# MODELLING NETWORK TOPOLOGY AND MOBILE AGENT INTERACTION: AN INTEGRATED FRAMEWORK

Marco Cremonini, Andrea Omicini
LIA – DEIS – Università di Bologna
Viale Risorgimento 2 – 40126 Bologna – ITALY
*E-mail: {mcremonini, aomicini}@deis.unibo.it*

Franco Zambonelli
DSI - Università di Modena e Reggio Emilia
Via Campi 213/b – 41100 Modena – ITALY
*E-mail: franco.zambonelli@unimo.it*

**ABSTRACT**

The paper introduces the HiMAT model for mobile agent applications, which provides a unique, coherent framework for the design and development of mobile agent applications, where critical issues such as topology, authentication, authorisation and coordination can be effectively addressed in a uniform way.

## 1 INTRODUCTION

A mobile agent is an autonomous computational entity which *(i)* proactively moves through a multiplicity of different execution frameworks, and *(ii)* there interacts with local resources and other mobile agents. As a result, in order to effectively support application design, a mobile agent model should provide the abstractions and mechanisms *(i)* to deal with mobile agents roaming a space of hosting environments, and *(ii)* to manage the space of mobile agent interaction.

From the mobile agent's viewpoint, the former issue is essentially a *topology* one: how agents represent the space which they roam, whether their representation is partial or complete, and whether it is statically given or dynamically acquired. From the hosting environment's viewpoint, the same issue is related with the notion of agent *identity*, where different execution frameworks have to authenticate the same agent in motion. The latter issue concerns the *coordination* of mobile agents and local computational resources [5][7]: how their interaction can be constrained and driven so as to result in a system behaviour accomplishing the global system's requirements. The above two issues are then strictly related by the problem of *authorisation*: mobile agent's access to local resources has to be ruled according to both global policies, shared by a collection of hosting environments and locally specialised policies.

Starting from the above considerations, this paper introduces the HiMAT model for the design and development of mobile agent applications, which uniformly addresses all the above issues. HiMAT derives from two models for mobile agents, MA [3] and TuCSoN [7]. HiMAT models the Internet as a collection of hierarchical locality domains (derived from MA), where agents can exploit programmable tuple spaces (derived from TuCSoN) both to coordinate their interactions and to dynamically acquire knowledge. On the one hand, locality domains reflect the hierarchical topology of Internet administrative domains and makes agent motion and authentication easier to be managed. On the other hand, TuCSoN programmable tuple spaces (called *tuple centres*) uniformly act as both the authorisation engines providing agents with the knowledge about the domain resources they can access, and the media for actually interacting with those resources. By extending and integrating the two models, HiMAT provides a unique, coherent framework enabling mobile agent system designers to effectively deal with topology, authentication, authorisation and coordination.

## 2 THE HIMA EXTENSION

Internet-based applications typically deal with intrinsically structured domains: *(i)* Internet nodes are often grouped in clusters, subject to highly coordinated management policies; *(ii)* large clusters can be further characterised by the presence of enclosed sub-clusters, in a hierarchical structure of protected administrative domains. The first observation *(i)* led our research group to the design and the implementation of a mobile agents system, called MA and firstly presented in [3], which explicitly models Internet as a collection of *places*, *domains*, and *gateways.* The *place* provides the abstraction for the mobile agent's execution environment, where agents executes their code and interact with local resources. At a higher level of abstraction, the *domain* concept is used to model subnetworks by grouping a set of places sharing common policies and privileges. Finally, mobile agents moving from one domain to another one rely on a specific abstraction, the *gateway*, in charge of inter-domain routing for both incoming and outgoing agents. The second consideration *(ii)* led us to present the HiMA extension, where the MA notion of domain becomes *hierarchical* and may contain both places and other (sub)domains. This provides HiMA with high flexibility for modelling complex network structures, which are often intrinsically hierarchical, and with a better support for decentralised management and for the definition of global policies at different organisation levels, given the clear separation of all the administrative domains of a network. Consequently, the distribution of HiMA domains on the network can be represented with a tree structure where each node is a domain (containing places) and each arc is an interconnection gateway. Each hierarchical HiMA domain has a most external gateway bridging HiMA with Internet and enabling communication between different domains. Note that also the internal structure of a HiMA domain is Internet-based: in this context, the distinction is between the portion of the net structured according to the HiMA framework and the rest of the Internet.

## 3 THE TUCSON MODEL

TuCSoN defines an interaction space spread over a collection of Internet nodes and built upon a multiplicity of independent tuple-based communication abstractions called *tuple centres*, associated to each Internet node, and used by agents for their interactions [7]. Tuple centres [4] are (enhanced) tuple spaces *à la* Linda [2] and, then, enable temporally and spatially uncoupled interactions. Each tuple centre is associated to a node and is denoted by a locally unique identifier. In its turn, each node provides its own version of the TuCSoN name space (the set of the tuple centre identifiers) by virtually implementing each tuple centre as an Internet service. As a result, any tuple centre can be identified either via its full Internet (absolute) name or via its local (relative) name, thus supporting the double role of mobile agents, as network-aware entities explicitly accessing to a remote tuple space, and as local computational entities implicitly interacting through tuple centres with both other agents and local resources of their current execution node.

TuCSoN tuple centres enhance tuple spaces with the notion of *behaviour specification*: each tuple centre can be programmed so as to implement its own observable behaviour in response to communication events [1]. Instead of simply triggering the basic pattern–matching mechanism of the Linda model, the invocation of any of the TuCSoN basic communication primitives can be associated to specific computational activities, called *reactions*. The result of the invocation of a communication primitive is perceived by agents as a single-step state transition of the tuple centre state combining altogether the effects of the primitive itself and of all the reactions it triggered.

As shown in [7], coordinating mobile agent interaction with TuCSoN leads to several interesting features: *(i)* interactions related to different application contexts can exploit the multiplicity of the tuple centres, each one separately and independently programmable; *(ii)* agent's interaction protocols can be made independent of the particular execution framework architecture, thus dealing with typical heterogeneity of an Internet application; *(iii)* coordination policies can be charged upon programmable coordination media, thus freeing agents from the burden of coordination–awareness; *(iv)* agent's accesses to resources, always mediated by the tuple centre, can be easily controlled and managed. A suitable behaviour specification, associated to a tuple centre, enables the definition of any access control policy.

## 4 HiMAT

As they address complementary issues of mobile agent computing, HiMA and TuCSoN present complementary limitations, too. While HiMA defines no specific abstractions to deal with mobile agent's space of interaction, the TuCSoN model falls short in considering and modelling complex and secure application environments.

The HiMAT model combines HiMA and TuCSoN in a uniform, coherent framework suitable to support the four fundamental phases of the life of a mobile agent: its movement over the network (*topology*), the verification of its identity (*authentication*), the permission to access resources (*authorisation*), and the management of its interactions with other agents and with local resources (*coordination*). The first (topology) and the last (coordination) are handled by exploiting the features of HiMA and TuCSoN. In addition, the seamless integration of the two models enables TuCSoN to be exploited as the authorisation engine with respect of resource access, and HiMA gateways to be used as filters controlling agent's movement and domain representation in an application-oriented environment.

More in details, a HiMAT *place* belongs to one HiMAT *domain*, and each domain groups a set of places along with other (sub-)domains, implicitly defining a tree structure (HiMAT *tree*). Each child domain is connected to its parent domain by its associated *gateway*, enabling agents to traverse the hierarchical structure of a HiMAT tree. Thus, in order to provide a uniform interaction style between agents and nodes of the hierarchy, the communication space is implemented as a TuCSoN environment in both gateways and places. In the hierarchical structure of HiMAT, gateways, besides acting as centralised points for domain authentication and access control, can be now naturally exploited to provide mobile agents with a multi–layered description of the network topology, where each gateway only describes a single level (the structure of its associated domain). This enables a better management of the system knowledge, by delegating to each domain (through its gateway) the representation and management of what is related with its inner structure only, security issues included. This simplifies the task of mobile agents when dealing with network topology, since information about domains can be acquired incrementally whenever crossing gateways. Each gateway could perform the authentication of incoming agents on behalf of its associated domain.

In addition to its authentication role, a gateway works as a knowledge repository, providing agents with information about the structure of the domain, filtered according to agent identity. By shifting the complexity of knowledge management from agents to domains and gateways, this approach avoids the agents need for *a–priori*, complete knowledge of the system topology. This makes also gateways implicitly work as the first authorisation level of the HiMAT model: a mobile agent can retrieve from a gateway the set of the accessible places and subdomains, as well as the set of the visible tuple centres locally provided by each place of the domain. The second authorisation level relies on each place which provides specific access privileges over local tuple centres to control the (mediated) interaction with local resources.

In order to better understand HiMAT, we present the scheme of a possible interaction protocol of a mobile agent along a HiMAT tree (Fig.1), where a clear distinction between the space which mobile agents roams (*network topology*), and the management of the interaction among mobile agents and resources (*local interaction*) is shown. The exploitation of the same interaction protocol, mediated by TuCSoN tuple centres, allows both authorisation and coordination policies to be dealt with in a uniform way.

## 5 EXPLOITING THE HIMAT MODEL

Let us consider an application in which mobile agents look for book references through the Internet nodes of University libraries. Suppose to model the department network with HiMAT: we define a tree with the CS (Computer Science) domain (`cs`) as the root and all the research group domains as children, such as the MAG (Mobile Agent Group) one (`mag`).

| | |
|---|---|
| *network topology* | |
| `<goto d>` | *migration to gateway* `d` |
| `<identify>` | *gateway* `d` *authenticates the agent on behalf of all the places of its domain* |
| `<get domain information>` | *information about domain structure, in terms of accessible subdomains* (`subdlist`)*, places* (`placelist`)*, and tuple centres* (`commspace`)*, filtered according to agent's identity* |
| `<for p in placelist do>` | *exploration of accessible places of the domain* |
| `<goto p>` | *migration to place* `p` |
| *local interaction* | |
| `<for n in commspace do>` | *for all the visible tuple centres of place* `p` |
| `n?op(Tuple)` | *ask tuple centre* `n` *of place* `p` *to execute* `op` *over* `Tuple`*, if authorised by* `p` |
| *network topology* | |
| `<for sd in subdlist do>` | *exploration of the accessible subdomains* |
| `<goto sd>` | *migration to gateway* `sd` |
| `<...>` | *keep on exploration and access* |

Fig. 1. A scheme of possible HiMAT interaction protocol

The MAG domain defines the `maglibrary` and `magvarious` places of the `mag` domain providing for the `books` tuple centre and representing, respectively, the books already catalogued, and the not–yet catalogued ones. The application `bookreader` is in charge of exploring the `books` tuple centres for finding book references, by exploiting two classes of agents: the first one (role `ordinary`) takes into account catalogued books only, the second one (role `advanced`) considers not–yet–catalogued books too. Once authenticated by the `cs` gateway, agents access its default tuple centre to discover what places are in the domain and which tuple centres are available. Here agents recognise the presence of an inner gateway, the `mag` one, which they may be interested to explore. As `bookreader` agents arrive to the `mag` gateway and access its default tuple centre, they can be provided with a particular view of the internal domain structure, based on their identity, as shown in the example: *(i)* a `bookreader` agent with `ordinary` role is made aware of the `maglibrary` place only and can access its `books` tuple centre only (Fig. 2a); (ii) a `bookreader` agent with `advanced` role is made aware of both the `maglibrary` and the `magvarious` places, and can access both the associated `books` tuple centres (Fig. 2b).

The two `books` tuple centres mediate agent's access to different information structures. Catalogued books are recorded in the `maglibrary` place exploiting a DBMS. The DBMS interfaces with the local tuple centre `books` either directly or through a wrapper, translating tuples into queries, and back answers into tuples. Not–yet–catalogued books, since they are typically a small number, are instead directly recorded as (logic) tuples in the `books` tuple centre provided by the `magvarious` place.

As a result, semantically homogeneous information (concerning books) is represented in an heterogeneous way by the two places. However, coherently with the fact that mobile agents dynamically acquire knowledge about the domain, they should not be forced to take into account heterogeneity of the information sources in their code. Instead, they should be allowed to adopt the same, straightforward interaction protocol, independently of the (architecture of) their current hosting environment. Let us suppose `bookreader:advanced` agents ask both tuple centres `books` for tuples `bookTitle(`*`Title`*`)`. Then, programmability of the tuple centres can be exploited to provide agents with a uniform interface composed only of tuples of the form `bookTitle(`*`Title`*`)`, translating them into DBMS queries and returning the answer tuples.

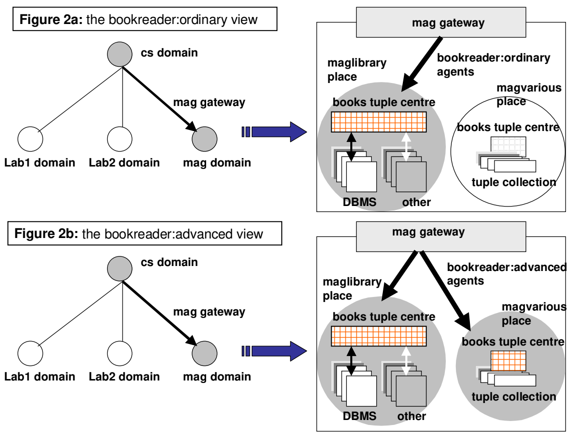


Fig. 2. An example of HiMAT interaction space

## FUTURE WORKS

The HiMAT system is being implemented by integrating the Java implementations of MA and TuCSoN. Further work will be devoted to a deeper investigation of the issues of the HiMAT security and access control models, as well as on the identification of suitable high-level design patterns for HiMAT application development.


**ACKNOLEDGEMENTS:** This work has been carried out under the financial support of the Italian MURST (Ministero dell'Università e della Ricerca Scientifica e Tecnologica) in the framework of the Project MOSAICO, "Design Methodologies and Tools of High PerformanceSystems for Distributed Applications".


## REFERENCES


[1] Cabri G., Leonardi L., Zambonelli F., 1998, Reactive Tuple Spaces for Mobile Agent Coordination, *2nd International Workshop on Mobile Agents*, Stuttgart.

[2] Carriero N., Gelernter D., 1989, Linda in context, *Comm. of the ACM,* Vol. 32, No. 4, pp. 444-458.

[3] Corradi A., Cremonini M., Stefanelli C., 1998, Locality Abstractions and Security Models in a Mobile Agent Environment, *IEEE WETICE'98*, Stanford University.

[4] Denti E., Natali A., Omicini A., 1998, On the expressive power of a language for programming coordination media, *Proc. of the 1998 ACM Symposium on Applied Computing*, Atlanta, Georgia.

[5] Gelernter D., Carriero N., 1992, Coordination languages and their significance, *Comm. of the ACM, Vol.* 35, No. 2, pp. 97-107.

[6] Karnik N. M., Tripathi A. R., 1998, Design Issues in Mobile-Agent Programming Systems, *IEEE Concurrency*, Vol. 6, No. 3, pp. 52-61.

[7] Omicini A., Zambonelli F., 1999, Tuple Centre for the Coordination of Internet Agents, *Proc. of the 1999 ACM Symposium on Applied Computing*, San Antonio, Texas.