How to Support Internet-based Distribution of Video on Demand to Portable Devices

Paolo Bellavista, Antonio Corradi
Dip. Elettronica, Informatica e Sistemistica - Università di Bologna
{pbellavista, acorradi}@ deis.unibo.it

Abstract

The increasing diffusion of mobile computing and of portable devices with wireless connectivity identifies new challenging scenarios for service provisioning. The access from devices with limited heterogeneous capabilities to traditional and novel Internet services requires new infrastructures capable of integrating with the fixed network and of supporting service tailoring/adaptation. The paper presents a mobile agent-based middleware for the distribution of Video on Demand (VoD) to portable devices. Mobile agents can act as device proxies over the fixed network, can negotiate the proper QoS level and can dynamically tailor VoD flows depending on profiles of terminal characteristics and user preferences. The paper also describes the design and implementation of a movie-info service prototype, built on top of the proposed middleware. The prototype shows the feasibility of distributing movie trailers ubiquitously even to portable devices with strict constraints on computing power and visualization capabilities, e.g., Palm personal digital assistants hosting the Java KVM/CLDC/MIDP software suite.

1. Introduction

The enlarging market of portable devices and the recent advances of wireless networking stimulate the provisioning of Internet services to a wide set of client terminals with very heterogeneous and limited resources, from computing power to visualization capabilities, from local memory to available network bandwidth. Several technical challenges need to be addressed for enabling the integration of portable devices with the fixed network infrastructure to access both traditional Internet services and new service classes, in particular, location-dependent services [1]. Several solutions have recently emerged, at the network level, to provide connectivity in mobile scenarios [2, 3]. However, several issues related to supporting mobile computing in the global and open Internet environment call for middleware solutions at the application level. Notable examples are dynamic un/installation of infrastructure/service components, configuration management, resource accounting, security, and interoperability [4, 5].

Several application-level issues are connected to device mobility. Portable devices usually roam among localities in an unpredictable way and cannot assume any a-priori knowledge about the locally available infrastructure components, resources and services. They should be able to either bind to unknown resources/services in the current locality or to maintain remote connections with resources/services in previously visited localities, transparently to the user. Middlewares for portable device integration should permit both to discover automatically resources at runtime and to support easy access and utilization of highly heterogeneous components. Recent research activities have provided notable solutions for resource/service discovery, with different balances between efficiency, flexibility, scalability and abstraction level [6, 7]. However, these solutions are usually hard to apply to resource-constrained portable devices. For instance, Jini requires discovery clients either to run a Java Virtual Machine (JVM) or to associate them with fixed docking stations hosting a JVM on their behalf. This makes Jini more suitable for easily configuring wireless devices in office/home environments than for supporting the integration of highly mobile devices with the fixed network.

In addition to the above mobility challenges, other support issues stem from the specific constraints of current portable devices, i.e., strict limits on hardware/software equipment and high heterogeneity. On the one hand, limited processing power, memory and file system capacities make portable devices unsuitable for traditional services designed for the fixed network. Middleware solutions should dynamically tailor service provisioning to the specific characteristics of the client access device. Moreover, client limited memory suggest to dynamically deploy thin clients only when needed and to automatically discard them after service provision. On the other hand, portable devices exhibit a high heterogeneity of hardware/software capabilities, hosted operating systems, and supported network technologies. Heterogeneity makes harder the provision of statically tailored services and may raise configuration hassles for device users. Middlewares should answer the goal of automating device configuration management as much as possible, to hide away from final users the wide variety of implementation details stemming from device heterogeneity.

The paper proposes a middleware capable of extending the fixed Internet infrastructure to support the distribution of Video on Demand (VoD) to portable devices with strict limits on local resources. The main design idea is to dynamically deploy components that act on behalf of devices over the fixed network. These middleware components can operate autonomously to carry on service requests even in case of temporary device disconnection, can follow device movements, and can operate on VoD flows (usually downscaling the flow quality and/or transcoding the multimedia format) to fit specific device capabilities. The proposed solution, called Mobile UbiQoS (MU) is the result of the extension to portable devices of the existing ubiQoS middleware for the tailoring, control and adaptation of Quality of Service (QoS) in the distribution of VoD flows over best-effort global networks [8]. MU components are implemented in terms of Mobile Agents (MAs), i.e., active entities that can migrate from one network node to one another during their execution by carrying their code and by preserving their reached execution state. They are able to exploit the MA properties of mobility, asynchronicity and autonomy during service provision. The MA technology enables the deployment of the MU middleware by enhancing the fixed network and by extending infrastructure services only when and where needed, without imposing any operation suspension [9].

The paper reports about the usability and effectiveness of the approach by presenting how to design and implement VoD services on top of MU: it describes a movie-info service prototype to carry to portable devices the information about movies currently shown at the theaters within the client locality. The service can distribute not only the movie title list (with theater addresses and timetables) but also movie trailers even to portable devices, such as Palm personal digital assistants hosting the Java KVM/CLDC/MIDP software suite.

2. Discovery and Tailoring: are they Viable for Portable Devices?

Portable device accessibility to Internet services requires addressing two classes of application-level issues. The former relates to the possibility of dynamically retrieving the set of resources/services available in the current client locality without any a-priori knowledge. The latter deals with resource limits and heterogeneity of portable devices that usually implies downscaling the set and the presentation format of service results to fit specific client characteristics. Recent research activities have explored discovery solutions and profile-based content tailoring to face these issues. However, we intend to point out that these proposals tend not to fit well the emerging market of limited and heterogeneous portable devices.

2.1. Discovery Services

Several solutions have tackled the problem of dynamically finding the resources available in a network locality to successively operate with them without a static knowledge. These solutions, usually identified as *discovery services*, are available with different features and follow various technical approaches.

Among commercial ones, the UPnP Simple Service Discovery Protocol (SSDP) [10] is a representative of simple effective solutions at a low level of abstraction. SSDP aims at identifying resources available in a network locality by exploiting UDP multicast. Only in a second phase it is possible to retrieve representations of the offered services by requesting XML-based descriptors based on standardized templates. The simple and effective multicast protocol is suitable for portable devices, but does not scale well for global environments. In addition, XML does not fit the limited parsing possibilities of portable devices.

The IETF Service Location Protocol (SLP) [11] and the Salutation suite [12], instead, define more complex and articulated architectures of middleware components, with rich functions to query advertised services by attributes. SLP works over IP to implement a naming system based on <attribute, value> pairs. It achieves good scalability by employing specific naming resolution components called directory agents. SLP directory agents, usually consisting of LDAP directory servers, are in charge of maintaining associations between resource descriptions and IP addresses. The SLP discovery protocol is light and suits also devices with limited capabilities, but it does not address interoperability of devices and retrieved resources. The Salutation solution is able to work over different network protocols because Salutation Managers are separated away from the underlying network layer by a common Transport Manager interface. Salutation also describes how clients can operate with retrieved service components: the suite defines different protocols, with different degrees of manager intervention, from simple session starting (Native Personality) to full control of data formats and communication (Salutation Personality). The weak point of the Salutation architecture for portable devices is the complexity of its managers that concentrate in a unique component all the functions that SLP distributes to service agents, user agents and directory agents. This well-known problem is pushing Salutation to define a specific version for portable devices, called Salutation-Lite [13].

The Jini discovery is interesting because language homogeneity permits not only to advertise/discover resources but also to distribute information about their usage (via interface proxy objects dynamically migrated towards the clients) [14]. However, Jini requires clients co-located with (or at least close to) a complete JVM: this assumption is too strict for most highly mobile portable devices.

Other commercial discovery solutions exist and keep on emerging, such as the Bluetooth Service Discovery Protocol [15], and aim at covering all different levels of abstraction available for service developers. According to this perspective, the Bluetooth discovery can be classified as intermediate between UPnP and SLP solutions.

Recent discovery research activities confirm that, at the state of the art, commercial solutions cannot satisfy the needs of portable devices. DEAPspace investigates completely decentralized discovery solutions, specifically suited to wireless ad hoc networks [7]. The Dynamic Mobility Agent proposal organizes localities in a hierarchy to achieve global scalability [7]. A relevant step in the evolution of discovery solutions for limited devices is the Jini Surrogate project that fills the gap between Jini discovery and roaming devices with no JVM. It specifies Jini-enabled surrogate components acting as docking stations to be dynamically retrieved by portable devices via low-level Bluetooth discovery [16].

2.2. Profile-based Negotiation and Tailoring

The enlarging market of portable devices is forcing service providers to face extremely heterogeneous access terminals, in terms of hardware resources, operating systems and supported software. Heterogeneity motivates the effort to define and recognize a limited number of device classes with similar characteristics.

For instance, Sun is pushing the adoption of Java programming for embedded/portable devices, by proposing different and limited versions of the JVM (the KVM for devices with 128KB to 512KB of memory, and the CVM for devices over 2MB). Within these recognized JVMs, some configurations and profiles are introduced. Configurations consist of a set of common horizontal facilities (Connected Limited Device Configuration - CLDC, and Connected Device Configuration - CDC), while configuration-based profiles define additional functions for specific market segments (Mobile Information Device Profile - MIDP - for cell phone and pagers, Foundation profile for next-generation consumer devices such as UMTS terminals) [17]. Service implementations can be tuned to a specific profile and are portable to all devices supporting that profile. The MU middleware addresses devices that can host the KVM/CLDC/MIDP software suite, i.e., devices belonging to the classmark 3 according to the specifications of the Mobile EXecution Environment forum [18].

The heterogeneity of client devices requires having mechanisms and technologies to maintain and dynamically retrieve interoperable profiling information about terminal characteristics (and user preferences). The IETF Content Negotiation Working Group (Conneg) promotes a profile representation format and an extension to standard HTTP (Transparent Content Negotiation) to enable profile-based content negotiation [19]. Conneg profiles consist of features (standardized properties with registered ASN.1 unique identifiers) and describe not only client terminals but also resources and service components. Conneg specifies also algorithms to match service requirements with user/terminal profiles at service negotiation.

The other main activity in profile standardization is the Composite Capabilities/Preference Profiles (CC/PP) by the W3C Device Independence Activity group [20]. Similarly to Conneg, the CC/PP solution is not tied to a specific communication protocol, but its diffusion is due to its adoption in Wireless Application Protocol mobile phones. CC/PP addresses more limited issues than Conneg, but fits better resource-constrained execution environments with wireless connectivity: in fact, it only focuses on device properties, without defining complex matching algorithms. CC/PP supports profiles by composition and allows to put them together during service negotiation on the client/server/proxy side and suggesting the storage of (parts of) profiles in a completely decentralized way. This permits to define more concise profiles also as deviations from default (vendor-based) ones and to cache distributed copies of diffused profiles, thus reducing the network traffic due to profile exchange during the negotiation phase.

```
<?xml version="1.0"?>
<RDF xmlns=http://www.w3.org/1999/02/22-rdf-
     syntax-ns#
   xmlns:rdf=http://www.w3.org/1999/02/22-rdf-
     syntax-ns#
   xmlns:ccpp=http://www.w3.org/2000/07/04-ccpp#
   xmlns:ccpp-client=http://www.w3.org/2000/07/
     04-ccpp-client#
<Description about="ldap://lia.deis.unibo.it/MU/
   MyProfile">
   <ccpp:component>
<Description about="ldap://lia.deis.unibo.it/MU/
   TerminalSoftware">
<type  resource="ldap://lia.deis.unibo.it/Schema
   #Software-Platform">
<ccpp-client: name>Palm OS</prf: OS>
<ccpp-client: version>???</prf: OS>
<ccpp-client: virtual machine>KVM</prf: Java>
<ccpp-client: configuration>CLDC</prf: Java>
<ccpp-client: profile>MIDP</prf: Java>
</Description>
</ccpp:component>
…
```

Figure 1. CC/PP-compliant MU profile for a PalmOS device with KVM/CLDC/MIDP

The MU middleware adopts CC/PP to represent metadata about hardware/software characteristics of client devices. For instance, the MU profile in Figure 1 describes a PalmOS portable device hosting the KVM/CLDC/MIDP software suite.

3. MU: an MA-based Middleware for VoD Services to Portable Devices

The provision of Internet services to portable devices requires downsizing service content to the characteristics of the access terminal. In particular, dynamic content negotiation and tailoring are crucial for resource-consuming services such as VoD distribution. An important design choice is where to operate VoD tailoring, typically reducing the frame rate/resolution and transcoding the format. In server-based solutions, the host that provides the service is in charge of either selecting the version of the requested VoD flow with the most suitable QoS level among a pool of statically pre-determined ones (off-line tailoring) or dynamically operating downscale transformations on the unique high-quality stored version (on-the-fly tailoring). These solutions concentrate service and tailoring functions on the server side: this makes distributed caching ineffective because provided VoD flows are tailored from the beginning to fit the specific client characteristics.

On the opposite, client-based tailoring imposes more distributed intelligence of VoD flow transformations and assumes a significant amount of computational resources to exploit on the client side. In this case, it is the client that requires VoD flows with the highest QoS level over the network infrastructure; this can cause overhead in the network traffic but may improve the effectiveness of distributed caching solutions. However, portable devices with limited resources clash with client-based tailoring.

In the global Internet, the portable device accessibility to VoD services requires a distributed and decentralized infrastructure fully hosted in the fixed network and consisting of middleware proxy components working on behalf of the device. Proxies are in charge not only of supporting device connectivity and of discovering resources/services in the current device locality, but also of tailoring the QoS level of VoD flows depending on device profiles. In addition, proxies should follow device movements during provision and install automatically, only when needed, in any newly visited network locality.

The MU middleware provides any portable device with one companion entity, called *shadow proxy*, and with several application-specific processors, called *QoS adapters*. Shadow proxies and QoS adapters are implemented as mobile agents, built on top of the SOMA programming platform (additional information and the SOMA code are available for download at `http://lia.deis.unibo.it/Research/SOMA`). The shadow proxy exploits discovery to retrieve local resources and service components, possibly negotiates tailoring parameters, acts on behalf of the device in case of disconnection, and can follow the devices independently of their movement between network localities. QoS adapters can operate QoS management operations on VoD flows depending on user/device profiles and can exploit MA mobility to keep co-located (or at least close to) with the shadow proxies they work for.

Let us note that the importance of proxies that move close to clients is recognized also in the Jini discovery solution where proxy objects are dynamically installed at the JVM on the client side to handle the interaction with remote services. Most current portable devices do not host any version of the JVM; even in case of devices supporting limited versions of the Java programming environment, such as the KVM/CLDC/MIDP, there are severe constraints on code mobility because of the rigidity of class loading. A notable example of the limitation is the impossibility of overriding the embedded class loader to instantiate a newly programmer-defined one. Differently from Jini proxies, the MU components can migrate together with their code and their reached execution state to follow portable devices during service provisioning.

3.1. The MU Middleware Components

The possibility to define suitable and flexible network localities is crucial for the development and deployment of Internet services to portable devices. MU offers locality abstractions to describe any kind of interconnected system, from simple Intranet LANs to the Internet (see Figure 2). Any node hosts at least one place for agent execution; several places are grouped into domain abstractions that correspond to either fixed or wireless network localities. In each domain, a default place is in charge of inter-domain routing. MU locality abstractions permit to define loosely coupled localities organized in a hierarchical way; this locality scenario naturally maps into discovery services with the visibility scope of one single domain [5, 6].

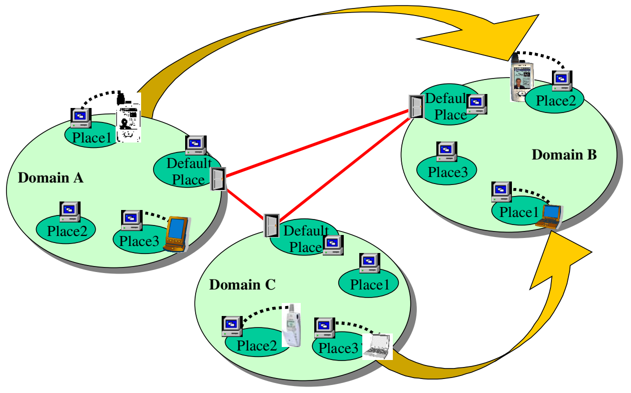


Figure 2. Device mobility among MU localities

The MU middleware consists of different components, as depicted in Figure 3. The ***shadow proxy*** represents a portable device and solves the problems of keeping information on behalf of the devices with intermittent connections and system limitations. The proxy retrieves the profile of its companion device and its current user. The profile information plays a central role in service discovery at the MU Portable Device Lookup Service (see the following). In addition, the MU middleware detects portable device inter-domain movements and triggers corresponding migrations of the shadow proxies to follow the companion device. The proxy keeps the reached state to implement migration of a service session during service provisioning. The shadow proxy is implemented as an MA running within the locality of the MU domain where the portable device is currently attached. Several proxies for different devices can execute concurrently on the same node without interfering with each other because the underlying SOMA platform provides isolated execution environments and separated security domains. We usually associate one shadow proxy for each portable device (with a 1-to-1 mapping). However, it is also possible to define group shadow proxies in charge of managing a set of portable devices with synchronization constraints, e.g., when distributing a synchronized multimedia guide to a tourist group visiting the rooms of a museum.

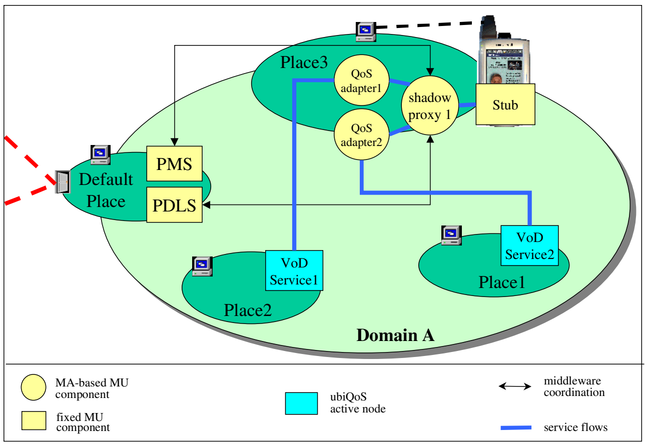


Figure 3. The MU middleware while distributing VoD to portable devices

QoS adapters are in charge of compression, e.g., reduction of frame size/rate, and format transcoding, e.g., from MPEG-1 to H.263, on VoD contents to tune the provided QoS level to suit the profile of device characteristics and user preferences. QoS adapters are implemented as SOMA agents that can follow the movements of their shadow proxy. They receive multimedia flows, operate transformations on them and forward processed flows to device-specific VoD players. The current implementation of QoS adapters is based on SUN Java Media Framework (JMF) for multimedia reception, transmission and processing [21]. For the transport and control of packet flows, QoS adapters exploit the JMF APIs to integrate with the Real-time Transport Protocol (RTP) and its corresponding RTCP control protocol [22]. Any MU component is generally portable on any platform that hosts a JVM. For performance sake, QoS adapters sometimes exploit local plug-ins available as native components. To achieve portability, they retrieve dynamically the list of plug-ins installed on their hosting execution environment to bind only to the available components.

Device/player-specific stubs run on portable devices and are in charge of announcing the device entering/exiting into a network locality, of transferring VoD flow requests from locally installed device-specific players to associated shadow proxies, of receiving multimedia flows and of forwarding them to the local players. We have currently implemented two different stubs. The first one is based on the KVM/CLDC/MIDP suite for Palm devices, with either USB or modem connectivity, and acts as a stub for the TealPoint TealMovie client [23] (see the movie-info prototype in the following). The second stub runs on top of the Java standard distribution over Compaq personal digital assistants with IEEE 802.11b support, and interworks with the standard JMF video player.

MU includes also two non-moving middleware components, called Portable Device Lookup Service and Profile Manager Service. They are the only infrastructure components that any participating domain should install statically to participate in VoD service distribution. The ***Portable Device Lookup Service*** (PDLS) is responsible both for sensing when a portable device enters its MU domain and for managing tailored lookup requests. Triggered by device arrivals, PDLS asks the SOMA mobility-enabled naming system [5] if the shadow proxy of that device is already running somewhere in the global system. If the proxy is already active, PDLS triggers the proxy migration to its network locality. Otherwise, PDLS instantiates a new local shadow proxy for the device. Internet service providers willing to integrate with the MU infrastructure simply register to PDLS, possibly by providing additional service-specific QoS adapters. When shadow proxies ask for services, PDLS does not provide a reference to the service that can carry the request (the usual lookup services behavior) but only a reference to a suitable QoS adapter able to act as the intermediate between the shadow proxy and the actual service component. PDLS is based on Jini and extends the SUN Reggie reference implementation of the lookup server [14]. It additionally considers the user/device profiles carried by the shadow proxy to identify the needed QoS adapter, then binds it to the requested service component, and finally triggers the adapter migration to the proxy.

The ***Profile Manager Service*** (PMS) maintains profiles of supported devices and registered users. It implements a partitioned and partially replicated directory service specialized for profiles. PMS maintains local copies of profile information and is able to coordinate with PMSs in other domains, via either LDAP or HTTP, to provide global profile visibility to shadow proxies. Profiles are expressed according to CC/PP; they can also consist of parts dynamically retrieved, even from different sites, and merged together during the negotiation phase.

The above infrastructure of MU components permits to distribute and tailor VoD flows within a network locality. This infrastructure is tightly integrated with our active middleware for QoS tailoring, control and adaptation of VoD flows over best-effort networks, called ubiQoS. ubiQoS components can dynamically build an active path between the VoD source and its group of receivers: active nodes negotiate the QoS level on any path segment depending on client requirements (user preferences and terminal characteristics). Active nodes monitor and control QoS levels during provisioning and react promptly with service management operations in case of network congestion. In addition, any active node can cache traversing VoD flows before their possible local downscaling, thus permitting the realization of an effective distributed caching infrastructure [8]. MU extends significantly the applicability of the ubiQoS middleware in mobile computing, by enabling the integration of portable access devices with wireless connectivity and strict limits on local resources.

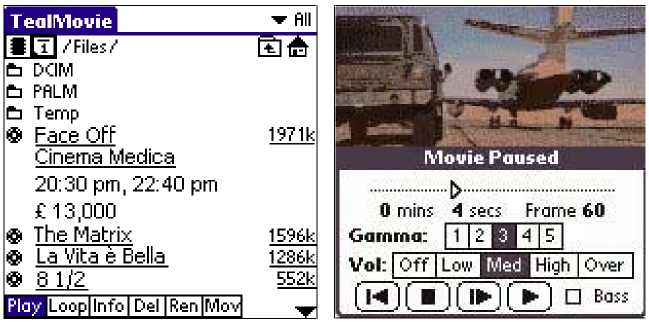
4. The Movie-Info Service in MU

We have designed and implemented a movie-info service prototype to make portable devices access suitable information about the movies shown at the theatres in their current location areas. The movie-info service is an example of a service deployed within the MU middleware, also with the goal of testing the usability and effectiveness of the support in a real application scenario.

Any MU domain models a different area. In any domain, there is at least one place able to give wireless connectivity to the portable devices hosted in the locality. When a portable device enters a new domain, the installed stub announces its arrival and the associated shadow proxy either is instantiated or moves to the place currently supporting the device connectivity. In the case of new instantiation, the proxy authenticates the device client and requests device/user profiles to PMS. When the device client requests the movie-info service, the proxy interrogates PDLS by providing user/device profile information; PDLS answers by triggering the migration of a suitable QoS adapter to the proxy. The QoS adapter requests the movie list to a movie-info server available in its domain. The server can maintain locally the information about the movies currently on in its area, or can request part of the data, e.g., movie trailers, to distributed remote servers, such as the film producers' Web sites. In the latter case, the movie-info server component local to the QoS adapter acts as the first active node in the active path dynamically established by the ubiQoS middleware components [8].

Let us focus on how Palm terminals specifically access the movie-info service, being the most challenging case to deal with device resource limitations. We only assume that Palm devices host the MU KVM/CLDC/MIDP stub and the TealMovie proprietary client. TealMovie is one of the first multimedia player for Palm and permits to play multimedia files represented in a suitable proprietary format. In the original TealMovie solution framework, it is necessary to statically convert AVI or WAV files to the proprietary format on a standard Windows host. In our solution, the format transcoding is made dynamically by a specific QoS adapter, retrieved automatically and transparently via PDLS at the trailer request, when the device profile specifies that the access terminal is a Palm device with the TealMovie client installed. Figure 4 shows screenshots of the movie-info prototype when visualizing the movie list and the "Face Off" trailer.

Apart from Compaq stub and video player based on the standard Java edition, we have also implemented other simple QoS adapters capable of discarding parts of the movie-info results (trailers and fixed images of movie posters) when the device profile indicates that the access terminal cannot support their visualization. A more precise description of these components is out of the scope of this paper, which specifically focuses on VoD distribution.


TealMovie
/Files/
DCIM
PALM
Temp
Face Off 1971k
Cinema Medica
20:30 pm, 22:40 pm
£ 13,000
The Matrix 1596k
La Vita è Bella 1286k
8 1/2 552k

Movie Paused
0 mins 4 secs Frame 60

Figure 4. Screenshots of the MU Palm client

Let us finally note that the MU middleware has enabled portable devices to access Internet services without forcing to modify service design and implementation. In particular, in the case study of the movie-info service, the service component is a standard HTTP server that maintains textual information about the movies currently on in the area, fixed images of their posters, and their multimedia trailers in either the WAV format or the AVI one. Web pages are expressed in XML according to the standard Dublin Core Metadata specifications [24].

In addition, there is no need to keep the portable device connected to the MU components on the fixed network. The shadow proxy can obtain autonomously the suitable QoS adapter and it can adapt autonomously the movie information. When the proxy receives service results, it forwards them immediately to the stub if the device is connected; otherwise, it caches them locally and waits for device reconnection, either in the same domain or somewhere else. In the latter case, the shadow proxy migrates to the new domain and asks whether the device client is still interested in receiving the movie information requested in the previous area or it prefers to update the movie list to its new location.

5. Conclusions and Current Work

The provision of traditional and novel Internet services (in particular of location-dependent services with differentiated and controlled QoS) to portable devices requires facing several specific issues, mainly deriving from device mobility, heterogeneity and limited resources. This scenario strongly suggests the organization of a dynamic support infrastructure capable of tailoring service provisioning to the specific characteristics of mobile access terminals. Mobile agents are a suitable and effective technology to implement such an infrastructure, because of their intrinsic ability of extending the fixed Internet, where and when needed, by following client movements.

The first encouraging results obtained by the MU middleware implementation, both in terms of feasibility and usability, are stimulating additional research. We are collecting experimental results about the MU performance in different large-scale scenarios of real service provisioning. We are extending MU with adaptive strategies for VoD caching in intermediate nodes, with pre-fetching driven by mobility prediction based on common mobility patterns, and with new identification policies of optimal locations for downscaling actions. Finally, we are applying the same proxy-based tailoring approach to other application domains. We are working on the development of context-aware services adapting to client context (location, user preferences and role, terminal profile of characteristics, ...) and to availability of distributed resources (computational load, available network bandwidth, ...). In particular, we are completing the prototype of a Museum Assistant Service that retrieves information about the artworks exposed in the currently visited museum room and fits the presentation to both the access device characteristics and the visitor expertise level/interests.

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] C. Jyh-Cheng, K. M. Sivalingam, P. Agrawal, R. Acharya, "Scheduling Multimedia Services in a Low-Power MAC for Wireless and Mobile ATM Networks", *IEEE Trans. Multimedia*, Vol. 1, No. 2, June 1999, pp. 187-201.

[2] C. Perkins (ed.), Special Section on “Autoconfiguration”, *IEEE Internet Computing*, Vol. 3, No. 4, July 1999.

[3] L. Bos, S. Leroy, “Toward an All-IP-based UMTS System Architecture”, *IEEE Network*, Vol. 15, No. 1, Jan. 2001.

[4] R. Oppliger, "Security at the Internet Layer", *IEEE Computer*, Vol. 31, No. 9, Sep. 1998, pp. 43-47.

[5] P. Bellavista, A. Corradi, C. Stefanelli, “Mobile Agent Middleware for Mobile Computing”, *IEEE Computer*, Vol. 34, No. 3, March 2001, pp. 73-81.

[6] G. G. Richard III, “Service Advertisement and Discovery: Enabling Universal Device Cooperation”, *IEEE Internet Computing*, Vol. 4, No. 5, Sep.-Oct. 2000, pp. 18-26.

[7] S. K. S. Gupta, W.-C. Lee, A. Purakayastha, P. K. Srimani (eds.), Special Section on “An Overview of Pervasive Computing”, *IEEE Personal Communications*, Vol. 8, No. 4, Aug. 2001, pp. 16-59.

[8] F. Baschieri, P. Bellavista, A. Corradi, “Mobile Agents 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)*, IEEE Computer Society Press, Japan, Feb. 2002.

[9] A. Fuggetta, G. P. Picco, G. Vigna, "Understanding Code Mobility", *IEEE Transactions on Software Engineering*, Vol. 24, No. 5, May 1998, pp. 342-361.

[10] Universal Plug and Play (UPnP) Forum, http://www.upnp.org

[11] Sun Microsystems Inc. - *The Service Location Protocol (SLP)*, http://www.srvloc.org

[12] The Salutation Consortium, http://www.salutation.org

[13] The Salutation Consortium - *The Salutation-Lite White Paper*, http://www.salutation.org/whitepaper/Sal-Lite.PDF

[14] Sun Microsystems Inc. - *Jini*, http://developer.jini.org

[15] P. Johansson, M. Kazantzidis, R. Kapoor, M. Gerla, "Bluetooth: an Enabler for Personal Area Networking", *IEEE Network*, Vol. 15, No. 5, Sept.-Oct. 2001, pp. 28-37.

[16] Sun Microsystems Inc. - *The Jini Surrogate Project*, http://developer.jini.org/exchange/projects/surrogate

[17] Sun Microsystems Inc. - *The Java 2 Platform Micro Edition (J2ME)*, http://java.sun.com/j2me/

[18] The Mobile EXecution Environment (Mexe) Forum, http://www.mexeforum.org

[19] IETF Content Negotiation (Conneg) Working Group, http://www.imc.org/ietf-medfree

[20] WWW Consortium - Composite Capabilities/Preference Profile (CC/PP), http://www.w3.org/ Mobile/

[21] Sun Microsystems Inc. - *The Java Media Framework (JMF) API*, http://java.sun.com/products/java-media/jmf/

[22] T. Braun, "Internet Protocols for Multimedia Communications - Resource Reservation, Transport, and Application Protocols", *IEEE Multimedia*, Vol. 4, No. 4, 1997.

[23] TealPoint Software - *TealMovie*, http://www.tealpoint.com

[24] Dublin Core Metadata Initiative, http://au.dublincore.org