworks, including the TMN[15] or SNMP[13] based ones, view the managed network environment with its set of resources as a set of managed information entities, as depicted in figure 3.

Figure 3: Network Management View



In this context, the managed resources like switches, hosts and processes are controlled by some management agents. Such management agents implement the abstract view of the underlying resources, defined as certain *managed information entities*. Such information entities can be managed or processed by the managing stations or managing processes via standardized management protocols. The protocol (SNMP, or CMIP in case of TMN) and the specification of the managed information (based on either the SMI/SNMP or the TMN/GDMO [17] specification methodology) determine together the execution environments for any network managing entities including the IAs.

In this paper, we discuss only the TMN/CMIP-based management environment. The framework, however, can be easily adapted to a SNMP-based environment.

In a TMN/CMIP-based environment, the managed information entities are represented as Managed Object (MO) Instances. Each of such MOs can possess the following information capabilities:

- the *attributes*, which contain the status and characteristics information concerning the managed resources,
- management *actions*, which support the management operations upon the underlying resources, and
- the *notifications* with their parameters, which the managed resources can issue to the managers for signalling special events in the network. In the following we will call notifications received by the managers *Event Reports (ERs)* to maintain a simple terminology.

A manager in this context can make the following basic operations on the MOs:

- *get* for obtaining the value of an attribute by providing the Identifier of the attribute
- *set* for setting the value of a MO attribute
- call a management *action* which is supported by the MO
- receive an *ER* from the agent

In such an environment, the *current situation* in the network is characterized by the set of MO instances (with their attribute values) and the generated ERs (with their parameters).

Based on this consideration, we can develop a situation theory for modelling the situations that can occur in the network environment ([22], [23]).

Within this situation theory, a *situation concept* characterizes some possible situations that belong to a specific cate-

gory and can be used to specify the pre-condition for management solutions.

To test and determine the current situation in the network, we can apply a java method *situation test*, which checks whether the current *situation* in the next work corresponds to a specific *situation concept*.

Some *situation concepts/situation tests* put more restrictions on the possible situations they characterise, and therefore characterise a smaller set of possible real situations. If a situation concept or situation test characterises a subset of situations characterised by a second situation concept/situation test, we say that the first situation concept/situation test is a *specialization* of the second one.

With the specialization relationship, we can derive the hierarchical structuring of the *situation tests* in an Object-Oriented style. Based on the logical operations of the set theory, we can also derive logic operations on situation tests to derive specialized situation tests from more general ones. E.g. the *conjunction (composition)* of a situation test τ and any other situation tests is a specialization of τ .

Intuitively, a specialized *situation concept* contains fewer real situations and can make a closer approximation of the situation in the network. Correspondingly, a specialized *situation test* contains more restrictions on its semantics and can therefore more closely characterize the network situations.

Management plans based on specialized *situation tests* can therefore support safer, and more effective solutions of the current problems in the network. Management plans based on more general situation tests however can be applied in a wider context.

In our Java-based management environment, the pre-condition for a management solutions is specified by the *situation tests* that characterize the network situations for the management actions (management plans).

To get the side-effect free testing of the managed network environment, we restrict the management operations within a *situation test* to the CMIP *get*. With some *approximation*,

we can consider the tests as side-effect free. The result of the test composition can be considered as independent of the order in which individual tests are carried out, and is always the specialization of a single test.

A *situation test* in our IA framework is just a boolean Java method which uses only the *get* methods from the CMIP API and any other Java basic methods. This property will be guaranteed by the editing environment for defining IA knowledge.

4.3 The IA Management Solution

As discussed above, a *management solution* consists of a *situation test*, which checks the situation in the environment, and a *management plan* for addressing the corresponding problems in the network by issuing management operations.

Based on the hierarchical structuring of the *situation tests*, we can have also the hierarchical relationship among the management solutions. For building up such a hierarchy of management solutions, we impose the following conventions on the development of management solutions:

- Each management solution will be contained in a separate Java package. Each management solution package for a more specific problem is positioned under the package for a more generic solution.
- The *situation test* within each management solution implies automatically the *situation tests* in all the ancestor (more generic) management solutions.

In this way we build up a hierarchy of management solutions which starts with generic management solutions at the upper part, and grows with specific management solutions positioned under more generic solutions.

The management solutions will be pre-compiled to the Java *class* files. At the time of IA creation, these *class* files will be loaded to build a corresponding hierarchical Java data structure, called the *managementHierarchy*, which contains the management solution classes.

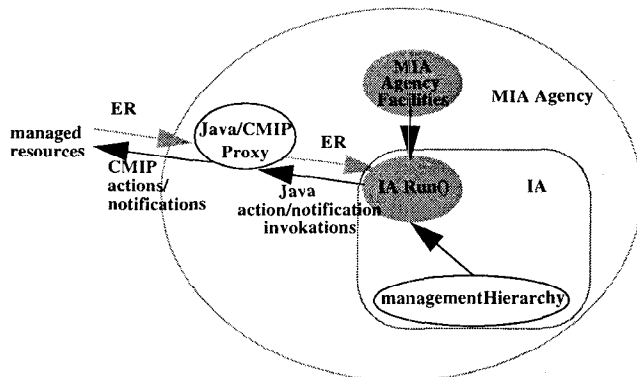
An IA consists of the following key components in addition to the components of the mobile agents:

- the *eventMemory* for holding the set of events collected by the IA during its life time, and which are still relevant for the IA processing,
- the *currentEvent* which is driving the inferencing of the current IA,
- the *managementHierarchy* of management solutions,
- a specific *run()* method for the execution of the IA functionality.

4.4 IA Operation

The IA operation will be realized by the *run()* method which is the same for all the IAs in this context.

Figure 4: IA Operation Environment



IAs are generated by some management stations upon ERs from the resources. The initial IA will be equipped with an initial *managementHierarchy*.

Each time when an IA enters an *agency* and is started by the *agency*, the IA first integrates the local management solutions hierarchy into the IA, and then starts inferencing on the knowledge hierarchy to find the suitable management operations to be issued to the managed resources via the Proxy.

To update the management solution hierarchy, the IA traverses through the local hierarchy of the Java management solution package file directories, creates an record of

each management solution class, and builds up a hierarchy of management solutions (implemented in Java as vector of vectors) to be assigned to the corresponding IA variable.

After the creation of the *managementHierarchy* and the initialization of other variables, the IA starts the traversing of the hierarchy to find a most *specific (specialized)* management solution for the current situation in the network. The traversing will be done in pre-order and started at the root of the hierarchy. At each management solution node the IA will do the following:

1. create and initialize the management solution instance
2. call the *situation test* method
3. if the result of *situation test* is true, and all the children *situation tests* returned false, then call the *management plan*, otherwise continue the traversing of the hierarchy on the next management solution node.

The result of the *managementPlan* method has three possibility:

1. the method returns *null*, and sets the *currentEvent* variable to *null*
2. the method returns *null*, but sets the *currentEvent* variable to a new ER (one from the *eventMemory*, or a newly generated one)

In this case the IA will set the hierarchy to the subhierarchy under the current node and restart the process of traversing, based on the intermediate result coded in the new ER.

3. the method returns a new *Agency* address different from the current host

In this case, the IA host sets the *managementHierarchy* to the subhierarchy under the current node and moves the whole IA instance to the new IA Host (figure 5).

In the new network node, the IA first updates the *eventMemory* by adding to it the local events, and then tries to update the hierarchy of management solutions with the local hierarchy of solutions.

To update the management hierarchy, the new IA can also replace those nodes in the IA which are also defined in the

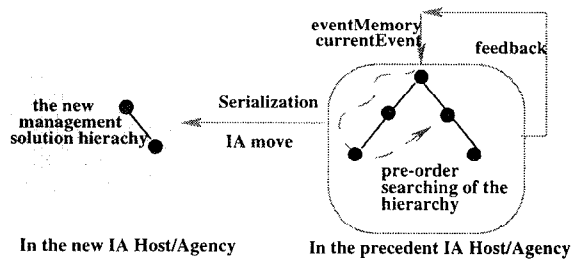
local hierarchy (with the same package names) and the descendant nodes of such shared nodes.

After updating the IA knowledge, the IA starts execution just like in the procedure described above.

The advantage with this whole scheme is that each local network node can maintain its proprietary solutions for its local resources. The travelling IA will consider these local solutions even if these are not available in the origin node of the IA.

The pre-order traversing of the management knowledge hierarchy imposes the implicit priorities on the management solutions. This proves to be very useful in the real applications.

Figure 5: IA Execution and Mobility



5 Summary

In the above we presented a Java-based mobile agent platform and an intelligent platform for deploying Java-based intelligent and mobile agents in open system management. The combination of mobility and the intelligence based on knowledge processing offers many new possibilities in this context.

The intelligence framework is based on:

- the collection of management solutions in an open global network environment,
- the organization of such management solutions into some global knowledge hierarchies following the application context situations of such solutions,
- using such hierarchies to support the mobile IA in finding out an optimal solution for the problems existing in the networks.

The whole approach has, among others, the following major advantages:

- Automation and delegation of management solutions via mobile, knowledge-based Agents

The management framework enables the delegation of autonomous management solutions. In this way, an enterprise can outsource ([2], [5]) part of the management responsibility and management problems to some specialized management solution providers.

Such outsourcing and the automation of management processes can greatly reduce the complexity and costs within the enterprise for managing the enterprise network resources. With the increasing complexity of broadband telecommunications environments, such reduction of costs will be a major issue in applying any management frameworks.

- Compatibility with existing environments

The approach is designed to be based on the TMN management framework and can easily be adapted to a SNMP environment. Each IA communicates with the managed resources via standardized management protocols and operates on the management view offered by the Management Information Base (MIB). The co-operation among the IAs is realized via TMN Event Reports and Event Forwarding [16] service, instead of novel co-operation languages like KQLM [6]. This approach can therefore be easily embedded in a standard conform network management environment.

- Reusability and Interoperability

A management solution hierarchy starts with some generic knowledge/solutions which are applicable in different environments, and it derives specific knowledge by adding specific information about the individual environment classes. If applied in a different environment, the generic management knowledge in the upper part of the hierarchy will continue to be valid, while the lower part situation knowledge will not be satisfied and will be simply ignored. The result

is that the IA with this hierarchy operates also in a new environment, although with some loss of effectiveness.

For the same reason, one management solution hierarchy can be reused in different environments to build there the management systems.

- Accumulative and distributed construction of the global knowledge

When starting to manage an environment with an IA-based framework, we usually use a partial management solution hierarchy to initialize the management functionality. Later, with accumulating experiences and expertise, we can add more specific management solutions with new situation tests and their associated management plans, to improve the performance of the IA system.

Moreover, the management solutions can be developed/advertised by any players anywhere in the open world. The players can also decide to maintain such solutions only in the proprietary environments, and to ask IAs who visit their sites to use such proprietary solutions instead of the generic ones.

- Standardization of global available management solutions

The hierarchical structuring of the management solutions in the Object-Oriented style enables the standardization of the relatively generic management solutions in various contexts and domains within the upper part of the hierarchy under the root. Such management solutions can be further made available to all users via hierarchical information services like the ITU-T X.500 Directory Services[18].

With these characteristics and possibilities, we expect the approach to provide a realistic management framework for the current and emerging telecommunication networks and services.

6 References

- [1] Communication of the ACM Journal, *Intelligent Agent*, Vol.37, No. 7, July 1994.
- [2] F. Dotti, S. Covaci, *ODP Viewpoints of Management Outsourcing*, Proceedings of the IEEE /IFIP Network and Management Symposium (NOMS'96), Kyoto, Japan, April 1996.
- [3] Crystaliz, *Mobile Unstructured Business Object (MuBot)[TM] Technology - A Fundamental Revolution*, <http://www.crystaliz.com/MobileAgents/marketing.html>.
- [4] Crystaliz, General Magic, GMD FOKUS IBM and The Open Group. *Mobile Agent Facility Specification* (Joint Submission), Draft 5, April 21, 1997.
- [5] F. Dotti, *Management Outsourcing in Open Distributed Environments*, Proceedings of the Enterprise Networking Workshop (ENW-96), Dallas, June 1996.
- [6] T. Finin, *KQLM as a Agent Communication Language*, Proceedings of the 3rd International Conference on Information and Knowledge Management (CIKM'94), Amsterdam, 1994.
- [7] FIPA, *Agent Management*, FIPA'97 Specification Part 1, April 1997.
- [8] FIPA, *Agent Communication Language*, FIPA'97 Specification Part 2, April 1997.
- [9] General Magic, *Odyssey*, <http://www.genmagic.com/agents/odyssey.html>.
- [10] General Magic, *Telescript Technology: Mobile Agents*, <http://www.genmagic.com/Telescript/Whitepapersh/wp4/whitepaper-4.html>, 1996.
- [11] *The Global Information Infrastructure: Agenda for Cooperation*, <http://www.iitf.nist.gov/documents/docs/gii/giia-gend.html>.
- [12] IBM, *IBM Aglets Workbench White Paper*, <http://www.trl.ibm.co.jp/aglets/whitepaper.htm>.
- [13] IETF, *Structure and Identification of Management Information for TCP/IP-based Internets*, RFC 1155, May 1990.
- [14] IKV, *JMAF - A Development Kit for Intelligent Mobile Agents*, <http://ikv.de/products/fjmaf.html>.
- [15] ITU-T, Recommendation M.3010, *Principles for a Telecommunications Management Network*, March 1995.
- [16] ITU-T, Recommendation X.724, *Information Technology - Open Systems Interconnection - System Management: Event Report Management Function*, 1993.
- [17] ITU-T, Recommendation X.722, *Information technology - Open System Interconnection - Structure of Management Information: Guidelines for the Definition of Managed Objects*, January 1992.
- [18] ITU-T, Recommendation X.500, *Information technology - Open System Interconnection - The directory: Overview of concepts, models, and services*, November 1995.
- [19] Sun Microsystems, Inc., *Java Object Serialisation Specification*, Revision 0.9, May 1996.
- [20] Sun Microsystems, Inc., *Java Remote Method Invocation Specification*, Revision 0.9, May 1996.
- [21] The Open Group, *Mobile Objects and Agents (MOA) Project*, <http://www.opengroup.org/RI/DMO/dmo.htm>.
- [22] T. Zhang, S. Covaci, *The Semantics of Network Management Information*, Proceedings of the 1996 IEEE INFOCOM Conference, San Francisco, March 1996.
- [23] T. Zhang, S. Covaci and Radu Popescu-Zeletin, *Intelligent Agents in Network Management*, Proceedings of the 1996 IEEE GLOBECOM Conference, London, 18-22 November 1996.