Journal of Network and Systems Management, Vol. 11, No. 2, June 2003 (© 2003)

Mobile Agent-Based Performance Management for the Virtual Home Environment

C. Bohoris,[1,2] G. Pavlou,[1] and A. Liotta[1]

Virtual Home Environment (VHE) encompasses the deployment and management of adaptable services that retain any personalized service aspects, irrespective of terminal, network' and geographic location. We assert that the dynamic nature of the VHE requires management capabilities that can be suitably provided through the use of mobile agent technology. We examine four different engineering solutions for the realization of a VHE performance management component that allows service adaptation in relation to the available network Quality-of-Service (QoS). The mobile agent approach is compared with competing technologies in order to identify the benefits of this novel application of mobile agents, discuss its drawbacks' and finally focus on the lessons learned from our prototype system. Although mobile agents are typically associated with increased performance costs, it is through agent migration that we were able to address the VHE requirements of universality, dynamic programmability, and network technology independence.

KEY WORDS: Mobile agents; distributed objects; programmability; performance management; virtual home environment.

1. INTRODUCTION

The preparation of the way for the so-called 3rd generation mobile network infrastructure is complemented by an increased interest of the industry in new advanced services that will "intelligently" cooperate with their environment in order to support features not possible so far. Large industrial consortia, most importantly, the 3rd Generation Partnership Project (3GPP) [1] (with its Open Service Access-OSA-APIs) as well as the Parlay group [2], are working in a merging direction towards the specification of open interfaces for services that can operate across multiple networking platform environments. Within this vision of

[1]Center for Communication Systems Research, School of Electronic Engineering and Information Technology, University of Surrey, Guildford, Surrey GU2 7XH, United Kingdom. E-mail: {*c. bohoris, g.pavlou, a.liotta*}*@eim.surrey.ac.uk*
[2]To whom correspondence should be addressed.

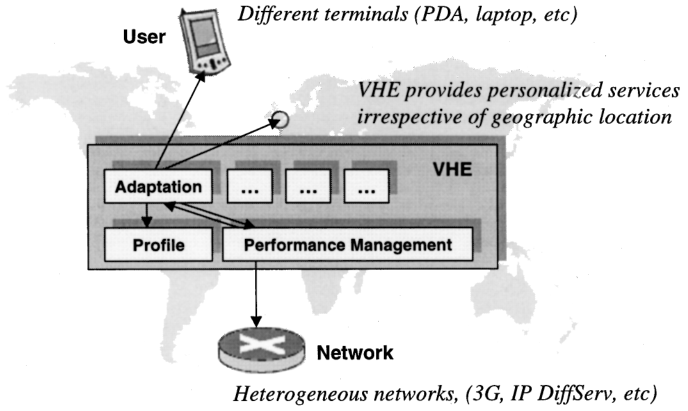
Different terminals (PDA, laptop, etc)
User
VHE provides personalized services irrespective of geographic location
VHE
Adaptation
...
...
...
Profile
Performance Management
Network
Heterogeneous networks, (3G, IP DiffServ, etc)

Fig. 1. The VHE environment.

"intelligent", distributed software systems for telecommunications applications, an important emerging concept is that of the Virtual Home Environment (VHE) [3]. A VHE manages a number of services with the aim of consistently providing to the user personalized service aspects, irrespective of the terminal, network and geographic location involved (see Fig. 1).

To fulfill this goal a VHE depends on a sophisticated adaptation process. Initially, when the user requests a service, a VHE adaptation component gathers information about the environment involved in service provisioning. This information reflects on a number of parameters describing the network (for adaptation to network performance conditions and Quality-of-Service (QoS) offered), terminal (for adaptation to terminal capabilities) and geographic location (for location-based adaptation of service content). In order to trigger an adaptation action the VHE adaptation component performs a careful analysis of this information in relation to the user's preferences kept in a VHE user profile. Following this initial adaptation process and during service usage, the VHE adaptation component depends on real time notifications of any changes in the environment.

Most importantly, a network performance management component crucially supports VHE adaptation by keeping it informed of changing network conditions that are analyzed and may trigger an adaptation action. This real time adaptation process is particularly important for the VHE as it ensures service continuity and correct service operation. Performance management involves two different aspects namely QoS management and performance monitoring. The QoS management part configures connectivity so that the service traffic is delivered with QoS assurances. The performance monitoring part monitors the network performance conditions affecting the connection and informs the VHE of network QoS changes that may require a service adaptation action. This is particularly important when only weak

QoS guarantees can be delivered, e.g., qualitative as opposed to quantitative ones, or when only best-effort connectivity is supported.

The dynamic nature of the VHE gives rise to a number of crucial requirements on a VHE performance management component. An important requirement stems from the fact that a user may unexpectedly request a VHE service from any geographic location. In a decentralized manner, VHE performance management functionality should operate at any “nonprovisioned” location to dynamically configure and manage the user’s connectivity path. Additionally, a user may rely on heterogeneous network infrastructures (e.g., Universal Mobile Telecommunications System (UMTS) or Internet Protocol (IP) based networks). The involved network is dynamically determined by a VHE performance management component that decides on the appropriate management logic required (e.g., on network specific performance parameters, measurement methodologies). Finally, the user may select from a number of diverse services with different management requirements. VHE performance management should be able to configure its operation based on the requirements of the specific service (e.g., a network delay sensitive video transmission service or a packet loss sensitive software download service). A natural way for a VHE performance management component to address these requirements is by dynamically deploying tailor-made entities in the appropriate network nodes. Among the various alternative management approaches available today, mobile agent technology has the potential to provide a preferable engineering approach to the realization of a dynamic performance management component for the VHE.

Our VHE work was performed within the context of the IST-VESPER project [4] aiming to specify and develop a complete VHE architecture and examine the potential benefits of mobile agent technology in the VHE environment. Although studies of the applicability of mobile agents were made for various components of the VESPER VHE, the research presented in this article focuses on the network management aspects and in particular on the performance monitoring aspects of a VHE performance management component. In this article we first present various modern alternative technologies for network management. This is followed by a detailed description of the VHE requirements imposed on network performance management providing the basis for our arguments in favor of a mobile agent-based approach. Subsequently, we describe a mobile agent-based VHE performance monitoring system in terms of its functionality, the approach to programmability as well as its external interfaces specified in a Parlay compliant manner and extending Parlay with required interactions. Following this, we present a thorough evaluation of our proposed mobile agent-based performance monitoring system (based on IKV++’s Grasshopper [5] mobile agent platform), compared with three other systems of similar functionality based on the Common Object Request Broker Architecture (CORBA), Sun Microsystems’s Java Remote Method Invocation (Java-RMI) and Jasmin Script Management Information Base (Script-MIB), representing distributed objects and mobile code approaches to management. With

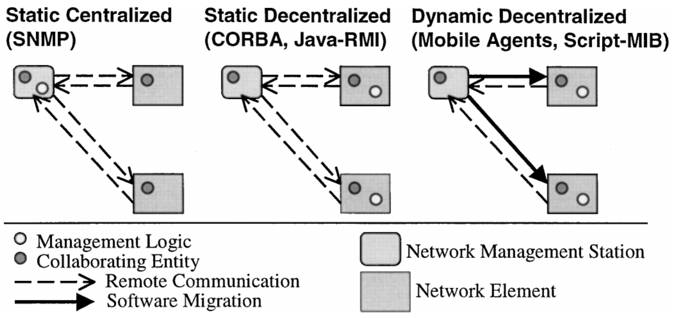
Static Centralized (SNMP)
Static Decentralized (CORBA, Java-RMI)
Dynamic Decentralized (Mobile Agents, Script-MIB)
Management Logic
Collaborating Entity
Remote Communication
Software Migration
Network Management Station
Network Element

Fig. 2. Network management approaches.

respect to Jasmin Script-MIB, we should mention that this article presents one of the first experimental evaluations of the technology complementing and extending initial results obtained by its development team [6]. Finally, in Section 7 we discuss the lessons learnt, the conclusions drawn and the arising issues influencing our future research directions.

2. TECHNOLOGIES FOR NETWORK MANAGEMENT

Network management approaches have gone through a number of phases of evolution (see Fig. 2 and Table I). The initial protocol-based approaches proposed

Table I. A Comparison of the Transfer Capabilities of Alternative Technologies

	Mobile agents	Java-RMI	CORBA	Script MIB
Transfer of logic to a remote node	Agent mobility involving the creation of an agent, migration to a generic execution environment and resumption of its autonomous execution.	Object mobility involving the creation of an object, transfer to a server of the system, use of the object and its data in the specific server context.	Not supported.	Code mobility involving code transfer to an execution environment, object creation and stand alone execution.
Transfer of data to a remote node	Between any agent (using dynamic communication mechanisms based on sockets)	From a client to a server (using the Java Remote Method Protocol (JRMP))	From a client to a server (using the Internet Inter-ORB Protocol (IIOP))	Client queries the server (using the User Datagram Protocol (UDP))

in the early 90's involve management systems that are "centralized" and "static" as exemplified by IETF's Simple Network Management Protocol (SNMP) [7]. The approach is centralized as it relies on a limited set of capabilities at network nodes while management processing has to be performed at the network management station. Any capabilities at network nodes are fixed, embedded by the manufacturer at the network element construction time. While a protocol-based approach is specific to a management framework, a generic approach based on the client/server model can be supported through the use of a distributed object framework as proposed in the mid-90s.

Management based on distributed object frameworks allowed "decentralized" and "static" systems, as exemplified by CORBA [8] and Java-RMI [9]. Decentralization is achieved by placing required management logic in network nodes and by creating instances of management objects specific to interested clients. Although distributed object frameworks such as CORBA have succeeded in allowing the development of decentralized systems, they still suffer from lack of support for programmability, given that the management logic located in network nodes is static and cannot be easily altered. The issue of programmability of managed nodes is particularly important for network management systems. The lesson learnt from the deployment of management systems based on early standards was that they were highly complex and suffered from long standardization cycles [10]. The latter means that network administrators had to wait several years before a standardization cycle was completed and the required management functionality was embedded in network nodes. To address this problem, research efforts have focused on the exploitation of software mobility.

The first effort on the use of software mobility for decentralized and programmable network management operations was made in the early 90's with Management by Delegation (MbD) [11]. The approach is based on mobile code that is sent to a remote node for execution and it was later adopted by the IETF as the basis of the Script-MIB proposal [12] and its first implementation in the form of the Jasmin platform [13].

The Script-MIB uses the MbD paradigm to define an SNMP-compliant Management Information Base for the delegation of management functions in the SNMP framework. MbD can be considered as a successor of the Remote Evaluation (REV) paradigm for code mobility described by Stamos and Gifford [14]. In this paradigm, code containing the required logic is pushed along with initial parameters to a remote node, where object instantiation and stand-alone execution takes place.

The REV paradigm evolved further into the 'Constrained' mobility model involving mobile software agents [15]. The model was termed constrained mobility since the software agent, upon its creation at a client site, performs a single migration to a remote server where its execution is confined. An important aspect of constrained mobility is that a mobile entity is not restrained to be a remote service, as in the case of REV, but instead acts as an autonomous software agent (e.g., choosing its migration node, intelligently collaborating with other agents to achieve its task, etc.). Earlier studies on constrained mobility of agents (e.g., assessments see [15, 16]) have shown that the model fits well typical network management requirements in systems that involve long-term management tasks for which programmability of the distributed management logic is required.

Naturally the more advanced capabilities of mobile agent-based systems are also linked with an increase in performance overheads compared to systems based on distributed objects (see [15]). Such performance overheads affecting the managed network are typically associated with mobile agent migration. In an effort to keep these overheads to a minimum many mobile agent-based systems today exhibit single-hop, constrained agent migration only or multi-hop migration of small agents only. Another area of increased performance overheads of mobile agents in comparison to static distributed objects is typically associated with their remote communication. This is commonly attributed to the much more dynamic communication mechanisms required (e.g., using capabilities of component frameworks such as reflection) in order to accommodate the remote communication of migrating agents.

3. REQUIREMENTS ON VHE PERFORMANCE MANAGEMENT

The VHE typically exhibits the following important characteristics:

1. It is a large-scale system, as it serves a large number of users and needs to control and manage several networks for this.
2. It is a heavily dynamic system, with users, service types, locations, management and control requirements, access terminals and networks changing as a typical part of its operation.
3. It is able to operate over a heterogeneous network environment.

These three characteristics impose three important requirements on the VHE management aspects and particularly on performance management:

1. *Universality*: A user can request to use a service in a VHE manner from anywhere at anytime. In order to accommodate this requirement, suitable management functionality should be available for execution at ‘any’ network element that might be involved in a user’s connectivity path.
2. *Dynamic, programmable management functionality*: The requirements on management functionality vary depending on the service the user chooses. For example, a video conferencing service has different management requirements compared to an on-line calendar service (this is a shared agenda used concurrently by many users). In addition, a key target of a competitive VHE provider will be the rapid introduction of new VHE services. In this

respect, management flexibility should be provided to allow for efficient addition and customization of the available functionality in order to accommodate different service needs.

3. *Network technology transparency*: The user may be connected using a heterogeneous network environment. Management logic should translate a user view of requirements from the network into an abstract network view of these requirements and then map them into the specifics of different underlying network technologies.

As we can see from these requirements, the VHE environment introduces a number of challenges for VHE performance management. The management logic of the large-scale VHE should be dynamically deployed in a scalable manner and it should also support different network technologies and service needs. Such requirements should be crucially considered during the selection of a preferable approach of delivering VHE performance management capabilities. In the following section we move on to examine how several management approaches available today cope with the VHE requirements.

4. VHE BENEFITS FROM AGENT MOBILITY

SNMP is currently the most popular management technology and has proven to be very effective for basic management tasks (e.g., monitoring of local area networks). Despite this, scalability problems due to the centralized nature of SNMP make it inappropriate for the management of a large-scale VHE. In addition, any upgrade/customization of functionality in network elements requires introduction (re-) compilation and activation of SNMP code and this clearly cannot satisfy the requirement of a VHE for dynamic, programmable management.

In comparison to SNMP, distributed object systems provide a more scalable approach to managing networks, capable of fulfilling the VHE requirement for scalable operation. However, in a similar fashion to SNMP, distributed object frameworks still rely on fixed management functionality that cannot be customized or upgraded without system re-installation, a fact that fails the VHE requirement for programmability. The lack of support for programmability in distributed object approaches also clashes with the ‘Universality’ requirement of the VHE. For example, in order to accommodate a user that may unexpectedly request a service from/to a “non-provisioned” location, we would need management functionality satisfying the VHE requirements in the edge network elements.

By exploiting software mobility, the Script-MIB and mobile agent approaches both fulfill the VHE requirements for programmability and ‘Universality’ by dynamically deploying software entities where and when needed. However, the Script-MIB approach is specific to the SNMP management framework and thus does not fulfill the VHE requirement for network technology transparency. On the other hand mobile agents provide a generic approach. Agents can be configured to autonomously acquire the appropriate context (e.g., support for measurement of performance parameters specific to a particular network infrastructure) for cooperation with the underlying network technology identified and thus fulfilling the VHE requirement for network technology transparency.

Based on this discussion, we see that the mobile agent approach is the only one that can fulfill all three stated VHE performance management requirements detailed in Section 3. On this basis, a mobile agent-based approach for the realization of a VHE performance monitoring component is presented in the following section.

5. PROPOSED APPROACH

5.1. System Design

The performance monitoring system monitors the edge nodes involved in a user's connectivity path, following an approach that separates the management logic from network technology specific parts that are loaded dynamically, as and when required. The VHE functionality for performance monitoring we developed follows a generic design approach to constrained mobility involving three different agent roles as presented below (see Fig. 3):

1. *Master Agent*: Stationary agent responsible for the interactions between the user and the system. The Master interacts in both directions with the user (i.e., the VHE adaptation component), initializing and controlling the performance monitoring process and sending performance monitoring notifications and reports that may trigger an adaptation action.
2. *Worker Agent*: A mobile agent autonomously equipped with the required management logic and supporting context appropriate to the underlying network technology that subsequently migrates to the targeted node for execution. Our Worker agent for performance monitoring creates a number of monitors of required performance parameters and migrates with them to the targeted network element to perform its task.
3. *Target Agent*: Stationary agent at the targeted network element allowing the monitors to access a number of required resources. The Target of our system allows passive and active performance measurements. For passive measurements the Target 'wraps' the underlying network technology allowing access to 'raw' resources of the network element (for measurements of used bandwidth, loss, etc). For active measurements the Target provides an "echo" facility that remotely returns a test stream to the sender upon which a measurement can be taken (for measurements of delay, jitter, etc).

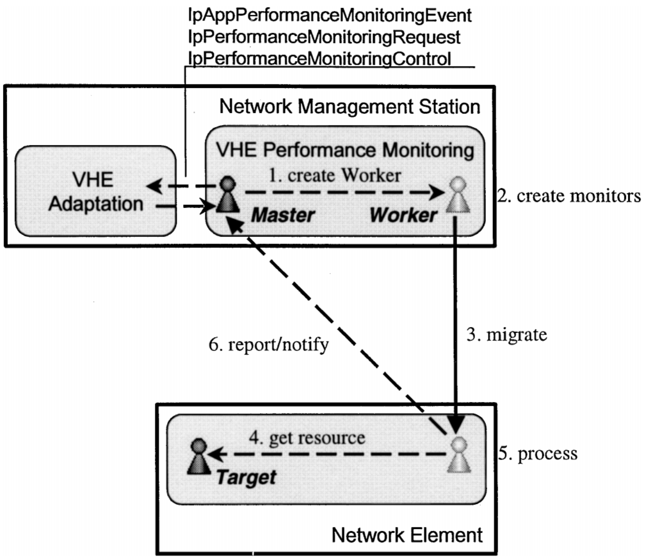
IpAppPerformanceMonitoringEvent
IpPerformanceMonitoringRequest
IpPerformanceMonitoringControl
Network Management Station
VHE Adaptation
VHE Performance Monitoring
1. create Worker
Master
Worker
2. create monitors
3. migrate
6. report/notify
4. get resource
Target
5. process
Network Element

Fig. 3. An agent-based VHE performance monitoring system following the model of constrained mobility.

The user of this system, in fact the VHE adaptation component, controls the performance monitoring operation and receives network performance events in a generic manner. These external interfaces and data types involved in the communication with the user are compliant with the design guidelines of the Parlay group (see Fig. 4). This was done as a complement to current Parlay work (Connectivity Manager APIs version 2.1, [2]), which although still in progress, enjoys a strong industry support and has been selected by 3GPP as the basis for their OSA APIs. In order to gain wider acceptance it is important for any VHE component interacting with different networks or services to provide a generic set of interfaces that are in line with current standardization efforts.

Using the API shown in Fig. 4, a VHE adaptation component request for performance monitoring of a node is initially passed to the Master agent, which in turn creates a suitable mobile Worker agent (see Fig. 3). The Worker agent, based on the given performance parameters, loads the associated Java class files at runtime. Upon completion of this process, the Worker migrates to the targeted node where it contacts the Target agent that pre-exists in the node and gives access to 'raw' performance information. The Worker initiates the operation of a number of monitors each responsible for the monitoring of a single performance parameter and running on its own execution thread. Each monitor initiates its task with periodic requests for 'raw' performance information (counter type values) provided

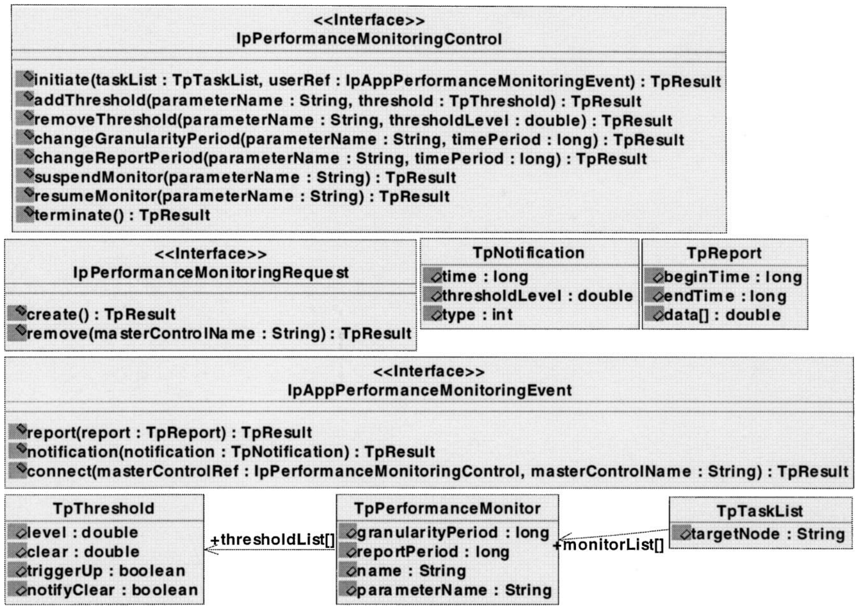
<<Interface>>
IpPerformanceMonitoringControl
initiate(taskList : TpTaskList, userRef : IpAppPerformanceMonitoringEvent) : TpResult
addThreshold(parameterName : String, threshold : TpThreshold) : TpResult
removeThreshold(parameterName : String, thresholdLevel : double) : TpResult
changeGranularityPeriod(parameterName : String, timePeriod : long) : TpResult
changeReportPeriod(parameterName : String, timePeriod : long) : TpResult
suspendMonitor(parameterName : String) : TpResult
resumeMonitor(parameterName : String) : TpResult
terminate() : TpResult
<<Interface>>
IpPerformanceMonitoringRequest
create() : TpResult
remove(masterControlName : String) : TpResult
TpNotification
time : long
thresholdLevel : double
type : int
TpReport
beginTime : long
endTime : long
data[] : double
<<Interface>>
IpAppPerformanceMonitoringEvent
report(report : TpReport) : TpResult
notification(notification : TpNotification) : TpResult
connect(masterControlRef : IpPerformanceMonitoringControl, masterControlName : String) : TpResult
TpThreshold
level : double
clear : double
triggerUp : boolean
notifyClear : boolean
+thresholdList[]
TpPerformanceMonitor
granularityPeriod : long
reportPeriod : long
name : String
parameterName : String
+monitorList[]
TpTaskList
targetNode : String

Fig. 4. Parlay compliant interfaces for VHE performance monitoring.

by the Target. Based on this information a monitor performs metric monitoring for a number of performance parameters (e.g., Used Bandwidth, Loss, Delay, etc), checks the thresholds set and gathers the information produced in order to generate reports. When a monitor generates a performance event (report or notification) this is passed from the Worker remotely to the Master agent and finally to the VHE adaptation component. The remote reports of the results gathered are generated on a scheduled basis (e.g., every 15 min.) while notifications are sent on the fly every time a performance threshold is triggered.

The functionality included in the monitors carried by the Worker agent is based on the Metric Monitoring and Summarization Open Systems Interconnection (OSI) Systems Management Functions (SMFs) [17]. While such performance monitoring functionality is fixed in OSI Systems Management (OSI-SM), the logic included inside the Worker may be customized by the VHE; e.g., to provide a different model for triggering notifications, based on semantic knowledge of the monitored resources. The important customization aspects are described next.

5.2. Customization of the Management Logic

Network elements tend to provide a set of standardized as well as proprietary objects that can be accessed remotely to support management functionality. These

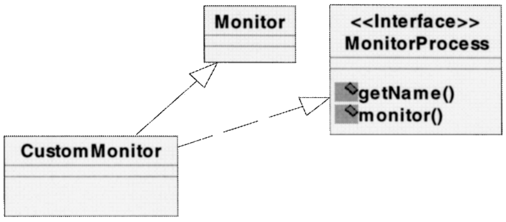

Fig. 5. Allowing programmable performance monitoring logic.

are fixed and cannot be altered or extended. In that sense a network element can be characterized as a black box with pre-programmed management capabilities. With the evolution of the telecommunications industry, we see today support for distributed object architectures at the network element level (e.g., [18]) and in the future, eventually, support for mobile agent technologies will also be included. The introduction of dynamic applications such as the VHE may help to speed up changes in this direction. The key benefit is that mobile agents implementing customized functionality could migrate and execute there, augmenting dynamically the elements' capabilities. We examine here how the functionality of a mobile agent can be easily customized in our system.

Initially, the VHE adaptation component analyses the performance management requirements of a specific VHE service, the underlying network technology involved as well as the available network resources and decides on any required customizations of the performance management logic. This customized functionality can be provided in an object that inherits the functionality of the standard monitor. In Fig. 5, we see an example CustomMonitor class that can extend the standard monitoring functionality by inheriting from the standard Monitor class provided by the system. In addition by implementing the MonitorProcess interface as shown in Fig. 5 the CustomMonitor can also override the monitoring behavior of the standard monitor in order to introduce its own customized approach.

The standard monitor provided by the system already supports some basic form of customization as any performance parameter involved is loaded dynamically at runtime. This allows the monitor to keep a generic view of its task and use a suitable performance parameter implementing a customized measurement approach reflecting on a specific network technology or different measurement methodologies.

6. EVALUATION AND ASSESSMENT

6.1. Environment and Methodology

In our evaluation we are interested to highlight the performance overheads in the managed network associated with the VHE performance monitoring task based on several technologies available to date. In addition, we consider a number of software metrics assessing each of the approaches in terms of quality and efficiency. Within this scope we have developed four performance monitoring systems with similar functionality based on Mobile Agents, CORBA, Java-RMI and the Jasmin Script-MIB. For our development we used the Java programming language, with all system classes built and run using Sun Microsystems's JDK version 1.3.1. An exception to this was Jasmin Script-MIB (version 1.0) that requires also the much older JDK 1.1.6 in order to run its scripts. Regarding CORBA and Java-RMI the supporting facilities and APIs included in Sun's JDK 1.3.1 were used. Software mobile agent capabilities were provided by IKV's Grasshopper mobile agent platform version 2.2.3. Grasshopper was chosen as a typical all-round platform that combines benefits such as compliance with agent standards, simplicity in usage and agent programming as well as good support and documentation.

For our VHE performance monitoring systems the important areas of performance overheads involve the tasks of software migration as well as remote communication between the various entities. In this context and for the four systems developed, measurements were taken for the following typical system operations:

1. The remote transmissions of a scheduled performance report and a real-time notification (see Fig. 3, step 6) from the monitoring entity at the managed node to the Master entity at the network management station. A performance report involves the TpReport data type (see Fig. 4) containing a list of all the performance monitoring information produced. For our measurements we considered reports containing a list of 25, 50, 75 and 100 'double' real numbers, reflecting on information gathered during the metric monitoring process. Similarly, notifications involve the transmission of an object based on the TpNotification (see Fig. 4) data type informing the Master that a performance threshold was crossed.
2. The software migration occurring in the mobile agent and Script-MIB-based systems. Our measurements involve the migration of a Worker Agent (see Fig. 3, step 3) as well as the equivalent Script-MIB code carrying the management logic to a managed node.

For these operations we have taken the following measurements:

1. Traffic measurements: Taken using the tcpdump utility with the sizes reported reflecting on the total payload at TCP level. As an exception for the Jasmin Script-MIB system the sizes reported reflect on the total payload at UDP level.
2. Response times measurements: Taken using the System.currentTime Millis( ) method included in the API of Sun's JDK. The measured values of response times reported represent the "steady-state" running costs, excluding the initial setup costs.

All measurements were taken using a testbed of Linux workstations with homogeneous features (Redhat 7.1, Pentium Celeron 466MHz and 64MB of RAM), connected to a 100Mbps Ethernet network. Each node is configured as a software router with Differentiated Services (DiffServ) support, as provided by the Linux kernel version 2.4.2.

6.2. Experimental Results

For remote reporting and notification operations sent from the Worker agent in the network element to the Master agent in the network management station we have taken the following traffic (Fig. 6) and response times (Fig. 7) measurements for the four different approaches.

From the two plots of Fig. 6 we can observe that for a small amount of data (a notification or a report of 25 elements) the Grasshopper system performs competitively incurring around 15% more traffic than CORBA. Jasmin performs better than these two while the best performer is Java-RMI. The slope of the CORBA line in the reports graph indicates that the approach scales better than the other approaches and can eventually outperform them for reports containing more than 100 elements.

From Fig. 7 we can see that the distributed object approaches offer comparable performance. The Jasmin system required almost twice the time for the operations while the Grasshopper system was about 5 times slower. These results confirm the initial concerns discussed in Section 4 on the additional mobile agent communication overheads commonly attributed to the dynamic communication mechanisms required for the remote collaboration between migrating agents.

Regarding software migration, the mobile agent Worker needed 1,418 ms to migrate while it incurred 2,932 bytes of traffic. The Jasmin Worker performed

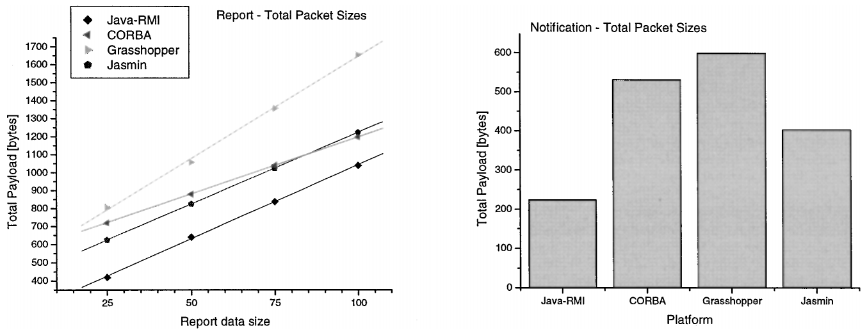

Fig. 6. Report and notification traffic.

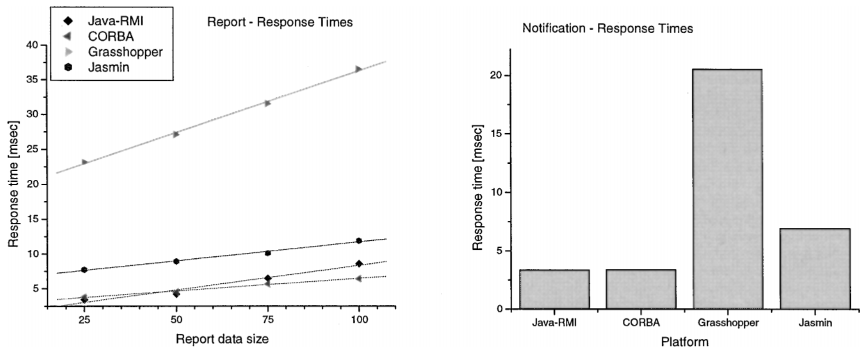

Fig. 7. Report and notification response times.

slightly better requiring 1,216 ms and incurring 2,496 bytes of traffic. For comparison, typically the creation of a distributed object through a factory requires less than 15 ms to complete and incurs around 500 bytes of traffic [19].

In addition to the performance characteristics of each system a number of software metrics were considered as an indication of the complexity of working with each approach and appear in Table II. In this table we report on "Generated" code representing platform-specific supporting code (e.g., object skeletons and stubs) that is bundled along with the performance monitoring code written by a programmer. From this table we see that CORBA and Java-RMI both rely on a large amount of platform-specific code contributing significantly to the total compiled code of the system. Both platforms offer a stable and well-documented environment for developing and running distributed applications. The Jasmin platform was found to be unstable as well as poorly and/or incompletely documented making its use for critical systems still difficult or even inappropriate. Its mobile code approach is very limited to a simple download and start mechanism used to extend in this way the SNMP functionality in the network element. The agent features available in the Grasshopper platform provided a basis for the development of a more capable and fine-tuned (i.e., based on autonomous agent behavior)

Table II. Software Metrics for Each System

	Grasshopper	Java-RMI	CORBA	Jasmin
Generated compiled code [Kbytes]	0	60	136	0
Written compiled code [Kbytes]	86	84	92	138
Total compiled code [Kbytes]	86	144	228	138
Generated source code [Lines]	0	2064	2460	0
Written source code [Lines]	2363	2540	2583	2972

performance monitoring system, requiring significantly less written source code compared to the Jasmin system. Grasshopper was found to be a stable platform, providing a simple and well-documented API, allowing the fast introduction and enhancement of applications and associated management solutions.

7. CONCLUSIONS

In this article we have presented first the challenges that the VHE environment brings to performance management. Our initial analysis considered the VHE requirements on performance management along with the capabilities of several management technologies available today in order to identify the most suitable candidates. Through this work we have identified the limitations of modern mobile code and distributed object solutions making them unsuitable for addressing requirements of the VHE adaptation system. In particular distributed object technologies, despite their good performance are let down by the lack of support for easy programmability. As an alternative, it was through mobile agent technology that we managed to fulfill the critical VHE requirements for network element programmability, 'universality' and network technology independence, as discussed in Section 3. Our experiments confirmed that the more advanced mobile agent capabilities are associated with an increase in performance overheads, which may not be acceptable in all cases. Here we should note that the Jasmin Script-MIB solution offers significantly better performance compared to mobile agents while also allowing programmability. Unfortunately, Script-MIB is specific to the SNMP management framework and thus does not fulfill the VHE requirement for network technology transparency. In order to give the correct perspective it should be clarified that the VHE concept is only envisioned to impact the telecommunications industry in the future where mobile agent performance issues may not be anymore an issue. As an analogy, by today's standards we claim that performance of distributed object technologies is excellent but we should recall that in the early 90s the same overheads where considered by many as a major obstacle to the future of distributed object frameworks. For now and as an alternative to this situation, we could say that distributed objects and mobile agents should ideally coexist in management systems that combine the best of both approaches (i.e., distributed object performance and mobile agent programmability). Real synergy could be achieved if stationary agents could be provided using static objects, with method invocations being possible between mobile and static objects in both directions. Although some previous work considered the architectural aspects involved in the integration of mobile agents with static CORBA objects (e.g., [20, 21]), important issues of system design are yet to be thoroughly investigated. This is the direction of our current research work and an issue that standardization bodies such as the OMG should attempt to address.

REFERENCES

1. 3rd Generation partnership project (3GPP), *http://www.3gpp.org*.
2. Parlay group, *http://www.parlay.org*.
3. 3GPP Technical Specification 22.121 v4.0.0, The Virtual Home Environment (Release 4), October 2000.
4. IST VESPER project (IST-1999-10825), *http://vesper.intranet.gr*
5. IKV^{++} Grasshopper Mobile Agent platform, *http://www.grasshopper.de*.
6. F. Straub, Script MIB Performance Analysis, *Simple Times*, 7(2), November 1999.
7. J. Case, M. Fedor, M. Schoffstall and J. Davin, A Simple Network Management Protocol (SNMP), IETF RFC 1157, May 1990.
8. Object Management Group, The Common Object Request Broker: Architecture and Specification (CORBA), Version 2.0, 1995.
9. Sun Microsystems, Java Remote Method Invocation (Java-RMI), *http://java.sun.com/products/jdk/rmi/*
10. G. Pavlou, G. Mykoniatis and J. Sanchez, Distributed intelligent monitoring and reporting facilities, IOP Publishing, *IEEE Distributed Systems Engineering Journal (DSEJ)*, Special Issue on Management, Vol. 3, No. 2, pp. 124–135, 1996.
11. Y. Yemini, G. Goldszmidt, and S. Yemini, *Network Management by Delegation, Integrated Network Management II*, Krishnan and Zimmer (eds.), Elsevier, pp. 95–107, 1991.
12. D. Levi and J. Schoenwaelder, Definitions of Managed Objects for the Delegation of Management Scripts, IETF RFC2592, May 1999.
13. Jasmin Script-MIB, *http://www.ibr.cs.tu-bs.de/projects/jasmin/*
14. J. W. Stamos and D. K. Gifford, Implementing remote evaluation, *IEEE Transactions on Software Engineering*, Vol. 16, No. 7, pp. 710–722, July 1990.
15. C. Bohoris, G. Pavlou, and H. Cruickshank, Using mobile agents for network performance management, *Proceedings of the IFIP/IEEE Network Operations and Management Symposium 2000 (NOMS '00)*, Hawaii, USA, J. Hong, R. Weihmayer (eds.), pp. 637–652, April 2000.
16. P. Simões, J. Rodrigues, L. Silva, and F. Boavida, Distributed retrieval of management information: Is it about mobility, locality or distribution?, *Proceedings of the Network Operations and Management Symposium 2002*, Florence, Italy, April 2002.
17. ITU-T Recommendation X.739–Metric Objects and Attributes (1992); ITU-T Recommendation X.738–Summarization Function (1993).
18. Cisco Systems, Element Management Framework (EMF) CORBA Gateway, *http://www.cisco.com/warp/public/cc/pd/nemnsw/emf/prodlit/crba_ds.ht*
19. G. Pavlou, Telecommunications management network: A novel approach towards its architecture and realization through object-oriented software platforms, Ph.D. Thesis, University College London, March 1998.
20. F. Chatzipapadopoulos, M. Perdikeas, and I. Venieris, Mobile agent and CORBA technologies in the broadband intelligent network, *IEEE Communications Magazine*, Vol. 38, No. 6, pp. 116–124, June 2000.
21. P. Bellavista, A. Corradi, and C. Stefanelli, An open secure mobile agent framework for systems management, *Journal of Network and Systems Management*, Vol. 7, No. 3, pp. 323–339, September 1999.

Christos Bohoris is a researcher working towards a Ph.D. degree within the Networks Group at the Center for Communication Systems Research, University of Surrey, UK. He holds a B.Eng. in Electrical and Electronic Engineering from the University of Wales, Cardiff' and an M.Sc. in Telematics

(Communications and Computer Engineering) from the University of Surrey. He has worked as full-time researcher in the MIAMI, MANTRIP and VESPER European collaborative research projects, looking at the impact of mobile code at management systems for ATM, QoS-enabled IP and 3G advanced services. His research interests include enabling software technologies and novel architectures for network and service management.

George Pavlou is a Professor of Communication and Information Systems at the Center for Communication Systems Research, School of Electronics and Computing, University of Surrey, UK, leading the activities of the Networks Research Group. He holds a Diploma in Engineering from the National Technical University of Athens, Greece, and M.Sc. and Ph.D. degrees in Computer Science from University College London, UK. At Surrey he is responsible for a number of European and UK research projects and Industrial collaborations. His recent research interests include network planning and dimensioning, traffic engineering and management, mobile ad hoc networks, novel architectures and technologies for network and service management, multimedia service control, programmable/ active networks and communications middleware.

Antonio Liotta is a Lecturer at the Center for Communication Systems Research, School of Electronics and Computing, University of Surrey, UK. He obtained a ‘Laurea’ degree in Electronic Engineering from the University of Pavia, Italy, an M.Sc. in Information Technology from Polytechnico di Milano’ and a Ph.D. in Computer Science from University College London, UK. Since 2000, he has worked at the University of Surrey as a researcher and academic, and has been involved in several collaborative projects in the areas of network and service management. Recent research interests include the use of mobile agents for distributed network monitoring, service management for the Virtual Home Environment and middleware for advanced services in 3G mobile systems.