Dynamic resource management for Mobile Services

Carmelo Ragusa, Antonio Liotta, George Pavlou

Centre for Communication Systems Research, School of Electronics and Physical Sciences
University of Surrey, Guildford, Surrey, GU1 7XH, UK
{C.Ragusa, G.Pavlou, A.Liotta}@eim.surrey.ac.uk[1]

Abstract: In this paper we propose a dynamic resource management system addressing the key requirements of mobile services – i.e. services realized as Mobile Agents (MAs). MAs are exploited here in different ways. First, to realize adaptable, configurable, context-aware services for 3G and beyond. Second, to develop a distributed monitoring system that suits the hurdles posed by service and network mobility. Finally, to construct a management system that can dynamically re-configure MA-based services for load-balancing and adaptation purposes. We present the resource management system architecture, a scheme providing run-time adaptation through agent mobility, and a prototype implementation along with some important simulation results.

1 Introduction

The increasing availability of relatively low-cost portable devices, accompanied by a rapid deployment of wireless and cellular infrastructures, has made mobile computing a reality. We are also witnessing how personal communications as well as business have become dramatically dependent upon networked computing systems relying on high-speed communication for multimedia and web-based services. The newly deployed 3rd Generation (3G) networks allows not only personal mobility and low-latency communication but also the so declaimed "always on" connections, which means access to advanced services in (almost) all situations while on the move.

In the meantime, the delay of the 3G launch has allowed sufficient time for the maturation of those sophisticated services that could not fully exist without 3G. Virtual Home Environment services, context-aware services and adaptable services are just examples of the enormous potential that can be unleashed in the 3G context.

Fundamental 3G services and applications have started to appear, but operators will only see a real return of their investments if they will be able to efficiently deploy more advanced services that exploit the full potential of 3G. Advanced services are also needed to justify further investments on communication infrastructures beyond 3G. Services and service platforms seem, therefore, the key element of the current

[1] The work reported in this paper has formed part of the WA1 area of the Core 2 Research Programme of the Virtual Centre of Excellence in Mobile & Personal Communications, Mobile VCE, www.mobilevce.com, whose funding support, including that of EPSRC, is gratefully acknowledged. More detailed technical reports on this research are available to Industrial Members of Mobile VCE.

telecommunication scenario and it is for this reason that so much attention is currently being dedicated to their effective realization.

Many problems still exist when it comes to deploying mobile services, that is services realized through software components or objects that are mobile and distributed. Mobile Agent (MA) technologies are good candidates for the development of such services. In the context of 3G, mobility assumes also a second dimension, that of node or terminal mobility. Software mobility can then be exploited in two different ways. The first one follows the conventional use of MA systems, which uses software mobility for applications such as user profiling, delegation of responsibility and so forth. A second important application consists of exploiting software mobility to overcome the issues arising from network and terminal mobility. The latter represents an attempt to deploy services in a way which guarantees an efficient use of the networked resources, despite of the dynamic changes in their status caused by node mobility.

Mobile services also have to overcome the hurdles arising from the peculiarity of mobile networks. They have to be resilient to Quality of Service (QoS) variations and temporary loss of connectivity. They also have to be designed in a way which accounts for the relatively higher cost of wireless links with respect to wired ones. Finally, they have to deal with the limited power and capability of mobile terminals. This implies that it should be possible to deploy resource-intensive services to users that are connected through thin clients.

The abovementioned requirements call for effective resource management solutions. We have been looking at the issues related to 'dynamic resource management for 3G services and beyond' within the Virtual Centre of Excellence in Mobile and Personal Communications (M-VCE) project [13]. In this paper we describe our approach and illustrate our initial findings and lessons learned. We have developed a middleware-based system, a logical service layer sitting on top of the network and providing an environment for the execution and management of MA-based mobile services.

While the M-VCE project has an ample scope ranging from low-level communication aspects to the development of 3G services, this paper is focused on those aspects related to managing MA-based services, mainly exploiting code mobility for dynamic resource balancing purposes.

In the rest of this paper we first give an overview our service execution environment, the Common Agent Location and Execution Environment (CALExE) that is further described in [3] [18]. We also outline the services and scenarios developed by the M-VCE project, as described in [14]. These are used here as reference cases for the dynamic resource management system (Section 3). Our results are presented in Section 4, followed by a survey of related work (Section 5).

2 CALExE & Mobile Services

The CALExE architecture and system ([3] [18]) is being developed as part of the Software Based Systems work area of the M-VCE project, a consortium comprising 7 UK universities and 28 companies representing the major international telco operators and service providers [13]. Software agents have been chosen as the key underlying technology for the development of services aimed at surviving the transition between

3G and 4G (the generation of networks that will follow 3G). Hence, great effort has been dedicated to the development of a suitable, that is efficient and secure, middleware layer for a controlled, managed execution of MAs.

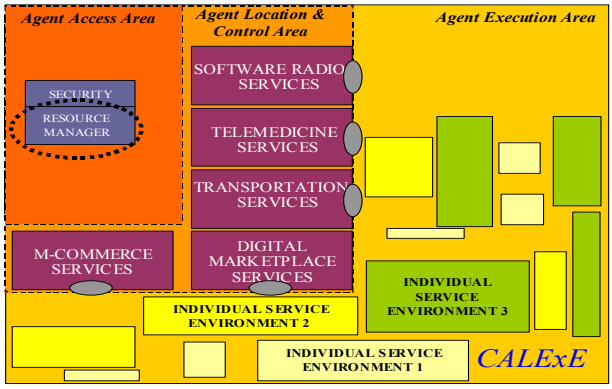

Fig. 1. CALExE system architecture.

The CALExE system architecture is depicted in Fig. 1. CALExE is a common platform where agents can be located and where they can interact to extend the variety of services available to mobile users. It provides computational resources as well as an environment for agents to execute and interact. CALExE offers management functionality for a controlled execution of agents in terms of performance and security. It is characterized by a modular architecture comprising three areas, each one related to particular aspects of the distributed system to be managed and providing important management functions for the various mobile distributed entities. These areas are:

- **Agent Access Area** is mainly devoted to controlling the access to resources and services provided by the platform. This area is responsible for the management of the physical resources under the control of the management platform. It is also responsible for the authorization and determination of access rights for the execution of different distributed entities, including both static and mobile agents. This area also manages the controlled access to remote resources and execution environments allowing, for instance, optimizing the execution of particular services by controlling the location of their sub-components or MAs. The resource manager enclosed in the dotted oval is described in further detail in the following sections.
- The **Agent Location and Control Area** is mainly concerned with the discovery, location, registration and control of the services provided and advertised by CALExE. This area is also responsible for monitoring the conditions in which the service is delivered – QoS monitoring is, for instance performed at this level. These functionalities are out of the scope of the paper (further details can be found in [3], [18])
- The **Agent Execution Area** provides a secure execution environment where agents can execute, negotiate, and act/react to particular events in the system. These

functionalities are out of the scope of the paper (further details can be found in [19] [20])

The dynamic resource management system described in Section 4 below uses as a reference the services and case studies that have been specified and are currently being developed by M-VCE. These are detailed in [14] [21] and briefly summarized below:

- **Home service** – Services in the home provide the user with a comfortable and secure environment, with easy access to features and intuitive responses, to enhance the user's productivity and general quality of life. Many tasks in and around the home can be performed by a set of co-operating mobile intelligent agents, acting on behalf of the user. An example scenario that highlights the emerging aspects of future generations wireless networks in the context of an automated home is where mobile terminals, wearable computers, appliances and personal area network (PAN) devices communicate with each other in an intelligent and autonomous way. Issues such as interoperability, uniform service provision, dynamic resource allocation and security pervade the whole of a distributed system.
- **Transportation** – The transportation scenario shows the chain of events during the travel of an executive. Travel consists of several stages where executive travels by different means of transport – car, plane, train and bus – continuously updating their travel details and ETA (estimated time of arrival) and using different services. Services can also be used autonomously by user representatives on behalf his/her behalf.
- **Mobile multimedia** – This scenario concerns the provision of multimedia, location-dependent or location-independent services. An example is and 'infotainment' services delivered to a user while on the move.
- **Telemedicine and Assistive Services** – Telemedicine provides an example of services, which have high importance but require ubiquitous availability. This scenario reflects the social role that mobile communications can play in achieving independent living for all. It has much in common with the first scenario but adds to it a human dimension.
- **Mobile commerce** Electronic commerce – fixed or mobile – between all combinations of business or people is hyped widely and loudly. Solutions exist for every scenario but it is not clear if they also solve known problems, or if they raise new issues yet to be addressed.

Fig. 2 shows an example of a scenario where a user requires a mobile service through a User Agent (UA). A mobile service can be composed of 'static' and 'mobile' components. While the static part of the service can be configured and renegotiated in terms of computational resources the mobile one can also be moved to exploit better resource availability. In the picture a number of CALExE subsystems, (or simply CALExEs) are distributed across the networked system while also being monitored by the monitoring agents. CALExEs can either be positioned in routers or, more realistically, in their vicinity in order to exploit the knowledge that routers have of the network. Each monitoring agent will monitor one or more CALExEs within its area of competence, holding information on their position and resource availability. Monitoring agents also know the location of the other peer-entities, forming an overlay logical monitoring network. Fig. 2 shows an example where the UA, after reaching the nearest CALExE, interacts with the static agent. From there, the MA can

subsequently decide to autonomously migrate or, alternatively, it can be forced to migrate by the resource manager, as illustrated in Section 3.

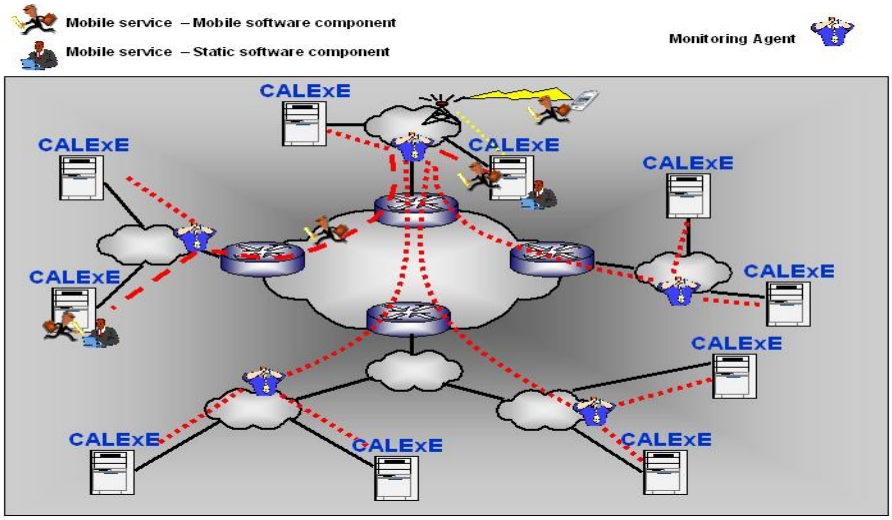

Fig. 2. Mobile Service example.

It may be important mentioning that the above scenarios are not meant to cover all possible situations; they have been specified with the aim of illustrating example applications and use them to extract requirements for the design of the CALExE middleware system. Each scenario has been used to identify specific issues, constraints and requirements on CALExE and, in particular, on the resource management system described below.

3 Dynamic Resource Management System

3.1 System Requirements and Functionality

The CALExE system consists of a number of distributed CALExE sub-systems that replicate some or all of the functionality of Fig. 1. For instance, the execution environment and the functionality exposing local resources to the resource manager should be present at every node running agents/services. The resource manager will use relevant information about the availability of distributed resources to trigger a variety of load-balancing actions. A typical action is, for instance, the reconfiguration of a service through the dynamic re-location of one or more MAs, in response to a scarcity of resource in current MA locations. Another example is the location of MA service components in a CALExE sub-system that acts as a proxy to a thin mobile terminal that could not otherwise execute a resource-intensive MA.

The resource manager plays therefore the key task of configuring the mobile services, matching the service resource requirements with the run-time resource availability. It also provides critical run-time resource availability information to those MAs that enjoy a certain level of autonomy and can, hence, self-relocate for load-balancing purposes. Therefore, the manager will not only manage the access and allocation of computational resources within each individual execution environment, but also determine, in real-time, the particular location where each service (or MA component) should be executed. In this case, the management ability is decoupled between the agents, that carry their service tasks, and the Dynamic Resource Management system that triggers agent migration for load balancing purposes.

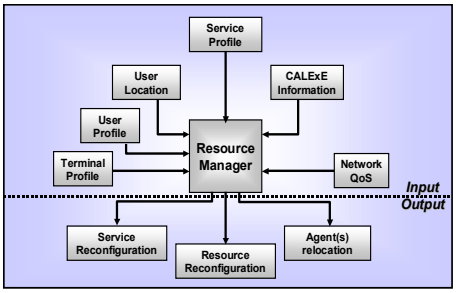

Fig. 3. System Input/Output model.

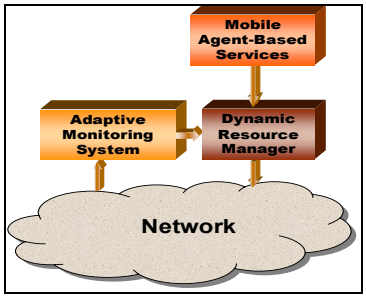

Fig. 4. Management System.

CALExE supports context-aware, adaptable services and has, thus, been designed to consider contextual information as a way to trigger service adaptation, again, through agent mobility. The input parameters to the resource manager are summarised below (Fig. 3):

- **User profile & SLA**: The user preferences and quality expectations are mapped onto an SLA (Service Level Agreement) agreed among user, service provider and network operator (NO).
- **User Location**: if the service is offered by a different entity that the NO, the NO should provide user location information which is fundamental to most context-aware, location-driven services. The recent developments in the process of standardizing open network interfaces (e.g. OSA/Parlay, web services) suggest that this requirement is feasible [22].
- **Terminal profile**: the knowledge of terminal capability is fundamental to most adaptable services in order to establish the best way of delivering the service. Memory or other terminal resource constraints are used as an input by our resource manager in order to best configure the service. Terminal capability information can be dynamically obtained using, for instance, the CC/PP protocol [23].
- **Service profile**: this input provides a broad model of the service requirements in terms of resource consumption (e.g. computation, storage, memory, QoS, etc). This information is associated to services on the basis of an accurate statistical analysis.
- **CALExE Information:** each CALExE subsystem includes an execution environment and an associated state/availability of the resources. Information collected from each CALExE sub-systems:

- **CALExE Location**: the management system uses the location of execution points when deciding on how to locate/re-locate MAs for load balancing purposes. Latency minimization is achieved by choosing appropriate nodes for MAs.
- **CALExE Workload**: the workload of each execution environment has to be supplied to the management system in order to extract the computational resources available and trigger due actions.
- **Local routing table**: routing tables are used to indirectly get the status of the network and estimate distances/latencies among CALExE subsystems.

- **Network QoS**: this relies on network layer mechanisms such as DiffServ or MPLS aimed at supporting the required QoS.

The effects of the manager can manifest themselves in different forms, depending on the value of the above set of input parameters (Fig. 3). One possibility is to re-configure the service, for instance, by renegotiating its QoS requirements in face of QoS degradation. QoS can be pursued through management policies aimed at preserving the overall system and network resources, while meeting the range of QoS levels acceptable to the user.

For this reason, in CALExE services/MAs are allocated a specified amount of resources, in terms of computation, memory, storage and network resources. Another possibility is to react to limited QoS by requesting an increased level of resource allocation to the CALExE system.

An alternative to service and resource reconfiguration is given by run-time MA relocation, that is triggering the migration of one or more MA service components. This operation is aimed at load-balancing the resources used across the CALExE subsystems.

Our approach to dynamic resource allocation and agent location is described in the following two sections. The overall system functionality, as depicted in Fig. 4, consists of two main blocks, the monitoring system and the resource manager. Our services rely upon the latter that, in turn, relies on an efficient monitoring system capable of coping with the dynamics and scale of the CALExE environment and its MA services.

3.2 Adaptive Monitoring System

The requirements that the CALExE middleware poses on the monitoring system are quite stringent. First, the resource management system relies on timely information including the state of the resources available through CALExE. Service configuration and MA location are in fact closely related to the availability of network, storage, memory and processing resources across CALExE subsystems. Each of those resources dynamically varies over the time, depending on the service load, user location, user QoS demand and so forth.

Our monitoring system keeps track of all the above parameters, feeding their values to the management system both periodically and following and event-based approach. Scalability and dynamics are therefore the challenges faced by the monitoring system.

After considering various approaches, we finally decided to develop our own monitoring system that could address those challenges. Because we could rely upon

an MA execution environment (i.e. the one used for MA services) we were able to design a fully MA-based monitoring system that would combine the advantages of distribution, activeness, adaptability, and reactivity. We have already reported our MA-based monitoring system in [2], assessing its scalability and adaptability to changes in the network state (e.g. faults, congestion, etc). Therefore we shall only give here a brief overview of the system (the interested reader can find more details in [2]).

In our system, monitoring of fixed or mobile resources (including nodes, MA service components, CALExE subsystems etc) is carried out by MAs. The monitoring process consists of two phases. Initially we need to deploy the monitoring MAs, depending on the status of the network and on the location of the target objects to be monitored. This is equivalent to partitioning the monitored system into sub-partitions that are, in turn, monitored by individual MAs. Our approach to the calculation of the number of agent and their location is to exploit information that is readily available in the system. Our monitoring system relies on the routing tables stored in the network, which are maintained up-to-date by the network routing protocols

Subsequently, upon agent deployment the agent system needs to be able to self-regulate in order to adapt to changing conditions. This is achieved by triggering agent migration in a controlled fashion to avoid instability due to continuous agent migration. Again, agents compute their location on the basis of routing information that, in turn, reflects the network state.

3.3 The Resource Manager and run time service adaptation

The resource manager is a core component of the CALExE middleware and is present in each CALExE subsystem. It performs the important tasks of *admission control*, *resource allocation*, and *resource discovery*. In order to describe its operation, let us suppose that we want to deploy a mobile service, composed of a combination of static and mobile agents. Fig. 5, illustrates the various steps starting from the submission of a job request (i.e., the intention to deploy a service) until service deployment. It depicts the logic executed at two different CALExE subsystems. The 'local' CALExE is the one that receives the job request in the first place. The 'remote' CALExE represents a generic candidate subsystem for the execution of some components of the mobile service under scrutiny.

The execution of service components within CALExE subsystems has to be endorsed by their respective resource managers. When a job request is received by a CALExE subsystem, this goes through an *admission control* procedure aimed at establishing whether sufficient resources are locally available. Admission control relies on information provided by the monitoring system and attempts a matching between the service profile (in terms of estimated resource requirements) and the local resource availability. It should be mentioned that service requirements might also specify that a maximum latency or a minimum throughput must be guaranteed between two different service components.

Those service components that pass the admission control procedure are instantiated and executed locally, following an appropriate *resource allocation* procedure. Static agents will, then, be forever bound to the particular CALExE subsystem that has initially accepted them. Instead, MAs can still be re-located at run time, either as a result of an action triggered by the local manager (e.g. because of

scarcity of local resources) or as a result of a decision autonomously taken by the MAs themselves.

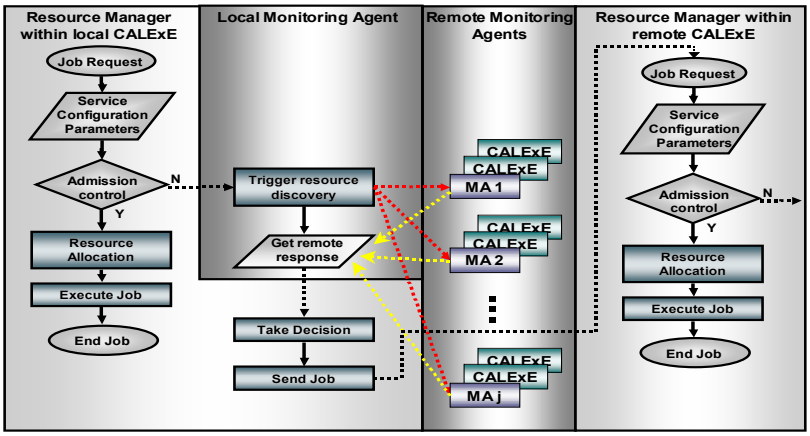

Fig. 5. Resource management system.

Any job that is not admitted locally triggers a *resource discovery* procedure through which the local CALExE finds out about alternative subsystems that have the potential to admit those jobs. Again, the monitoring system realized through local and remote monitoring agents (MA_1, MA_2, MA_j,) assists and provides information during this phase.

The discovery procedure will generally result in one or more candidate CALExE subsystems for each of the candidate jobs (i.e. MAs in search of a CALExE). The local manager can then employ different strategies aimed at selecting the best match between jobs and CALExE subsystems. We have considered three of them:

1. **Fastest first**: in this case each job is matched with the CALExE subsystem that responds first. The fastest reply may be received by the entity that is logically closer to the requestor. This may therefore be useful when trying to minimize latency between local and remote CALExEs.
2. **Largest availability**: in this case jobs are matched with CALExEs holding the largest availability of resources. This may assure an even utilization of resources across the system.
3. **Compromise distance-availability**: this solution tries to compromise between the above approaches, seeking a good load balance as well as latency minimization.

In such a way the local manager eventually delegates each of the pending jobs to the appropriate remote CALExE subsystems that will, again, put the job through the same admission/allocation/discovery procedure (right hand side of Fig. 5). MA delegation will continue until a suitable, unloaded CALExE subsystem is found. In our simulations this process tends to terminate after one or at most 2 iterations, depending on the level of load of the overall system. Clearly a heavily loaded system will require a longer MA allocation procedure.

4 System Evaluation

4.1 Methodology

The evaluation of the above resource management system has been carried out through simulation, developing three essential ingredients: a model of mobile services; an implementation of the algorithm of Fig. 5; and a design of realistic network conditions.

Our service model reflects the creation of composite services with static and mobile components/agents, the interactions among agents, and the resource utilization by the agents in terms of network links, CPU cycles, memory, and disk. Our service model reflects the requirements and constraints of the five services that are currently being developed within the M-VCE project and are documented in [14]. In this way we shall eventually be able to assess our resource management system for those services.

In order to carry out simulations under realistic network conditions, the algorithm has been implemented over the JavaSim network simulation which provides the underlying support for wired and wireless networking [15]. IP and TCP have been adopted as the basis for simulating routing and transport protocols, respectively. We have also enhanced the simulator with MA capability needed for the realization of the monitoring system and for the deployment of MA services. Functionalities include agent creation, cloning, migration, termination and so on.

The GT-ITM topology generator [16] was used to create realistic, Internet-like, transit-stub topologies. The simulations have been executed for topologies of 25, 40, 75, 100, 150, 200 and 300 nodes. To ensure statistical significance, simulations have been repeated 20 times for each of the above network sizes. That is, we have randomized the simulation process, generating families of 20 topologies at a time, characterized by comparable topological features (e.g. average node degree, number of nodes, etc). In this way we could assess the sensitivity of our metrics to network size, using the number of node as network size parameter and isolating other effects.

An important parameter of the system is its ability to adapt to variations in the system load. The management system achieves that by dynamically monitoring system resources and triggering service reconfiguration or MA migration. The metric used to assess the system performance was the total hop distance, computed as the sum of all the hops or 'link legs' involved in the service communication among all the service components. For instance, if the service is realized through two intercommunicating MAs, the total hop distance would be the number of links that interconnect those MAs. As the system/network evolves, the distance between those agents may increase, e.g. as a result of link failure, congestion and re-routing. It is the management system responsibility to trigger appropriate re-location, leading to reducing overall distances and, consequently, reducing overheads.

4.2 Results

Fig. 6 illustrates the behavior of a typical mobile service for an increasing number of congested links. We have measured the overall total hop distance (i.e. involving all service components) in two different cases: a) service realized with static agents only

(no run time adaptation through agent migration); and b) MA based services (dynamic MA relocation triggered by the management system).

As expected, system congestion results in an increase in total hop distance which, in turn, means an increase in traffic and response time. Our adaptation strategy, however, succeeded in significantly reducing the rate of performance degradation (as indicated by the difference in slope between the two best linear fit lines).

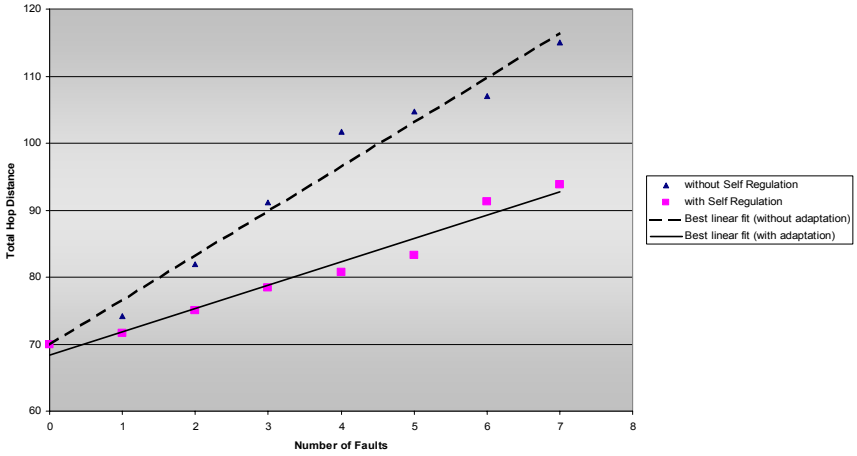

Fig. 6. Performance results of the resource management algorithm.

Fig. 6 depicts the results for a 40-node system and a family of Internet-like topologies. However, when we repeated the experiments for larger systems and for a variety of other scenarios the results were qualitatively equivalent.

The background load of the system and its overall congestion significantly affected the service initial deployment time only under extreme conditions, i.e. when the amount of resources available through CALExE could not meet the service resource demand. Instead, under average loading, MAs were mostly deployed in 1 or 2 iterations, depending on the relative number of CALExE subsystems with respect to the size of the system.

5 Related work

Various researchers have used MAs as well as distributed resource allocation techniques to support services. In [1] the use of MA technology in a wireless environment to address QoS issues through a load balancing solution is described. The approach performs load balancing of processes in different servers, assures a QoS to specific classes of users and supports an algorithm to predict the user movement within different cells. Advantages of this work include the agents' capability to trigger adaptation of applications on behalf of the customers and the implementation of resource-state-based policies in a dynamical fashion. It is worth mentioning that this

solution works more at network level, while the work presented in this paper is at service level.

In [4], the authors present a policy-based software architecture that implements application QoS management. The approach is dynamic in that the application is started with an initial resource allocation that can be dynamically incremented by the manager if needed. One limit highlighted by the authors is that if there are several applications on the same host with QoS requirements, the corrective actions may not be feasible.

In [5], the authors provide a resource allocation technique using a multi-agent system called *Challenge*". Challenger consists of agents that individually manage local resources. These agents communicate with one another to share their resources for an efficient use of the system. Although generally useful for resource allocation in distributed systems, the focus of the work is not aimed at systems with tight real-time requirements.

Smart Reminder shows how the interoperation of different agents focused on single tasks can realize complex applications [6]. This system is composed of a set of personal agents that support their user in all situations where people come together spontaneously, e.g. an encounter on a corridor. So the system tries to establish a framework where different autonomous agents can easily work together to deliver aggregated benefits to their users. The authors claim that these systems must be as autonomous as possible so that the time and effort saved is not countered by excessive co-ordination overheads.

The Odyssey architecture consists of a set of extensions to operating systems to support mobile, adaptive information access applications [7] [8]. Its central idea is that the system monitors resource availability, notifies applications of the relevant changes and enforces resource control. This approach extends the system-level functionality working therefore mostly at operating system level.

Paper [9] presents a middleware architecture called Agilos, which is used for QoS adaptation, achieved at two levels through the components adaptor and configurator. The adaptor works at system level and is in charge of specific type of resources, e.g. CPU or network bandwidth. In this way a fair and stable distribution of the available resources among concurrent applications is ensured. The configurator instead works at application-level and serves one application at time. After getting applications information it maps adaptation decisions made by the adaptor to application-specific parameter-tuning or reconfiguration choices within the application.

The TLAM (Two Level Actor Machine) is a two level model, which uses base actors and meta actors [10]. Base actors carry out application level tasks while meta actors are part of the runtime system, which provides a set of core services for distributed systems. These are, for example, communication, resource management, remote creation, migration, load balancing, scheduling, synchronization, and replication. Meta actors can access and change information within a base actor.

RAJA is an agent infrastructure for resource-adaptive systems that addresses resource-adaptivity by promoting agent interoperability [11]. Adaptation decisions can be negotiated between the agents or made corporately at runtime. It is a multi-level reflective architecture, which separates non-functional resource management from application functionality. The distinction between basis agents and their controllers allows structured programming and eases the transformation of resource-unaware legacy systems into resource-adaptive systems. The separation between

controllers and meta agents isolates application- from system-level resource management and advocates autonomy of agents.

Finally, Grid gives the possibility of using networks of computers as a single, unified computing resource [12]. It is possible to cluster or couple a wide variety of resources including supercomputers, storage systems, data sources, and special classes of devices distributed geographically and use them as a single unified resource, thus forming what is known as a "computational grid". Within Grid resource managers or applications must tailor their behavior dynamically so as to extract the maximum performance from the available resources and services. The grid resource management systems must dynamically trade for the best resources based on a metric of the price and performance available and schedule computations on these resources such that they meet user requirements.

The abovementioned work attempts to address the requirements of future 3G services. Some work does not use the mobile agent technology, which makes it different from our approach. When the agent paradigm is used their approach relies either on agent autonomy or on multi-agency. The CALExE middleware can be seen as combining autonomy with more conventional distributed management.

6 Conclusion and future work

The latest technological advances in the area of fixed, mobile and cellular communication are opening the avenue of advanced services that were unimaginable just a few years ago. In this paper we have looked at some of the issues related to effective service control. We have taken the perspective of the MA technology as a key enabler of mobile service realization and control, focusing on important agent location control aspects.

Our work started from the assumption that in an increasingly mobile environment, services are mobile and so should be the control and management system. We started by developing an MA-based distributed, adaptable monitoring system, as reported in [2]. We then moved onto designing and prototyping some reference advanced services in the context of the M-VCE project [13], leading to a more abstract model that would be suitable for evaluation purposes. Mobile services are naturally implemented as a combination of static and mobile components. Hence, MAs are one of the best candidates for service realization. Having come to this conclusion, we are then open to interesting ways of controlling and configuring services in such a way as to make use of distributed resources that should always be conserved for scalability purposes.

This paper represents an important first step of our investigation of a hybrid, service management approach. We are looking at how to combine agent mobility and autonomy with a more conventional approach in which it is the management system that triggers management actions. Mobility, autonomy and system-initiated management are not always compatible and may lead to instability or integrity problems. Some of these problems have already arisen during our experimentation work and will surely require further attention. However, our initial results of the proposed resource management algorithm indicate that MA-based service management is certainly an interesting research avenue.

References

1. G Anastasi, et al., *An agent-based approach for QoS provisioning to mobile users in the Internet*, Proc of SCI2000, Orlando (Florida-USA), July 2000.
2. A. Liotta, G. Pavlou, G. Knight, *Exploiting Agent Mobility for Large Scale Network Monitoring*, IEEE Network, special issue on Applicability of Mobile Agents to Telecommunications, Vol. 16, No. 3, IEEE, May/June 2002.
3. O Lazaro et al., *Management System Analysis – Security, Performance and Integrity Issues*. Deliverable D1.4 M-VCE project, (www.mobilevce.com), Oct 2002.
4. G Molenkamp, et al., *Distributed Resource Management to Support Distributed Application-Specific Quality of Service*, Proc of the 4th IFIP/IEEE International Conference on Management of Multimedia Networks and Services, Chicago, Illinois, October 2001.
5. A. Chavez, et al., *Challenger: a Multi-Agent System for Distributed Resource Allocation*, Proc of the 1st International Conference on Autonomous Agents, p. 323-331, 1997.
6. F. Kargl et al., *Smart Reminder - Personal Assistance in a Mobile Computing Environment*, Workshop on "Ad hoc Communications and Collaboration in Ubiquitous Computing Environments" at CSCW 2002, New Orleans, USA, Nov. 2002.
7. B. Noble, *System Support for Mobile, Adaptive Applications*, IEEE Personal Communications, February 2000.
8. B. Noble et al., *Agile Application-Aware Adaptation for Mobility*, In Proc. 16th ACM Symposium on Operating System Principles, 1997.
9. B. Li et al., *Middleware QoS Profiling Services for Configuring Adaptive Applications*, In Proc. of Middleware 2000.
10. N. Venkatasubramanian, C. Talcott, *Reasoning about Meta Level Activities in Open Distributed Systems*, In Proc. of ACM Principles of Distributed Computing, 1995.
11. Y. Ding et al., *a resource-adaptive java agent infrastructure*, Proc of Agents 2001, Montreal, Canada, 2001, pp. 332 – 339.
12. M Baker et al., *The Grid: International Efforts in Global Computing*, International Conference on Advances in Infrastructure for E-Business, Science, and Education on the Internet, SSGRR2000, L'Aquila, Italy, July 31-August 6, 2000.
13. Mobile Virtual Centre of Excellence (MVCE) project. www.mobilevce.com
14. Turner, P. and Lloyd, S. (Eds.) *Agent System Specification*. Deliverable R5, M-VCE project, (www.mobilevce.com), May 2001.
15. The JavaSim simulator, www.javasim.org
16. Source code of GT-ITM, available at http://www.cc.gatech.edu/projects/gtitm/
17. W. Tuttlebee, *Mobile VCE: the convergence of industry and academia*, IEE ECEJ, 12 (6), 245-8, December 2000.
18. O. Lazaro, J. Irvine, D. Girma, J. Dunlop, A. Liotta, N. Borselius, C. Mitchell, *Management System Requirements for Wireless Systems Beyond 3G*, Proc. of the IST Mobile & Wireless Telecommunications Summit, Thessaloniki, Greece, 16-19 June 2002.
19. Borselius, Security for agent systems and mobile agents, in: C. J. Mitchell, editor, Security for Mobility, IEE Press (to appear).
20. N. Borselius, Mobile agent security, Electronics & Communication Engineering Journal, October 2002, Volume 14, no 5, IEE, London, UK, pp 211-218.
21. Turner, P. J. *et al*, (2002) Scenarios for Future Communications Environments. Technical Report ECSTR-IAM02-005, Department of Electronics and Computer Science, Southampton University.
22. The Parlay Group, Parlay Web Services, Overview Status, Public Version 1.0, 31 October 2002, www.parlay.org
23. W.Okada, et al, *Applying CC/PP to User's Environmental information for Web Service Customization*, Tenth International World Wide Web Conference, June 11, 2001, Hong Kong.