Mobile Agent Solutions for Accounting Management in Mobile Computing

Paolo Bellavista, Antonio Corradi, Silvia Vecchi
Dipartimento di Elettronica Informatica Sistemistica - University of Bologna
{pbellavista, acorradi, svecchi}@ deis.unibo.it

Abstract

The convergence of mobile telecommunications and the Internet global system forces to reconsider traditional client/server solutions for network and systems management. The paper claims that accounting in the mobility-enabled Internet requires support infrastructures hosted in the fixed network. These infrastructures should monitor, control and register resource consumption locally within the domains where users/terminals dynamically move to, without requiring continuous connectivity with remote and centralized accounting home managers. In addition, the paper shows that the Mobile Agent (MA) technology is suitable to overcome the limits of traditional accounting solutions in several mobility-enabled usage scenarios. MAs can maximize locality in accessing monitoring data, can enable accounting even in case of temporary disconnection, can install new monitoring/control behavior dynamically, and can support session-dependent solutions. The paper finally presents the design and implementation of the MA-based Middleware for Mobility Accounting Management (MAM^2), together with some use cases showing the advantages of the MA adoption.

1. Introduction

Advances in mobile telecommunications and portable device miniaturization propose new service scenarios where users, access terminals and even service components can geographically move during provisioning. This forces to face several technical challenges at different levels of abstraction, from network connectivity to location tracking, from device-dependent service adaptation to location-dependent service provisioning. In addition, mobile computing and the Internet are converging and likely to merge, with the aim of providing a unique support infrastructure, capable of hosting network elements, service components, and fixed/mobile access terminals [1].

This mobility-enabled Internet also changes the perspective of network, systems and service management. Traditional management solutions are tailored to fixed infrastructures of network elements, and management research activities have proposed architectures to deal with geographic distribution and heterogeneity of resources and service components. IETF and OSI have proposed management models based on Client/Server (C/S) interaction [2, 3] and on several variations of it. If one can organize hierarchies of C/S components to achieve decentralization and scalability, the interaction among management entities is usually statically determined for clients and servers in exchanging remote information. These solutions do not fit well global systems in rapid evolution where the entering/exiting of unknown heterogeneous mobile components is usual. These cases ask for the possibility of dynamically installing new management behavior without any service suspension. Some significant management research efforts investigate the support of code mobility in a secure and interoperable way. They have been tagged with different names, such as Management by Delegation [4], Active Networks [5], Programmable Networks [6], Mobile Agent-based management [7, 8].

Among the different activities belonging to the management domain, accounting can be defined as the process of monitoring, controlling and registering the amount of administered resources that an authorized user (or a user group) exploits. Accounting becomes particularly relevant and challenging in mobile computing. Accounting solutions require tracking user locations in a global environment and coordinating remote resources, possibly located in different networks with heterogeneous mechanisms for monitoring and access control. Accounting systems should face temporary disconnections and network partitioning. In addition, they should organize control strategies on the base of previous user actions in remote network localities, e.g., to permit the access to one service component if the client has not used up the time quota reserved for that service.

The current hardware/software limits of several categories of mobile devices, from personal digital assistants to programmable cell phones, impose an infrastructure over the fixed Internet to support specific issues for their network connectivity and service access [1]. In the same way, a support is necessary to provide accounted services in any network locality willing to open its resources to mobile accessibility. The paper claims that mobility-enabled accounting management requires a dynamic and extensible support infrastructure, capable of evolving during service provisioning depending on client mobility. The infrastructure should be in charge of monitoring, controlling and registering resource consumption locally where mobile users/terminals move to, without requiring con-

Proceedings of the Seventh International Symposium on Computers and Communications (ISCC'02)
1530-1346/02 $17.00 © 2002 **IEEE**

tinuous connectivity with remote and centralized home accounting managers.

The Mobile Agent (MA) technology is suitable and effective for accounting in the mobility-enabled Internet. MAs can locally access monitoring data about resource usage, can enable accounting even in case of temporary disconnection, can install new monitoring/control behavior dynamically, and can support session-dependent accounting solutions. The paper discusses how MA solutions permit to dynamically develop and deploy the accounting infrastructure only when and where needed.

Along these guidelines, we have designed and implemented the MA-based Middleware for Mobility Accounting Management (MAM^2) within the Secure and Open Mobile Agent (SOMA) platform. The SOMA code and additional information about the platform are out of the scope of the paper and available for download at: `http://lia.deis.unibo.it/Research/SOMA`. On the base of a support layer for monitoring, storage and events, MAM^2 is organized in terms of accounting MAs that follow client movements in the network. Accounting MAs maintain the same location of their mobile users, thus enabling local monitoring, control and registration of resource consumption. MAM^2 integrates with our previous research in network/systems/service management and mobile computing middlewares [1, 9].

2. Accounting Management in Distributed Systems

The distributed management of networks, systems and resources has originated a wide range of solutions, with very different approaches and producing different implementations. Notwithstanding, the complexity of addressed issues and the evolution of network and systems technologies still call for further investigations.

On the one hand, we must consider the rising relevance of the Internet as the primary service provision carrier. This introduces novel requirements, typical of commercial networks, such as the adoption of billing policies, which require new functions notably absent in the Internet since its non-commercial origin [10]. For instance, the availability of differentiated quality levels in service provisioning is a strong motivation for charging: price differentiation according to offered quality is the only way to prevent resource monopolization of greedy clients.

On the other hand, we must consider the changing mode of service access. Advances in cellular telecommunications and device miniaturization increase the number and kinds of portable devices with Internet connectivity. User/terminal mobility to unknown and unpredicted network localities during service provisioning makes centralized C/S accounting management unsuitable, and pushes towards the investigation of novel solutions.

Accounting management addresses the issues of metering, storing and processing information about resource consumption. Metering is the process of tracking data on resource usage, performed with different collection modes (polling, event-driven, trap-direct polling [11]). Storing is the recording of metering data on stable storage, either in the same locality where information is collected or in a remote administration domain with the additional transport and security issues. Metering and storing are the basic steps for any other accounting activity.

Accounting has a wide range of employment, from capacity planning to user billing, from resource access control to denial-of-service attack detection. Therefore, the way of processing metering data can be very different depending on the accounting aims. In capacity planning, data are statistically analyzed to forecast future usage and to plan system expansion and reengineering. In auditing, they are employed to verify the correctness of a process, i.e., to determine the actions actually carried out and whether they conform to the expectations. In usage-sensitive billing, metering information is maintained to charge users for accessed services, and processing includes the assignment of consumption to responsible users with no possibility of repudiation. In pricing, the goal is defining best prices to charge for different classes of services, usually depending on complex economic models.

The above classification suggests an accounting architecture organized in two layers. A base resource-level layer involves basic accounting activities, such as metering and storing. An upper service-level layer involves goal- and application-specific accounting aspects. Upper layer accounting components can vary in number and types, and often need to interact and coordinate with each other and with other system components/services, e.g., authentication and authorization modules. The upper layer can include a policy repository to describe interoperable high-level rules about resource consumption, an event manager to define, throw and handle asynchronous events triggered by the evolution of the managed system (user login/logout, resource limit overcoming, changes in billing policies, etc.), and accounting entities to operate management actions for policy enforcement.

3. Existing Solutions for Accounting Management

Currently proposed solutions for accounting adopt different perspectives. The network management community stresses architecture and communication aspects, with the goal of minimizing latency and generated traffic. The Internet community, instead, is mainly focused on security and reliability issues.

According to OSI, accounting management is one of the five functional areas of network management, together with configuration, fault, performance, and security. Both

the OSI Common Management Information Protocol (CMIP) [13] and the IETF Simple Network Management Protocol (SNMP) [2] assume a C/S architecture where a central manager coordinates the operations of managed processes, called agents, running on the managed devices, with the reverse interaction obtained via trap-direct polling. These management solutions face also accounting aspects and have been widely deployed in intra-domain accounting systems [13, 14]. However, in global deployment scenarios they can exhibit limits in efficiency and latency, thus not fitting situations with strict constraints on processing delay or overhead, e.g., usage-sensitive billing applications with fraud detection requirements [11].

Internet accounting involves authentication and authorization. Authentication includes how to validate the user identity in order to allow/deny access to managed resources and to securely associate users with resource consumption metering data, ultimately preventing repudiation. Authorization defines the permissions granted to users once their access has been authorized, and is crucial for auditing and control. The tight relationship between the above aspects induced IETF to create the Authentication, Authorization, and Accounting (AAA) working group, which specifies frameworks to coordinate AAA activities over multiple network technologies and platforms. The most widely diffused AAA protocol is Remote Access Dial-In User Service (RADIUS) [15]. RADIUS is a UDP-based C/S protocol that provides event-driven authentication and accounting. RADIUS authorization and configuration exploit a database of attribute/value pairs and the transmission of security-sensitive data is encrypted with shared secret keys. RADIUS was originally designed for devices in small static-topology networks and supports simple C/S-based management actions and a limited number of users and network elements.

The rising demand of secure scalable AAA for Internet users is pushing the research activity for new models of solution. IETF has recently proposed Diameter, a follow-on to the RADIUS protocol, designed to offer a scalable foundation for introducing new AAA policies and services over existing (PPP) and emerging (roaming, mobile IP) network technologies [16]. The main idea in Diameter is to provide a base lightweight protocol that includes the functions common to any supported service, while application-specific functions are provided through an extension mechanism. Unlike RADIUS, Diameter operates over the Simple Control Transmission Protocol that provides reliable communications via a well-defined retransmission and timeout mechanism. The basic version of the protocol defines message format and a set of primitive functions, and assumes a peer-to-peer event-driven communication model. The main Diameter extensions are Strong Security, based on public key encryption, Accounting, which stores usage information in attribute-value pairs and permits real-time transmission, and Mobile-IP, briefly described in the following. Apart from Diameter, existing accounting frameworks have not been designed for mobility contexts, and even if they try to add ad-hoc mobility patches, they do not provide solutions fitting the novel requirements of the mobility-enabled Internet.

3.1. First Solution Proposals for Accounting in Mobile Computing

Service provisioning in the mobility-enabled Internet ask for novel solutions for accounting management. Only a few research proposals have recently addressed the challenging issues of accounting in contexts of user/terminal mobility.

Recent SNMP versions add the concept of SNMP engineID, a globally unique identifier for the accounting data source. The SNMP engine is the component, included in traditional SNMP agents/managers, that constructs SNMP messages, provides security functions, and maps to the transport layer. EngineID allows an engine to be uniquely identified, independently of the location where the SNMP agent/manager is attached to the network, thus supporting the possible mobility of network elements.

Diameter was designed from the beginning to support mobility and its model follows the solution pattern of mobile IP. When a mobile user attaches to a foreign network, a foreign Diameter agent is in charge of her accounting. It sends a Diameter request to its local Diameter server; the server determines if the request can be satisfied locally or must be forwarded via a Diameter broker to the Home Diameter server in the user home network.

Although these first proposals represent significant evolution steps for accounting solutions, especially in recognizing the central role of user/terminal mobility, the paper claims the suitability of novel accounting approaches, capable of facing the wide range of issues raised by mobility in a global and open environment. We refer to issues such as dynamic extensibility of the support, local access to monitoring data, local control of administered resources, and operation autonomy with regards to remote and temporarily unreachable accounting managers.

4. Mobile Agents for Accounting Management in Mobile Computing

The merging of mobility with the Internet scenario has changed both service provisioning and systems management. Traditional solutions for fixed and local networks do not suit the new scenario where users, terminals and even service components can change their allocation during service provision. MAs propose an effective technology to support different forms of Internet mobility.

4.1. Mobile Agents for Mobile Computing

The MA technology is intrinsically connected to migration, i.e., the possibility for entities to change allocation, and the same is for mobile computing. Migration is a form of mobility that can answer the requirement of intrinsically distributed resources: a mobile entity can move close to its needed resources by preserving locality in operations. MAs can even change their allocation by following mobile devices/users in their de/attachment to the fixed network and maintain tightly *local* to their handling devices. An important MA property is the possibility of *maintaining the execution state* when detaching from a previous location and of restoring it at the new location. That can help in tracing mobile devices and in keeping track of their history while connecting/disconnecting. In summary, MAs are the infrastructure counterpart components for moving mobile devices because they can move close to the resources of the current device point of attachment, by preserving the achieved state [1].

In addition, mobile computing can take advantage of the possibility of *asynchronicity* between requests of user/terminal operations and their execution. For instance, wireless connections impose strict constraints on available bandwidth and on communication reliability, and force to minimize the device connection time. The MA paradigm does not assume continuous network connectivity because connections are required only for the time needed to inject agents from mobile terminals to the fixed network. That avoids continuous requests of remote operations that can stress to the limit both connectivity and bandwidth. Agents are autonomous and permit to carry on services even when launching users/terminals are disconnected and to give results back at their reconnection [17, 18].

Mobility requires *dynamicity*, as the modification and extension of the network infrastructure at run-time with new components and protocols to adapt to evolving service/user requirements. Run-time code distribution and dynamic resource binding are very similar in case of both MAs and mobile users/terminals. MAs greatly benefit from the additional flexibility of installing code during provisioning and of discarding it when no longer needed.

The paper mainly focus on accounting and this function can significantly benefit from the above MA characteristics. Of course, the MA technology also exhibits many other properties intrinsically important in service design and provisioning. Services in the mobility-enabled Internet call for visibility of the location of client users/terminals at provision time. Location awareness is crucial to enable application-specific optimization and to adapt services to currently available resources. *Location awareness* is typical of the MA paradigm that propagates allocation visibility up to the application level, thus simplifying the support of dynamic adaptation to the local situation [19]. In addition, the MA autonomy from users simplifies dynamic *personalization*: MAs, when following user movements, can facilitate the tailoring of services to personal preferences. For instance, a mobility-enabled application should face mobile user changes of location and should dynamically tailor its results to the current properties of network connections, to the user profile, and to the hardware characteristics of the access device.

Mobility also stresses the *security* issue, to authenticate mobile users/terminals, to authorize the access to system resources and to grant communication secrecy/integrity. MA research activities have proposed solutions for security problems, by granting the most suitable security level [20, 21, 24]. For instance, many MA systems integrate with Public Key Infrastructures to simplify authentication of mobile users/terminals. Finally, mobile users/terminals need *interoperability* when moving to unknown environments to interact with the resources/services available there. To face similar problems, the MA research has promoted interoperable and standard interfaces. For instance, some MA platforms provide compliance with OMG MASIF and FIPA specifications [22, 23].

4.2. MA-based Solutions for Accounting

Network and systems management proposes a relevant arena for demonstrating the effectiveness of the MA technology [7, 24]. In particular, several research activities have investigated MA solutions for network monitoring to reduce the data exchange overhead and to solve the so-called micro-management problem. The paper claims the relevance of the MA technology also to build accounting management infrastructures in global systems when dealing with user/terminal mobility. Accounting can benefit from locality of MAs to monitoring information in several cases. The most significant ones are to enable accounting even in case of temporary network partitioning, to exploit code mobility in the deployment of the accounting functions where and when needed, and to exploit state mobility to enable session-dependent accounting policies, as stated in the following examples.

Let us imagine an accounting scenario where an access device D moves from a network locality to another one. In traditional monitoring and accounting, a central fixed accounting manager A regularly polls the resources and services (or the support components in the locality in charge of monitoring them) used by D, in order to register accounting data and to possibly enforce control policies on resource consumption (see Figure 1). A is forced to access to the remote monitoring and accounting data, by generating overhead in network traffic and by imposing delay in case of D roaming in a global system. Improvements to this basic C/S scheme can stem from the addition of trap-based coordination functions and the organization of decentralized hierarchies of accounting managers. MA-based accounting managers, instead, can exploit co-

location with resources and service components to process locally monitoring data, and to react promptly to modifications in resource state by operating local management actions. The same applies to the case of user mobility, i.e., when the user U moves to a new network locality where she exploits a local access terminal D1, different from the previously exploited D2.

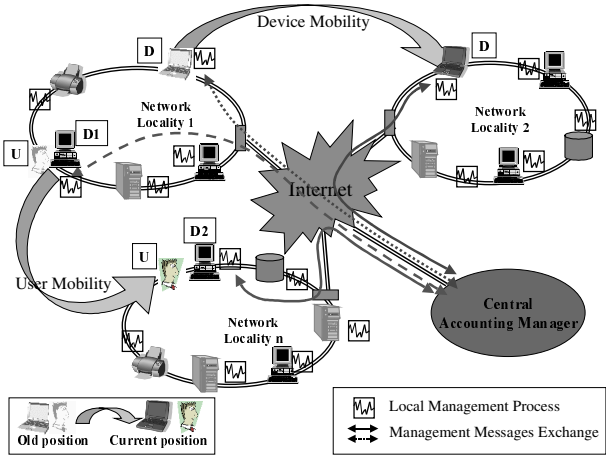

Figure 1. Traditional accounting: a central manager regularly polls monitoring components for controlled devices

MAs can help in overcoming critical situations, such in case of temporary disconnection due to link faults or to network congestion in the path between the controlling entity A and the current attachment of D. In the extreme case of network partitioning, MA-based accounting managers can autonomously and asynchronously go on monitoring, controlling and accounting for resource consumption in their current network locality, without any intervention of a remote manager A, as depicted in Figure 2. When network connectivity re-establishes or after mobile users/terminals have moved away from the disconnected locality, the MA accounting managers can decide to migrate to (or to coordinate remotely with) the central accounting entity, if any, to update the related accounting data. Let us also state that MA platforms often provide persistency services to temporarily suspend agents and to store them on persistent storage. This allows MA-based accounting managers to preserve accounting data reliably and not to waste system resources while waiting for asynchronous events of re-established connectivity. MAs can act as reliable persistent proxies both for the disconnected device D and for the remote accounting manager.

In addition, accounting managers cannot be statically implemented as large packed components capable of facing any possible situation, not to increase too much their code size. MA-based managers can modify/extend their behavior by exploiting code mobility during execution. MAs can permit to distribute new code where updating is required and can spread some disseminated code repositories by need, depending on the dynamically retrieved conditions of the working environment. That diffusion mimics the same usual deployment of MAs that employ code mobility mechanisms for dynamic and adaptive migration.

Advanced accounting management includes the possibility to enforce different policies for access control and pricing depending on session state. Mobile users/devices can be authorized to access system resources depending on global and previous access information, in the same usage session or according to previously agreed consumption thresholds. For instance, a user can be charged on the base of the amount of resources already consumed during the same session, within all different network localities of the same organization. The maintenance of session state and its dynamic migration are crucial for these forms of accounting, and they motivate further the MA adoption instead of other code mobility paradigms [26]. In fact, MA-based accounting managers can transport their reached accounting state to continue their operations at the new hosting locality, with no need to restart their execution from the beginning at any migration.

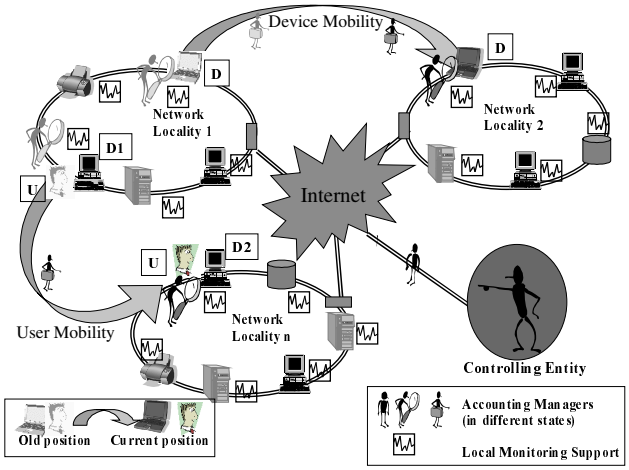

Figure 2. MA-based accounting: a central manager sends MA managers to administered nodes by need

The usage of the Internet as an open and global scenario requires accounting solutions capable of interoperating with different mechanisms and tools and of evolving dynamically without imposing any service suspension. Different network localities, even belonging to the same organization, can adopt different technologies to meter resource consumption, to process monitoring data for extracting concise accounting information, and to express control/pricing policies. For instance, one locality can adopt standard SNMP monitoring and access control based on password solutions. Another locality can use proprietary monitoring solutions to obtain also application-level monitoring information [10] and can control resource access depending on certificate-based user roles [21]. While waiting for a widespread standardization of

accounting APIs, the current heterogeneity requires accounting managers not only to integrate with a wide variety of solutions, changing from one locality to another one, but also to interoperate with different available legacy management components. MAs are suitable because of their interoperability research efforts [23].

5. Mobility-enabled Accounting in SOMA

SOMA is a Java-based MA platform that provides a layered infrastructure to assist in the design, implementation and deployment of MA-based services for global, open and untrusted environments. SOMA offers locality abstractions to describe any kind of interconnected system, from simple Intranet LANs to the Internet. Any node hosts at least one *place* for agent execution; several places are grouped into *domain* abstractions that correspond to network localities. In each domain, a *default place* is in charge of inter-domain routing functionality and integration with legacy components [1].

SOMA naming is based on care-of mechanisms to locate mobile agents and possibly mobile users connected to the system. Any SOMA agent can be traced via its care-of entity (*agent home*) maintained at the place of its first creation. Similarly, any mobile user has her care-of entity (*user home*) at the default place of the domain of her first registration. The default place is maintains the master copy of the user profile, as described in the following. The SOMA middleware transparently updates information at the homes of traceable agents at any migration and homes of traceable users at their connection/disconnection in different SOMA domains. For performance sake, SOMA allows also non-traceable agents/users not to pay the overhead of location tracking. All SOMA (mobile) entities have GUIDs, independent of their current position. MA GUIDs consist of the identifier of the corresponding agent home associated with a number unique in the home locality. This solution permits to identify immediately the agent home, without querying the SOMA naming service. User GUIDs are usually at a higher level of abstraction and defined externally to the SOMA system, e.g., taxpayer codes. SOMA maintains associations between user GUIDs and their homes in a directory service. In fact, on top of the basic naming mechanisms, SOMA provides a naming service that integrates a discovery protocol for resource access within the locality and an LDAP-based directory service for the registration of entities, such as profiles, in need of global visibility [1, 25].

MAM^2 operates on top of the SOMA-based general-purpose mobile computing middleware to provide an accounting management infrastructure to account/control the consumption of distributed resources. The infrastructure consists of two parts: a Base Accounting Layer, and an Upper Accounting Layer. The first one furnishes the mechanisms for resource metering, accounting data storage, and event notification. This layer is the basis of all accounting functions and it is necessary in every managed network host over the fixed network infrastructure. The second one adopts MAs to enforce account/control policies, by exploiting migration to maintain accounting entities co-located with accounted mobile users.

5.1. Base Accounting Layer

The base accounting layer includes three services: the Monitoring Service, the Storage Service and the Event Service. The Monitoring Service provides information about resource consumption both at the system level and at the application one. At the system level, it gets information on the processes acting on local resources and on their usage of the network. For any process, it reports the process identifier and name, CPU usage (time and percentage, both of the process and of its composing threads) and allocated memory (physical and virtual). The network information includes the total number of sent/received UDP packets, of sent/received TCP segments, of TCP connections, and of TCP/UDP packets received with errors. At the application level, this layer can collect information about all service components accessed from within the Java execution environment. For any active Java thread on the local Java Virtual Machine (JVM), it provides lifetime, number of loaded classes and used monitors, number and size of allocated objects, number of TCP/UDP and file read/write operations. The monitoring data are collected by exploiting Java extensions such as the JVM Profiler Interface and the Java Native Interface, as described in detail in [10].

The Storage Service records in a non-volatile memory the monitoring information about the resources that users have engaged locally to specific hosts, either when users execute directly at that host or when the specified host has been in charge of supporting network connectivity for users portable devices. On the base of this local information, monitoring data spread over managed nodes to compose a distributed accounting perspective, where MAs can collect global data whenever a global vision is required. The state of resource engagement is registered also with the aim of tracing a workload history for capacity planning. The Storage Service exploits the same persistency mechanisms used in SOMA to suspend the execution of agents and to store serialized MA code/state in persistent storage by exploiting standard Java solutions for database interfacing, e.g., Java Data Base Connectivity.

The Event Service can notify the interested local components when a registered event occurs. MAM^2 recognizes the events of user login/logout and the user change of domain of attachment. The Accounting Manager handles the whole interaction: it accepts registrations of interest for events and is in charge of event notification. We are currently using a proprietary and limited event support

with no possibility to define event aggregations and compositions, but we work to integrate our support with an external event-based distributed infrastructure, called JEDI, to improve the flexibility of event dispatching [27].

5.2. Upper Accounting Layer

The Upper Accounting Layer exploits the base one and determines how, where and when to perform accounting management. It consists of two types of entities: Accounting Managers (AMs) and Accounting Agents (AAs).

Any SOMA domain hosts one AM in charge of coordinating accounting activities in the locality depending on the enforced policies for service provisioning, billing and pricing. Policies are negotiated between users and providers, and are maintained at the same directory that stores user profiles. User profiles include personal data about identity, service preferences, e.g., required service quality and tariff plan. Service providers can deliver profiles and policies to MAM^2 by exploiting a registration/updating interface. Any default place maintains a directory server (Accounting Profile Directory) where profiles and policies are partially replicated and partitioned.

AAs are the entities delegated of the local enforcement of accounting management policies. Any AA relates to one authenticated user and works as her care-of entity: it traces and steers user behavior during service provision, in the sense of gathering information about resource usage and of controlling the respect of assigned consumption quotas. AAs exploit the base accounting layer: they get metering data from the monitoring service, and register the processed accounting results at the storage service.

We describe some mobility-enabled accounting scenarios to give a better idea of the MAM^2 components interoperation and coordination. We adopt a homogeneous notation: D1 is domain 1, P1.3 is place number 3 of domain 1, AM1 is the AM hosted at D1, etc. When one user U, from one either fixed or mobile terminal, enters a new domain, this generates an event notified to the AM in the domain where the U terminal is currently attached: Figure 3 shows U entering D1 and producing the event (1). The event notification triggers a sequence of accounting actions at AM1. Firstly, AM1 ascertains whether U has already one currently open session within MAM^2, i.e., controls whether U has previously visited another domain (D3 in the figure). AM1 requests the local directory service to retrieve the U home (2), if not already known, and queries the home to ask for the last domain visited. This action has also the effect of updating the home information of the current U attachment point. Once the previous domain has been found, AM1 coordinates with AM3 (3), the accounting manager of the previous domain, to trigger the migration of the user AA from D3 to D1 (4a). In the worst case of network partitioning, when AM1 cannot reach the user U home, it can overcome the unreachability by sending a multicast search message to all known neighbor AMs to ultimately discover a previous working AA for the entering user U. The receiving AM recursively asks other AMs by exploring the global domain interconnection. In case AM1 receives no answers within a specified timeout, it can only instantiate a brand new AA (4b).

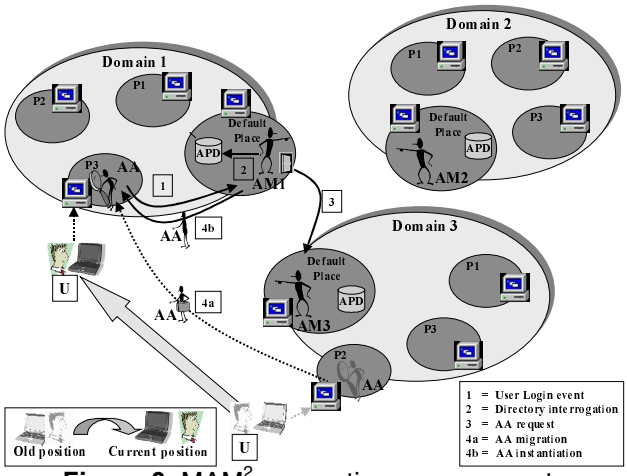

Figure 3. MAM^2 accounting management

In any case, when a new AA instance is needed, AM1 needs to retrieve the U profile in the directory to extract user-specific information about the desired accounting management. Then, AM ships the created AA to the place that needs to support U terminal connectivity (P1.3). When U moves to leave P1.3, her AA is freezed waiting for the event of U reconnection elsewhere in the system. MAM^2 provides also AA migration in advance before reconnection (*pre-fetching mode*) whenever U gives hints on moving intentions or exhibits a usual mobility pattern.

Let us point out that MAM^2 achieves the goal of continuing accounting activities even in case of different patterns of multiple connection failure. AAs can work autonomously and locally to current location of accounting users, requiring to communicate with the related AM only at the end of the U session. This event is automatically determined after U disconnection for an interval greater than a specified threshold. In fact, AMs need to exchange data with the U home only to answer location queries to determine if an AA for U is freezed in her previous domain of attachment. In any case, if a persistent network discontinuity between AM1 and AM4 makes impossible the correct information propagation, AM1 creates a new AA for the entering user after the expiration of a timeout. So, MAM^2 permits to overcome cases of faults, for sake of availability, by allowing several AA copies tied to the same user. When connectivity is reestablished, MAM^2 provides the join of multiple copies of AAs.

6. Conclusions and Future Work

The mobility-enabled Internet requires extending the fixed network infrastructure to permit service accessibility to mobile users/terminals when and where needed. Mobility changes the perspective also for network, systems and service management, and imposes to investigate more flexible solutions than the traditional C/S-based ones. Accounting in contexts of mobility can significantly benefit from support infrastructures based on mobile code. The MAM^2 system shows the feasibility and the effectiveness of an MA-based implementation of such an infrastructure.

The encouraging results obtained by the MAM^2 prototype are stimulating our further work. From the security point of view, MAM^2 already handles accounting data securely, by exploiting secure communications channels and encrypted storage. However, further work is necessary to guarantee non-repudiability of resource usage and to establish differentiated levels of trust between users and accounting components. In addition, we are implementing AMs capable of monitoring, controlling and registering resource consumption data of a user group, to enable accounting policies with the group granularity. Finally, we are interested in applying MAM^2 to SOMA agents themselves on the one hand to measure the usage of system resources due to accounting functions, on the other hand to leverage the commercial diffusion of services implemented in terms of cooperating MAs by preventing from MA-based denial-of-service attacks.

Acknowledgments

Work supported by the Italian Ministero della Ricerca Scientifica e Tecnologica in the framework of the Project "MUSIQUE: Infrastructure for QoS in Web Multimedia Services with Heterogeneous Access".

References

[1] P. Bellavista, A. Corradi, C. Stefanelli, “Mobile Agent Middleware to Support Mobile Computing”, *IEEE Computer*, Vol. 34, No. 3, Mar. 2001.

[2] J. D. Case, et al., *Simple Network Management Protocol (RFC 1157)*, DDN Network Information Center, SRI International, May 1990.

[3] G. Dickson, A. Loyd, *Open Systems Interconnection*, Trevor Housley (Ed.), Prentice-Hall, 1992.

[4] G. Goldszmidt, Y. Yemini, “Distributed Management by Delegation”, *15th Int. Conf. Distributed Computing Systems (ICDCS'95)*, June 1995.

[5] Hiroshi Yasuda (ed.), *2nd Int. Working Conf. Active Networks (IWAN'00)*, Lecture Notes on Computer Science, Japan, Oct. 2000.

[6] P. Morreale, K. Sohraby, L. Bo, L. Yi-Bing, "Active, Programmable, and Mobile Code Networking", *IEEE Communications*, Vol. 38, No. 3, Mar. 2000.

[7] B. Pagurek, Y. Wang, T. White, “Integration of Mobile Agents with SNMP: Why and How”, *IEEE/IFIP Network Operations and Management Symposium (NOMS'00)*, USA, Apr. 2000.

[8] P. Bellavista, A. Corradi, C. Stefanelli, “An Integrated Management Environment for Network Resources and Services”, *IEEE Journal on Selected Areas in Communications*, Vol. 18, No. 5, May 2000.

[9] P. Bellavista, A. Corradi, C. Stefanelli, “How to Monitor and Control Resource Usage in Mobile Agent Systems”, *3rd IEEE Symp. Distributed Objects and Applications (DOA'01)*, Sept. 2001.

[10] W. Fang, “Building an Accounting Infrastructure for the Internet”, *IEEE Globecom Internet Workshop*, Nov.1996.

[11] B. Aboba, J. Harko, D. Harrington, “Introduction to Accounting Management”, http://search.ietf.org/internet-drafts/draft-ietf-aaa-acct-06.txt, June 2000.

[12] International Standardization Organization, *Management Information Protocol Specification – Part 2: Common Management Information Protocol*, ISO IS9596, 1990.

[13] Tivoli Systems Inc. - *Tivoli*, http://www.tivoli.com

[14] AdventNet Inc. - *AdventNet SNMP API*, http://www.adventnet.com

[15] C. Rigney, *RADIUS Accounting*, IETF RFC 2139, Jan. 1997.

[16] P. R. Calhoun, P. Pan, H. Akhtar, *Diameter Framework Document*, http://search.ietf.org/internet-drafts/draft-ietf-aaa-diameter-framework-01.txt

[17] E. Kovacs, K. Rohrle, M. Reich, “Integrating Mobile Agents into the Mobile Middleware”, *Proc. Mobile Agents Int. Workshop (MA'98)*, Germany, 1998.

[18] S. Lipperts, A. Park, “An Agent-based Middleware: a Solution for Terminal and User Mobility,” *Computer Networks*, Vol. 31, Sep. 1999.

[19] F. Baschieri, P. Bellavista, A. Corradi, “Mobile Agent for QoS Tailoring, Control and Adaptation over the Internet: the ubiQoS Video on Demand Service”, *2nd IEEE Int. Symp. Applications and the Internet (SAINT'02)*, Jan. 2002.

[20] D. Lange, M. Oshima, *Programming and Deploying Mobile Agents with Java Aglets*, Addison-Wesley, 1998.

[21] N. Dulay, R. Montanari, C. Stefanelli, "Flexible Security Policies for Mobile Agent Systems", *Microprocessors and Microsystems*, Elsevier Science, Vol. 25, No. 2, Apr. 2001.

[22] IKV++, Grasshopper 2, http://www.grasshopper.de/

[23] P. Bellavista, A. Corradi, C. Stefanelli, "Middleware Services for Interoperability in Open Mobile Agent Systems", *Microprocessors and Microsystems*, Elsevier Science, Vol. 25, No. 2, May 2001.

[24] M. Baldi, G. P. Picco, "Evaluating the Tradeoffs of Mobile Code Design Paradigms in Network Management Applications", *IEEE Int. Conf. on Software Engineering (ICSE'98)*, Apr. 1998.

[25] A. Fuggetta, G.P. Picco, G. Vigna, "Understanding Code Mobility", *IEEE Transactions on Software Engineering*, Vol. 24, No. 5, May 1998.

[26] P. Bellavista, A. Corradi, C. Stefanelli, "A Mobile Agent Infrastructure for Terminal, User and Resource Mobility", *IEEE/IFIP Network Operations Management Symp. (NOMS 2000),* USA, Apr. 2000.

[27] G. Cugola, E. Di Nitto, A. Fuggetta, “The JEDI Event-based Infrastructure and its Application to the Development of the OPSS WFMS”, *IEEE Transactions on Software Engineering*, Vol. 27, No. 9, Sep. 2001.