Journal of Network and Computer Applications 34 (2011) 121–136

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

User-provided multimedia content distribution architecture in mobile and ubiquitous communication networks ☆

Chih-Lin Hu *, Chien-An Cho

Department of Communication Engineering, National Central University, No. 300, Jung-da Rd, Jung-li City, Taoyuan, Taiwan 32001, ROC

ARTICLE INFO

Article history:
Received 3 March 2010
Received in revised form
27 July 2010
Accepted 19 August 2010

Keywords:
Device discovery
Mobile content delivery
Mobile application
Multimedia distribution
Ubiquitous computing

ABSTRACT

The network convergence of wired, wireless, and mobile systems creates a ubiquitous network environment where modern networking devices feature multiple networking interfaces and can connect to different networks simultaneously. Mobile users in ubiquitous networks expect to access information services anytime, and from anywhere. This paper presents a mobile content sharing scenario in which a networked device can discover neighboring devices and share multimedia content in a convenient, networked manner. This ideal scenario differs from the traditional usage which requires the tedious manual operations of connection setup and file transfer. To achieve this goal, this study proposes a user-provided multimedia content distribution architecture for a mobile and ubiquitous network environment. The proposed architecture integrates several specific mechanisms, including device discovery, asynchronous content delivery, secure access control, and virtual file system. This design addresses several inherent limitations in wireless and mobile networks, and enables mobile users to transfer media content in a secure, mobile, and energy-saving way. The proof-of-concept prototype and performance evaluation in this study confirm that the proposed architecture not only provides better user experience, but also achieves a lightweight design without compromising performance.

1. Introduction

Recent advances in network communication technologies and mobile handheld devices (MHDs)[1] have provided users with a ubiquitous network environment, enabling access to numerous information services at any time and from anywhere. Many modern MHDs have at least dual or triple network connectivity modules—for example, mobile phones may have 3G, Wi-Fi, and Bluetooth. Since the device can attach to multiple networks simultaneously, it can run different network services concurrently in different networks. Combining these network services might lead to further collaborative network services (Moriya et al., 2007; Ohnishi et al., 2007). This area remains the topic of future research.

Users with MHDs, a.k.a. mobile users, now access Internet and Web services much like they used to on desktop computers. In addition to downloading Internet files and media content, MHDs upload user-created multimedia content. In fact, many MHDs include a digital still camera, audio recording capabilities, and audio/video (A/V) processing modules, with which users can conveniently and effectively take pictures, record audio sounds, or shoot video clips. Using external storage space, MHDs can store a large volume of media content. Therefore, it seems natural and inevitable that people will exchange, share, and publish media content among various networked devices in social communities (Belimpasakis et al., 2008; Hu et al., 2008).

The convergence of fixed, wireless and mobile network systems has created a new playground for ubiquitous content applications and services. Figure 1 illustrates that multiple wireless networks often surround an MHD: GSM, 3G, 3.5G, WiMAX, Wi-Fi, and Bluetooth PAN. An MHD with multiple networking interfaces can connect to different networks simultaneously. As a result of this improved user experience, mobile users now expect to access information services anytime and anywhere. Accordingly, *enabling user-provided multimedia content sharing across wireless and mobile-networked devices in a ubiquitous network environment* is an interesting possibility in mobile applications. Section 1.1 describes the ideation of a novel mobile multimedia content sharing scenario. Section 1.2 describes the proposed architecture and development to realize this design.

1.1. Ideation and scenario

Consider the evolution of wireless broadband network systems, such as IEEE 802.11a/g/n, UMTS, LTE, and WiMAX.

☆This work was supported in part by the National Science Council of Taiwan, R.O.C., under Contract NSC98-2221-E-008-041.

* Corresponding author. Tel.: +886 3 4227151x35513; fax: +886 3 4229187.
E-mail addresses: clhu@ce.ncu.edu.tw, clhu@ieee.org (C.-L. Hu), 955003020@cc.ncu.edu.tw (C.-A. Cho).

[1] An MHD, such as mobile phone, personal digital assistant, mobile Internet device, or ultra mobile PC, is the major product concept of portable information devices. Without loss of generality, we use this term to implicitly represent all types of wireless and mobile-networked devices.

1084-8045/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jnca.2010.08.010

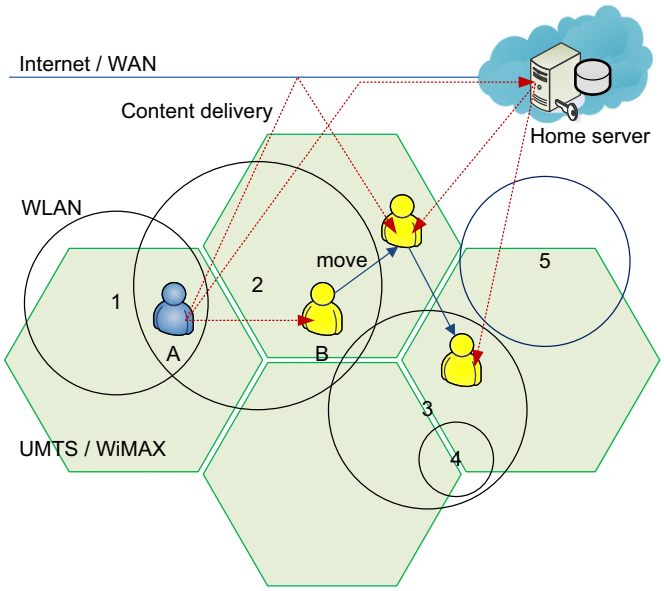

Fig. 1. A ubiquitous computing environment under convergence of fixed, wireless, and mobile networks.

Significant increases in bandwidth utilization and transmission in these system have so far satisfied the minimal conditions of multimedia content services. Users with modern MHDs can enjoy Internet and Web file access, and A/V content delivery and streaming. However, the intrinsic characteristics and constraints of wireless and mobile networks influence the design of mobile and ubiquitous applications.

First, wireless communication systems have *lower transmission throughput* than high-speed LAN/WAN systems. Second, energy consumption is critical because MHDs are battery-powered with *hourly energy capacity*. Third, a *limited single-hop transmission range*, which is dozens of meters in a PAN or WLAN, confines an MHD's movement within a certain coverage area to retain the connection and communication in a singly managed network. Finally, most wireless communication systems have *no support of service mobility*. In general, there is no monolithic system dedicated to mobile and ubiquitous applications. A ubiquitous network environment involves different heterogeneous network systems with asymmetric capabilities in bandwidth, transmission rate, coverage size, mobility, QoS, and security. Due to underlying network heterogeneity and interoperability, standard TCP/IP and HTTP functionalities support a uniform platform for application-level network service design and deployment.

The goal of this paper is to develop a user-provided multimedia content distribution system for a ubiquitous network environment. From the user-centric and application-level standpoints, the design of the scenarios below should satisfy some functional requirements, including efficiency, security, energy-savings, and user experience, in the varying conditions of underlying networks and target devices.

Scenario 1. Figure 1 depicts a ubiquitous network environment in which separate network systems and their coverage areas overlap. In this scenario, mobile users operate MHDs capable of multi-radio connectivity, e.g. Wi-Fi and UMTS. Suppose two friends, A and B, meet at a rendezvous place such as a classroom, office, coffeehouse, or home. A has some interesting photos and video records in her MHD that she would like to share with B. A and B first find and connect to a common Wi-Fi network around, i.e., WLAN-2. A's MHD discovers B's MHD in WLAN-2, and then automatically negotiates with B's MHD to transfer shared media objects.

Scenario 2. If B leaves during this media transferring, the content transfer remains incomplete. In this case, A redirects B's MHD to download the remaining content from her home server. A's MHD then immediately requests the home server, via the UMTS or WLAN associated with the back-end network, to prepare a secure download transaction. A then provides B with the certificates required to process the secure transaction with her home server. After leaving WLAN-2, B's MHD can download the remaining content from A's home server through other available networks though A and B are now in different network domains. Figure 1 shows that when B's MHD leaves WLAN-2, it can continue downloading via UMTS. Alternatively, B can pause downloading until B's MHD connects to WLAN-3. This approach takes advantage of cost-effective Wi-Fi, if the secure download transaction is still valid.

Mobile users can fill the roles of mobile content receivers and mobile content providers in user-provided communication networks (Digital Living Network Alliance, 2006). A mobile content provider shares media content with mobile content receivers in a networked and automatic manner instead of the traditional series of manual operations for connection setup and media content transfer. Either party can move from one network location to another during content delivery. A specific secure transaction model for the home server and its users can ensure trustworthiness and validity, irrespective of which terminal devices and network attachment points are in use. This type of ongoing content delivery is robust against terminal mobility and network context switching. This method allows mobile users to use a cost-effective network channel in terms of user-centric and application metrics, such as service time, energy saving, and user experience.

1.2. Contribution

This study considers several design requirements to deal with the weaknesses of wireless and mobile networks: discovery of neighboring devices and services, asymmetric energy consumption, mobility and disconnection, and secure access control. Accordingly, this study proposes a user-provided multimedia content distribution architecture for an integrated network environment. The proposed architecture consists of four basic components:

- device discovery;
- asynchronous content delivery;
- secure access control and management;
- virtual file system.

Specifically, the *device discovery* stage is a prerequisite to finding resources of interest in neighboring networked devices within a network domain. Runtime resource discovery (Vanthournout et al., 2005) is a desirable replacement of manual configuration, which is not only awkward for users, but impossible in large, complex or various system contexts. Previous studies (Edwards, 2006; Vanthournout et al., 2005; Zhu et al., 2005) present many discovery protocols, some of which Section 2 reviews. For example, the Universal Plug and Play (UPnP) device architecture connects clusters of networked devices in ad hoc or unmanaged networks, especially in domestic environments (UPnP Forum, 2003). The UPnP technology leverages existing Internet standards, including TCP/IP, HTTP, and Web technologies, to enable peer-to-peer proximity networking, device discovery, event notification, control action, and data transfer functions. Networked devices can follow the UPnP conventions to provide UPnP devices and services. A companion study presents an AV media content sharing framework for *home* networks (Hu et al., 2008). This paper aims to expand the location restriction from local

area networks to any network, and simultaneously achieves the secure and ubiquitous content delivery.

One of significant goals of this architecture is to design the role of a home server[2] as a mobile content provider. The collaboration between the mobile content provider and its home server supports *asynchronous content delivery* in a ubiquitous network. By leveraging the power of the home server for rapid and inexpensive connectivity and secure content storage, this component is valuable in terms of cost-effectiveness, energy savings, and mobility support. Because MHDs are battery-powered, direct content delivery from one MHD to another networked device is expensive. Energy consumption is symmetric because both the provider and receiver expend energy during content delivery. In the proposed design, the mobile content provider can instruct the receiver to access the indicated media content from its home server, where the mobile content provider stores duplicates. This asynchronous delivery model provides the proposed architecture with three major benefits. First, a home server in the wired network can provide higher data throughput to shorten the transmission duration. Second, the mobile content provider can avoid long transmission duration and so reduce energy consumption. Finally, both the mobile content provider and the receiver can get rid of the movement limitations inherent in a single-hop transmission range; they are free to move about in the network. In this case, the receiver can still reach the stationary home server which in turn processes the follow-up content delivery in place of the mobile content provider. The receiver can continue to access media content from the home server even though the mobile content provider has roamed to another network or disconnected from the network.

This study also develops a *secure access control and management* scheme to establish a trusted environment for the mobile content provider, receiver, and home server before any "transaction" occurs between these entities. Terminal portability and mobility enable users to change their network attachment points. Therefore, the home server must be able to determine whether a mobile content receiver is trusted or not, even over public infrastructures beyond its administration boundaries. To achieve this purpose, this study uses a simple *double key identification* scheme with a pair of symmetric keys, i.e., a provider key and a transaction key. The provider key is pre-determined and stored by the mobile content provider. This key maintains the trustworthiness between a mobile content provider and its home server. In contrast, the transaction key is transient and assigned a valid deadline. The mobile content provider contacts its home server for a transaction key to each transaction. This key secures the temporary transaction between the mobile content receiver and home server. Note that this transaction key is opaque to the receiver and becomes invalidated immediately after the transaction is complete. This policy prevents distributing a transaction key to unauthorized receivers. In addition, the provider key can be refreshed to avoid being cracked whenever a mobile content provider completes a transaction.

The home server employs a *virtual file system* to process directory and file browsing, and other operations in support of secure content delivery. For each transaction, the home server prepares a temporary location reference at which the receiving client can make a connection to fetch a media object. Different transactions involving the same media object correspond to different location references. The home server uses this *virtual file system* and *dynamic reference mapping* to protect the file system from possible threats.

This paper describes the design and the development of a user-provided multimedia content distribution framework in a mobile and ubiquitous network environment. To realize a novel mobile content sharing scenario, the proposed design addresses several design concerns and requirements to cope with the characteristics and limitations of wireless and mobile networks. Accordingly, this study proposes an architecture reference that integrates device discovery, asynchronous content delivery, secure access control, and virtual file system software components. The proposed architecture adopts UPnP middleware for networked devices and service discovery. The home server fulfills asynchronous content delivery on behalf of the mobile content provider. A lightweight secure access control mechanism with simple double key identification guarantees a trusted transaction process, while a virtual file system with dynamic reference mapping guards against security threats. Prototype development and performance results demonstrate the feasibility and effects of the proposed software architecture.

The rest of this article is organized as follows. Section 2 briefly describes the background knowledge of discovery protocols, the UPnP technique, and related works. Section 3 details the design of the proposed architecture, and its components and functions. Section 4 describes the prototype development and implementation. Section 5 presents the performance results under practical experiments. Section 6 provides conclusions and directions for future research.

2. Preliminary and related works

Section 2.1 surveys several discovery protocols in the literature. Section 2.2 describes the UPnP technique and its potential drawbacks. Section 2.3 provides related studies on home servers and gateways in in-home, multi-home, and ubiquitous networks to offer a complete knowledge base. Finally, Section 2.4 summarizes several observations and discussions to distinguish this study from previous research.

2.1. Device and service discovery

Discovery methodology enables devices and services to discover, configure, and communicate with each other in ubiquitous computing environments. Devices and services dynamically refer to resources on the network rather than pre-configured resource bindings. Thus, the appropriate discovery methodology can minimize administrative overhead and increase usability.

Previous studies (Edwards, 2006; Vanthournout et al., 2005; Zhu et al., 2005) present a taxonomic survey of discovery protocols. The following discussion further reviews several discovery protocols based on LANs and single-hop wireless communications.[3] The Service Location Protocol (SLP) is oriented towards enterprise service discovery, and benefits by maintaining heavyweight directories. The salutation protocol is primarily used for special services with directory lookup and hard service states. Jini requires that all devices throughout a network support Java runtime capabilities, but is difficult to implement. Bonjour, popularized by Apple Inc., is an interesting proprietary solution that meshes closely with existing Internet standards while operating in the absence of the traditional managed Internet infrastructure. However, its special naming rule is

[2] A home server with stationary network connectivity and storage capacity can be one of many sorts of networked devices, including a desktop, network attached storage (NAS), and IPTV Set-Top-Box (STB) (Kim and Bahn, 2008).

[3] This paper primarily considers the use of device and service discovery protocols in singly administrative network environments, in which the proposed architecture enables user-provided content sharing among neighboring devices.

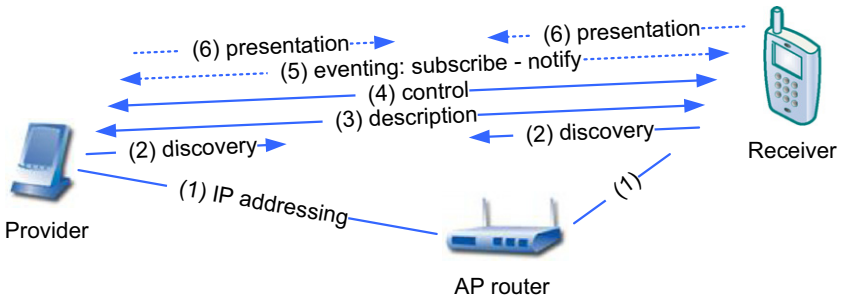

Fig. 2. UPnP devices' messaging and interaction flowchart.

not suitable for all types of networked devices in ubiquitous network environments. The Bluetooth communication protocol is based on actual physical proximity rather than closeness in an IP routing domain. Its effects of computation and bandwidth utilization only apply to profile-defined services. In contrast, UPnP complies with existing Internet conventions and *de facto* serves as an open "micro-middleware" that supports a variety of device categories and add-on services (UPnP Forum Standardization Device Control Protocols, 2010) in small IP networks.

The perspective adopted in this study coincides with the viewpoint of (Vanthournout et al., 2005), who described, "distributed resource discovery systems that support *dynamic and mobile resources* and use *attributed-based naming* as a main direction for future research in this area." The UPnP technology is qualified and designed for use in a singly administrative network context. More importantly, academic and industrial communities widely advocate the UPnP (UPnP Forum, 2003). The Digital Living Home Alliance (DLNA) (Digital Living Network Alliance, 2004) recognizes the UPnP technology as the core-networking framework for developing interoperable home-networked platforms and UPnP-specific devices and services in home networks.

2.2. UPnP device architecture and its discussion

The UPnP protocol provides a distributed, open networking architecture that contains two device categories: control points, and controlled devices or simply "devices." A controlled device functions as a server, offering services that a control point can monitored, or control. UPnP messaging between UPnP devices requires UPnP-specific Internet protocols, including Simple Service Discovery Protocol (SSDP) (Internet-Draft, 1999), Simple Object Access Protocol (SOAP) (W3C Consortium, 2000) and General Event Notification Architecture (GENA) (Internet-Draft, 2000).

UPnP devices perform six function layers in a bottom–up order, as Fig. 2 illustrates. *Addressing* is the underlying function that enables a device to acquire a unique network IP address when it first joins the network. The default addressing method is dynamic host configuration protocol (DHCP), while automatic IP configuration (Internet-Draft, 2004) functions in the absence of DHCP service. A device uses the *Description* layer to summarize its services and capabilities in a well-defined XML format, allowing control points to parse description files and know what a device offers. *Discovery* based on SSDP allows a device to advertise its appearance and services in the network, so a control point can access the device and its information. *Control* based on SOAP allows a device to handle requests from control points and invoke specific actions. Using *Eventing* with GENA, a device notifies the registered control points of any significant state changes. *Presentation* is the HTML-based interface that users use to control or monitor devices directly.

When a device is powered on, it first tries to obtain an IP address, organizes description files, and then broadcasts its device and service advertisements in a local network. When a control point finds the device, it can operate the device, and also subscribe to interested services at the device. The device responds to control actions and sends event notifications to the subscribed control point while any service state changes.

However, there are several performance problems in UPnP discovery, control, and eventing. Previous research (Liong and Ye, 2005) notes that UPnP discovery based on periodic SSDP messages can create network congestion and power dissipation of small electronics in an asymmetric home network; for example, a Bluetooth Personal Area Network bridged to an Ethernet or Wireless LAN with higher bandwidth. Although UPnP M-Search can control the jitter bound and avoid the message implosion caused by huge multicast queries and responses, it may perform poorly in large home networks. To alleviate this problem, previous researchers proposed an adaptive jitter control method to maintain performance by varying the combination of jitter bound and network size (Mills and Dabrowski, 2003). Under the UPnP control, SOAP can negatively affect the performance of Web service (Mazuryk and Lukkien, 2004; Saiedian and Mulkey, 2008; Zilora and Ketha, 2008). Its slow response time is primarily due to the processing overhead of its XML-based message and data encoding scheme, and the inefficiency of HTTP as a real-time transport protocol (Saiedian and Mulkey, 2008). Although the XML hardware acceleration and SOAP compression schemes can improve the overall response, previous research found that the appropriate selection of client software, server software, and data structures could have a significant effect (Zilora and Ketha, 2008). The work in Newmarch (2005) used the REST principles (Fielding, 2000) to design a coexistent REST-based method to improve SOAP response time. This method outperforms the original in terms of request-response time, message payload, and memory consumption. Furthermore, Mazuryk and Lukkien (2004) compared the UPnP protocols with a service-oriented architecture (Bieber and Carpenter, 2001), and examined the UPnP scalability problem in discovery, impractical binding of arguments in control, and inefficiency of eventing in the TCP manner. Their analytical results indicate that UPnP eventing is the weakest spot. Previous work (Hu et al., 2007) presents a compatible event multicast method that increases the efficiency of UPnP eventing.

2.3. Related research on home servers

A home server or a home/residential gateway[4] plays a very important role in home and ubiquitous computing environments (HGI, 2008; Garcia-Reinoso et al., 2009; Kim et al., 2008; Lawrence et al., 2003; Open Service Gateway Initiative (OSGi) Alliance, 2005). A home server provides inter-networking, quality of

[4] Hereafter the terms home server and home/residential gateway are used interchangeably, as remote home applications usually run in the home gateway platforms across heterogeneous networks, although they may not be identical in some cases.

service, and safe access for various intelligent applications and services across private and public networks in Internet and telecommunication systems. Many studies in this field present gateway designs involving in-home, multi-home, or remote and mobile home services. The following discussion briefly reviews many, but not all, previous studies.

Most research on this topic focuses on in-home and remote home automation systems.[5] The work in Kim et al. (2002) developed a home network system prototype based on UPnP middleware for controlling traditional home appliances inside a home network. Unlike the UPnP approach, the study in Al-Ali and Al-Rousan (2004) designed a Java-based home automation system for a home network, and focused on how to develop Java platforms, remote control, and management via web scripting and interface at PC hosts in the Internet. To construct a context-aware automatic home service, Choi and Shin (2005) developed a context-aware middleware based on Open Service Gateway initiative (OSGi) framework (Open Service Gateway Initiative (OSGi) Alliance, 2005). This middleware utilizes a neural-networked approach to learn a user preferences with several context attributes, and to predict and offer services in a smart home. Further, Kuriyama et al. (2006) developed a home appliance translator (HAT) and HAT-Sub architecture for controlling home appliances without network connectivity using protocol translation between X.10, new power line communication protocols, and SCP/UPnP bridging. The work in Rich et al. (2005) proposed a collaborative interface system to help users control traditional home appliances. Recently, Delphinanto et al. (2009) installed a UPnP proxy on an Home Gateway Initiative (HGI) gateway (HGI, 2008) to support the remote discovery and management of various devices in highly heterogeneous home networks. This approach uses a single data object to enclose and transmit various statuses and information to the remote server. These new configurations and security functions are not yet HGI specifications.

New progress in home gateway design attempts to provide multimedia content sharing among networked devices in in-home, multi-home, or ubiquitous network environments. DLNA v1.0 defines two basic devices, digital media player and server, and their playback functions in CE and PC products (Digital Living Network Alliance, 2004). DLNA v1.5 extends device categories and significantly includes mobile handheld devices (Digital Living Network Alliance, 2006). A related study[6] (Hu et al., 2008) introduces a new device type, i.e., a mobile media server, in a UPnP-based home network.

UPnP discovery depends on SSDP multicasting, and is confined to local area networks and managed subnetworks. Previous studies propose several mechanisms for allowing a home gateway to communicate with remote devices outside a home network, including the *message relay and tunneling*, *session initiation protocol* (*SIP*) *signaling*, *proxy* and *overlay* techniques.

The work in Lee and Kim (2007) designed a SHARE framework for content sharing in different UPnP-based home networks in which home gateways provide UPnP message relaying and virtual media server functions. In this framework, UPnP devices can play A/V streaming from remote devices through one or more in-between gateways. For heterogeneous home networks, (Kim et al., 2008) developed a multimedia service framework to inter-operate various A/V devices and provide multimedia services throughout the home regardless of physical constraints in a ubiquitous home network. In Oh et al. (2007), a DLNA-based media sharing system was proposed. This approach conveys control and information messages between a DLNA home gateway and its outside DLNA agents using SIP messaging. Though SIP can control QoS in the cases like voice and A/V streaming services, it is too heavyweight for home automation and best-effort content services, which have relatively simple interaction. Supporting SIP protocol, middleware, and user agents may complicate system design and increase processing load.

The work in Kim et al. (2007) designed a DLNA proxy on a gateway capable of sharing share in-home media content across public IP networks via secure channels on an IP security virtual private network (IPSec VPN). IPSec VPN is a progressive layer-3 network security mechanism. This protocol enforces end-to-end data packet security and integrity across the Internet. However, IPSec VPN suffers from some critical problems. First, IPSec may induce high computation overhead and slow down multimedia traffic, so Kim et al. (2007) adopted a hardware crypto-accelerator to decrease processing time. Second, a VPN limits user population to a special crowd (Motegi et al., 2008). Third, inside connected VPNs have the problem of no privacy and access control (Song et al., 2009). Therefore, Motegi et al. (2008) and Song et al. (2009) proposed different UPnP/DLNA proxies without regard to SIP and VPN concerns. Furthermore, Kawamoto et al. (2009) and Venkitaraman (2008) employed special peer-to-peer (P2P) overlay architectures on home gateways, among which the content search and retrieval operations comply with the P2P's conventions without loss of generality. Behind home gateways, home networks still use UPnP.

In Belimpasakis et al. (2008), this work proposed a mobile content sharing service for a group of mobile devices and a dedicated home server. Specifically, a user initializes a sharing session by requesting the home server to organize a sharing account and storage directory. The user then sends invitation messages including account and session information to other users via short message service (SMS). The user and invitees then exchange media content through the intermediate sharing directory. This allows a mobile user to use either Wi-Fi or GPRS/3G when accessing the content from the home server. Note that this approach is similar to the method proposed in this paper. The difference between these approaches is that (Belimpasakis et al., 2008) did not provide resource discovery, security management, or access control. Their special group membership organization, based on SMS, might also restrict the user community and usage cases.

2.4. Observation and discussion

The UPnP is suitable for small and local networks, including LAN, WLAN, in-home, and office networks. In spite of its potential problems, its fast connection and transmission in a local network can *implicitly* mitigate the concerns of resource utilization, processing computation, communication costs, and overheads due to UPnP messaging and representation. Unfortunately, when applications run in mobile and ubiquitous computing environments, or when applications run in mobile or resource-limited devices, several weak points in UPnP will be highlighted, as explained below.

Discovery based on SSDP messaging is unable to traverse multiple network domains. Though a home gateway often employs message relaying and tunneling mechanisms to intervene in communication with remote devices, these practices increase message overhead and communication expense. A VPN is another way to extend the scope of network service, but this service is exclusively available to a certain user group, and not suitable for people in public networks. Thus, a UPnP/DLNA proxy may be the best option. However, a specific communication protocol between remote devices and their proxy remains a sensitive issue. As mentioned above, SIP is a complicated and heavyweight protocol with real-time and QoS requirements.

[5] The controlled targets were ordinary household appliances, CEs, and PCs and their peripherals, and not simply mobile handheld devices dedicated to voice services in that context.

[6] This study focuses on A/V playback experience with MHDs in a UPnP-based home network.

Unless a target device has a lot of resources and requests real-time multimedia services, SIP is not a cost-effective approach. Accordingly, the design of an application-level communication protocol must account for the media and service type, protocol-specific message formation representation, target execution platform, security and safety, processing complexity, resource consumption, etc.

Most modern Internet and Web applications use SOAP to transmit control and management instructions over HTTP connections. The UPnP control layer in a home network also uses SOAP. This communication paradigm is not cost-effective due to its significant expense in processing and communication resources, and slow response time. Even though existing methods extend home services outside the home by relaying or tunneling in-home UPnP/DLNA messages to remote devices (Kawamoto et al., 2009; Kim et al., 2007; Motegi et al., 2008; Song et al., 2009), these problems remain and become worse in wide area networks. As Section 5 mentions, experimental results show that the UPnP/DLNA-specific SOAP control scheme creates a severe performance reduction in a wide-area network compared with that in a local network. Its poor performance may be the result of many factors (Fielding, 2000; Saiedian and Mulkey, 2008; Zilora and Ketha, 2008) as Section 2.2 describes. These problems call for a systematic solution or another method of substitute.

This paper proposes a "proof of concept" for a user-provided multimedia content distribution architecture in mobile and ubiquitous IP networks. To ensure compatibility and interoperability among home networked devices, the proposed design only uses UPnP inside a home network. Instead, it uses a separate, non-SOAP control scheme outside the home network. Accordingly, this paper designs an *in-band* XML-RPC messaging protocol that allows a home server to communicate with remote devices. Compared with SOAP, XML-RPC is compact, secure, and lightweight. The XML-RPC message representation includes authentication and verification information, and is opaque to mobile content receivers. This study successfully develops a prototype based on off-the-shelf mobile phone and PC platforms. While the design of a home server itself is important, it is parallel to the work of this paper. As a result, future research can migrate this software architecture to any target home gateway platforms, including OSGi and HGI.

3. Architecture design

This section describes the proposed architecture design and functionality. Section 3.1 gives a brief architecture overview. Section 3.2 describes several design considerations and requirements. Sections 3.3–3.6 present four inclusive software components: device and service discovery, asynchronous content delivery, secure access control, and virtual file system mechanisms.

3.1. Architecture overview

The objective of this architecture design is to make networked devices interconnected and to enable mobile content delivery anywhere in an integrated network playground. Consider the following user scenario: a user can use an MHD to easily and conveniently share stored media content to other networked devices, or to alternatively redirect them to download indicated media contents from the home server. This scenario requires that all mobile content *providers*, *receivers*, and *home servers* support the proposed architecture. Because this is a software architecture with none of any hardware-specific bundles, it can be deployed on any networked devices that support HTTP and TCP/IP protocol suites over IP-based network systems.

Figure 3 illustrates the functional blocks of the proposed architecture, which consists of four basic software components: UPnP middleware, content delivery service, secure access control and management, and virtual file system. First, the UPnP middleware performs the device discovery mechanism in a LAN, WLAN, ad hoc, or singly administrative network. UPnP-compliable devices can discover and learn what services the devices in the network support. The UPnP middleware provides IP networking and HTTP-based communication protocols. Second, the application-level content delivery service provides file access processes, such as browse and download actions, and the information necessary for asynchronous data delivery. This service performs in a way analogous to the remote procedure call (RPC) methodology that frequently appears in wide, distributed network systems. Third, a secure access control mechanism with double key identification employs the concept of secure transaction to guarantee access authentication among the home server, mobile content provider, and receiver. With the provider key and transaction key, the signatures applied to any transaction-based access process simply identify their owners and maintain trustworthiness in a short transaction duration. Finally, the mobile content provider and its home server provide virtual file systems with dynamic reference mapping to avoid disclosing the internal file system structure, thus guarding against security threats. Therefore, this architecture design achieves a secure and ubiquitous content delivery platform.

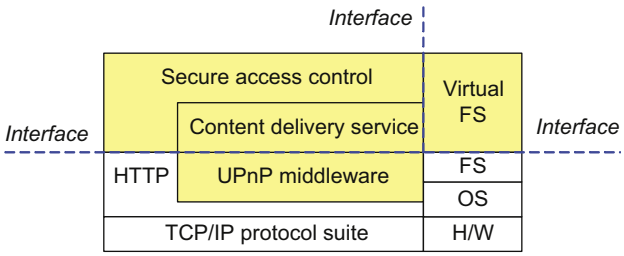

Fig. 3. The proposed software architecture and functional components.

3.2. Design considerations

The proposed platform design addresses several considerations against weaknesses in wireless and mobile network environments, particularly from the aspects of transmission throughput, transmission range, energy consumption, mobility, disconnection, and security.

- Since modern MHDs usually have at least dual or triple network connectivity capabilities, it is beneficial to leverage concurrent networking capabilities. The network with higher transmission throughput is preferred to reduce the transmission duration. A short duration, in turn, leads to lower energy consumption.
- When two MHDs coexist in multiple networks, the one with a shorter transmission distance is preferred to reduce energy consumption. This is because long distance transmission consumes energy on an order of magnitude as distance increases.
- When an MHD connects to different networks, the one with more reliable connection is preferred to avoid retransmission.
- When an MHD moves freely in different networks, the one with a larger coverage area is preferred to mitigate the negative effects of terminal mobility and service mobility.
- When an MHD moves freely in different networks, the network with the higher tolerance of terminal mobility and disconnection is preferred to sustain service performance and resource utilization in both the MHD and network system.
- When an MHD communicates with other devices in a public network context, trustworthiness relationship against security

threats must be guaranteed even when the networked devices involved may move to other network domains.

This design should be responsive to dynamic changes in user requirements and network conditions. In practice, these considerations may conflict with each other in different situations. For example, consider two MHDs are in the same WLAN, and that one is transferring media objects to the other but is about to run out of battery or move to another 3G network. The architecture design should provide necessary functionality for the provider to redirect the receiver to download the remaining content from the home server before the MHD leaves the WLAN. In this case, the content delivery service continues, though it may take the receiver more time to download content from a far home server over a xDSL or 3G network with a low transmission rate.

To satisfy the various usage requirements of ubiquitous content delivery, the proposed design includes several mechanisms to support significant functions: network context awareness, device and service discovery, transaction-based content retrieval, virtual file system, dynamic location reference mapping, authorization and authentication, asynchronous content delivery, and one-to-many content delivery. The following subsections describe these mechanisms.

3.3. Device and service discovery

This mechanism uses the UPnP protocol stack to develop the device discovery protocol. The UPnP technology specifies six function layers, as Section 2.2 describes. The proposed design leverages the network-addressing, discovery, description, and control function layers in partial, but excludes the eventing and presentation layers. In this design, the UPnP functionality of a mobile content provider is symmetric to that of a mobile content receiver. This is because both parties can have similar hardware and system conditions, and can act as the provider or receiver.

UPnP addressing offers the underlying function through which a device obtains a unique IP address upon joining a network. The DHCP, in most cases, is a function of the network router, which assigns IP addresses to network hosts in local networks. When UPnP devices power on, they check for the existence of DHCP server. If no server is present, they use automatic IP assignment to configure dynamic assignment of IPv4 link-local addresses in the 169.254/16 range. When network hosts have IP addresses in the same network domain, they can discover each other.

UPnP discovery is based on the SSDP specification (Internet-Draft, 1999). The SSDP is a simple HTTP-based discovery mechanism that discovers local resources in a small, local area network with no need for a centralized configuration, management, or administration. A UPnP device periodically advertises its appearance on a well-known address/port, 239.255.255.250:1900 (SSDP:NOTIFY), i.e., an HTTP multicast over UDP. Every active device in the network is then aware of its presence. Each device can directly query the network (SSDP: SEARCH), and each resource host can directly respond to the request (HTTP/OK Response). The HTTP LOCATION headers in these advertisement and response messages specify the URLs to the same description of the UPnP root device. A UPnP device uses a description to present its services and capabilities in a well-defined XML format. Interested UPnP devices can fetch and parse this description, learning what services the device offers and its profile information.

The UPnP control mechanism is based on the SOAP specification (W3C Consortium, 2000). The SOAP specification is an application-level communication protocol over HTTP that integrates both HTTP and XML conventions to provide a Web-based messaging and remote controlling mechanism. The proposed design preserves this function layer because the UPnP Forum has standardized several device control profiles to make consensus on the activities of different device categories (UPnP Forum Standardization Device Control Protocols, 2010). For example, UPnP AV profile is applied to instantiate any CE device that streams AV content to other UPnP devices. To support these profiles, the MHDs can extend their application scope to the dimension of home automation and entertainment. However, these profiles do not include *mobile content delivery services*.[7] Instead, the proposed architecture adopts a "customized control profile" that can satisfy the requirements of content browsing and meta-data retrieval.

The UPnP eventing and control layers are not appropriate for the development of media content delivery in mobile and ubiquitous network environments. Eventing based on the GENA performs well in a publisher–subscriber method (Internet-Draft, 2000). However, this may be useless for MHDs because they do not have static network addresses, particularly in wireless and mobile network systems. As for the control layer, the SOAP messaging over HTTP requires reliable network connections that are only attainable within a LAN/WLAN or small area network. Since the SOAP message format is complicated, parsing and resolving SOAP messages can induce higher computation overhead, as Sections 2.2 and 2.4 indicate. In fact, this approach is better for Web services than remote network services in unreliable wireless and mobile network environments.

Though the proposed architecture excludes the UPnP eventing layer, it employs the UPnP control layer in both mobile content provider and receivers. This allows both parties to connect in a small network domain, but not at the *far* home server. Sections 3.5.2 and 4 show that the home server interacts with remote devices across the wide area networks using an in-band XML-RPC messaging protocol. This RPC-like content delivery service can be deployed in mobile content providers without conflicting with SOAP-based control. Further, this protocol may replace the UPnP-specific control and data transfer functions.

3.4. Asynchronous content delivery

Figure 4 depicts the concept model of mobile and ubiquitous content delivery in the proposed architecture. This architecture provides three content delivery methods, *direct* downloading, *redirect* downloading, and *continual* downloading, to provide complementary usages. In addition, a *content directory service* makes the content presentation friendly to mobile content receivers.

3.4.1. Content directory service

The content directory service allows mobile users a retrieval interface for the media content that a virtual file system contains. This service checks all shared media items and summarizes their information into meta-data. Note that the UPnP and its AV profile does not standardize the representations of the object identifier and parent identifier of each media item. This study customizes this service to meet the requirements of media content management. Consider a native file system, provided by the mobile handheld platform, that is usually too simple and insecure to satisfy the demands of developers and users. This study develops a virtual file system with friendly and flexible UI structures that allow end users to perform media content browsing and other operations. The content directory service catalogs media items into image, audio, video, or document directories according to their formats rather than mixing them in a common directory. To avoid possible security threats, the location reference of a media

[7] Since this study considers mobile and ubiquitous IP communication networks, the terms "mobile content delivery" and "ubiquitous content delivery" may be used interchangeably hereafter.

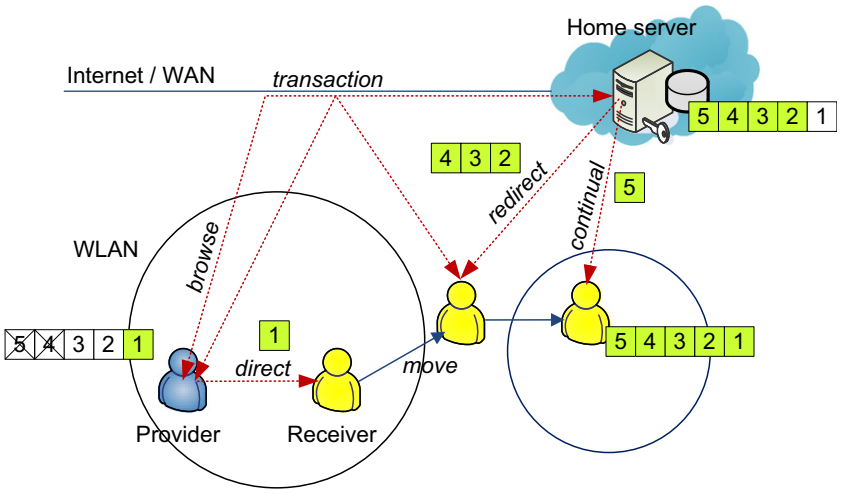

Fig. 4. Concept of mobile and ubiquitous content delivery.

item is assigned a "garbled" string. The content provider maps the factitious string to the real one in a native file system. To ease exposition, the following presents the meta-data structure of each media object in the proposed design.

```
< ItemList >
  < Item Type="#{TypeName}"    /* directory or file */
    Name="#{FileName}"
    Size="#{SizeInBytes}"
    URL="#{DownloadURLReference} "/ >
< /ItemList >
```

3.4.2. Case 1: direct downloading

The mobile content provider and receiver process the basic UPnP logic to discover each other. The receiver obtains the device and service description files of the mobile content provider. With the customized control profile, the receiver can *browse* the content directory of the mobile content provider and obtain a list of meta-data records for "shared" media objects that the mobile content provider intends to offer. Thus, the mobile content receiver acts an `HTTP GET` to the URL reference from which an indicated media object is downloaded. Besides, the receiver may download the content in an XML-RPC alternative if both of the provider and receiver support this option.

3.4.3. Case 2: redirect downloading

A redirect downloading method sustains the ongoing content delivery in response to several design considerations, as Section 3.2 describes. A mobile content provider instructs the receiver to access media content from its home server when the transfer of all indicated media objects is incomplete. The example in Fig. 4 shows that the provider has five media objects, 1–5, on its home server, but only keeps objects 1–3 in local storage due to limited capacity, object deletion, or other reasons. The provider queries its home server about a meta-data list that contains a number of location references for media objects stored there. By comparing the local list to the receiving list, the provider forwards the receiver a "working list" of indicated media objects that it redirects the receiver to download from the home server. This working list contains some or all of local items as long as the home server stores their copies. Figure 4 shows that only object 1 is delivered by the provider, though objects 2 and 3 are locally available. The receiver can download objects 2–5 from the home server through another network, unlike the one used to interconnect the provider. The use of a redirect downloading method depends on the considerations of resource utilization, network condition, system performance, or user scenario factors such as data throughput, transmission range, energy saving, mobility, and disconnection.

3.4.4. Case 3: continual downloading

The proposed design provides a continual downloading method in support of mobile content delivery after mobility or connection re-establishment by either the mobile content provider or receiver, if *transaction* deadline has not passed, or that "downloading intermission" is not longer that a specific interval threshold. Though the mobile content receiver keeps the working list that was assigned by the preceding redirect downloading process, it can still connect to download the indicated media contents until the home server invalidates those location references. The redirect downloading and continual downloading procedures are subject to a secure transaction scheme, as the following subsection indicates.

The home server is often deployed in an application gateway platform that runs on a PC or dedicated device with a large storage volume and computation power. From the viewpoint of mobile content delivery, a mobile content provider with limited resources can resort to the home server. The providers takes advantage of storage capacity and content transformation ability of the home server, like transcoding, transrating, or scaling content binary, to distribute multimedia content efficiently. This approach can significantly improve content delivery services and resource utilization among networked devices. Though the design of a home gateway device is orthogonal to this study, it is complementary to the prospect of mobile content delivery scenarios.

3.5. Secure access control

The proposed secure mobile content access and control management scheme is based on the concept of transaction process among the mobile content provider, receiver, and home server. Figure 5 shows the conceptual procedure. Double key identification with a pair of provider key and transaction key guarantees mutual trustworthiness. The following subsections describe the transaction process and the double key identification scheme.

3.5.1. Transaction process

This mechanism imposes a secure transaction process with authorization and authentication during mobile content delivery. When the provider commences a redirect downloading process,

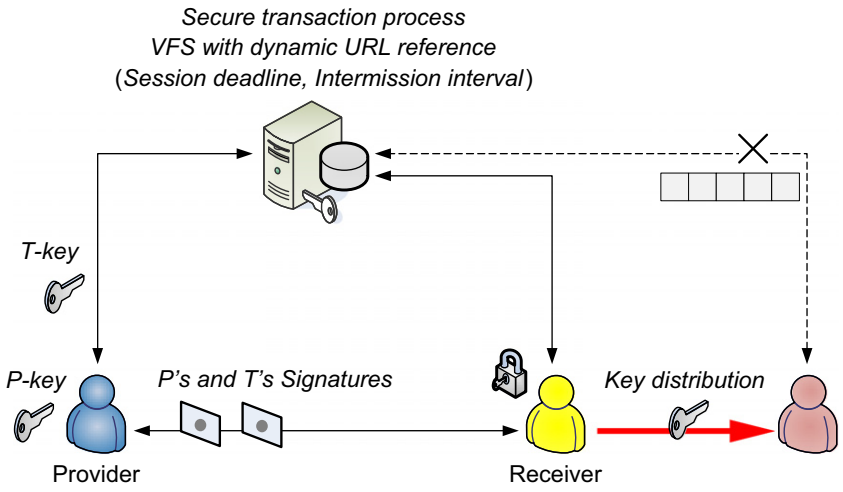

Fig. 5. Secure transaction-based media content access and control management.

it initializes a secure transaction process. The home server manages the transaction process and statuses in a centralized fashion since it has become the content provider in place of the mobile content provider. Neither the mobile content provider nor receiver incurs management expenditure. Each transaction process is assigned a transaction or *session* deadline. The "session" term means that the receiver can asynchronously perform multiple downloads to complete the working list before the transaction deadline or termination. Moreover, any downloading intermission before the session deadline must not be longer than a specific interval threshold. These two temporal limitations, i.e., session deadline and intermission interval, can alleviate possible drains on the network and system resources at the home server.

A secure transaction begins when the provider informs both its home server and the receiver, and continues until the last media item is downloaded completely. An exception is when any of the following situations occurs: any participant aborts the transaction process, the associated session deadline expires, or the intermission between two successive downloads exceeds the serving interval. The proposed design opts not to subsume direct downloading in the transaction process. Because the mobile content provider directly interacts with the neighboring receiver in a small vicinity, its owner is able to oversee and control the mobile content delivery.

3.5.2. Double key identification

The home server must be able to ascertain whether a mobile content receiver is trustworthy or not. To achieve this goal, the proposed design uses a simple double key identification scheme with a pair of symmetric keys, i.e., provider key and transaction key. These two keys facilitate the identification of trusted mobile content providers and receivers, respectively.

- The *provider key* (P-Key) is pre-determined and kept secure by the mobile content provider. This key determines the trustworthiness between a mobile content provider and its home server. The provider key may be refreshed to avoid being cracked after a mobile content provider has completed a transaction.
- The *transaction key* (T-Key) is dynamically generated by the home server whenever it initializes a new transaction. This key is used in the transaction between the mobile content receiver and home server. It is transient and effective to the end of the session deadline.

  For every redirected download, the mobile content provider asks its home server for a transaction key which is opaque to the mobile content receiver and is invalidated immediately after the transaction is complete. This enforcement prevents any parties from distributing a transaction key to unauthorized receivers.

The following subsection describes the double key identification scheme and its usage in a secure transaction process. To ease comprehension, Figs. 5 and 6 show the flowchart of this interaction procedure.

The proposed mechanism adopts an RPC-like and XML-based messaging model, abbreviated as XML-RPC, to perform the secure transaction process between the home server, and mobile content provider or receiver. Two field entities, *identifier* and *signature*, facilitate the resolution of authentication on XML-RPC sender and receiver. They are associated with the URL reference, indicated by an XML-RPC invocation. Precisely, each XML-RPC invocation expression contains the sender's identifier, which the XML-RPC receiver uses to determines the trustworthiness of an XML-RPC sender, or if an XML-RPC sender is authorized to execute the indicated procedure. This expression may include the sender's signature to convince the receiver that the critical data enclosed is intact. An XML-RPC receiver can determine the validity of the received signature. The canonical forms of identifier and signature entities in the URL expression are as follows:

$$\text{Identifier} \coloneqq \#\{\text{P}-\text{Key}\}|(\#\{\text{P}-\text{Key}\}\&\#\{\text{T}-\text{Key}\})$$
$$\text{Signature} \coloneqq \text{MD5}(\#\{\text{DigestURL\_Raw}\})$$

An identifier is added to the DigestURL_Raw expression that is further hashed by the MD5 function to generate a signature. The P-Key and T-Key values are unique during a transaction process, and so is the resulting signature. Accordingly, an XML-RPC receiver can believe an XML-RPC sender to be trustworthy.

A DigestURL_Raw element uses three different compositions, DigestURL_Raw_0, _1, and _2, as the inputs of the MD5 function for signature generation in support of different sorts of XML-RPC invocations. The first two elements are included in the XML-RPC invocation expression sent from the provider to the home server. The last element is enclosed in the XML-RPC invocation expression sent from the receiver, through the provider, to the home server. The provider takes charge of subscribing

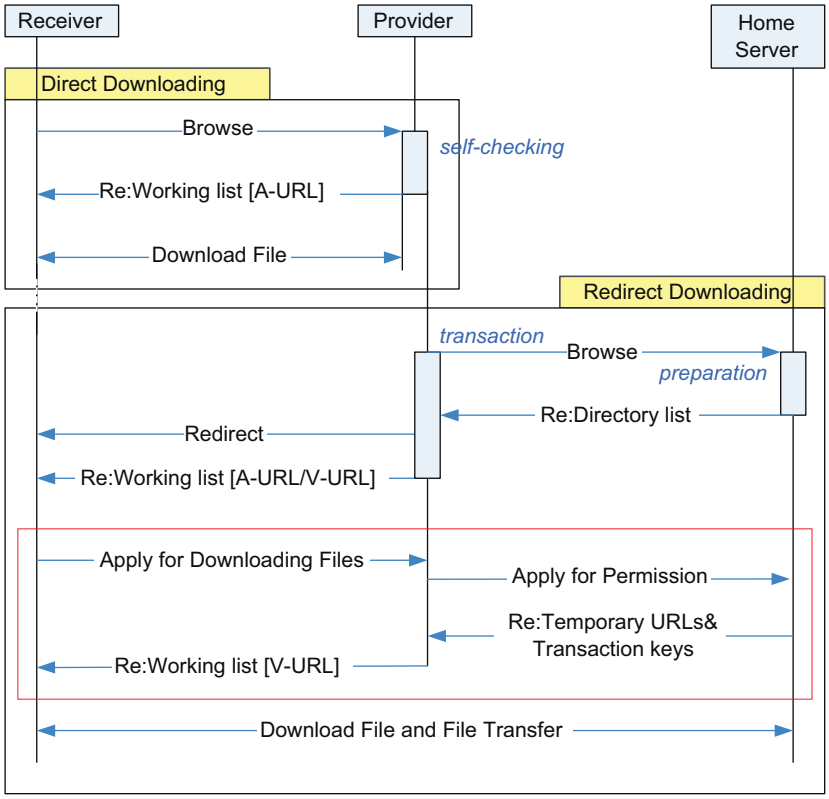

Fig. 6. Interactive flowcharts of direct and redirect downloading cases.

the corresponding MD5(DigestURL_Raw_(0|1|2)) at the end of the URL expression.

URL ≔ "HostURLBase/XML-RPCAction?
File=#{FileName}&
User=#{UserName}&
Signature=#{Signature}"
DigestURL_Raw_0 ≔ "/XML-RPCAction?
User=#{ProviderName}&
Key=#{P-Key}"
DigestURL_Raw_1 ≔"/XML-RPCAction?
File=#{FileName}&
User=#{ReceiverName}&
Key=#{P-Key}"
DigestURL_Raw_2 ≔"/XML-RPCAction?
File=#{FileName}&
User=#{ReceiverName}&
Key1=#{P-Key}&
Key2=#{T-Key}"

The URL refers to the host that runs the indicated XML-RPC action routine upon the target file objects. The HostURLBase element designates the host's location reference, on a home server or mobile content provider, depending on the downloading method chosen. In the case of a redirect or continual download, the HostURLBase value points to the home server. Otherwise, it refers to the mobile content provider itself.

Table 1 lists an initial set of XML-RPC actions for the secure mobile content delivery, as mentioned in the next section. Depending on which XML-RPC action is chosen, the mobile content provider must subscribe its appropriate signature using DigestURL_-Raw_0, _1, or _2. The major difference between DigestURL_Raw_0 and DigestURL_Raw_1 is the value of the User field, which indicates where an XML-RPC action originates from. A value of Key=#{P-Key} in DigestURL_Raw_1 means that the provider performed this action. The pair of P-Key and T-Key in DigestURL_Raw_2 means that both the provider and the home server committed themselves to this action for the receiver. Continual downloading is a good example of this case: the provider asks the home server in advance to grant the receiver's downloading request. However, the home server retains the right to deny any XML-RPC action.

The double-key identification scheme utilizes the signature technique to verify the XML-RPC action requests instead of encrypting those requests. The use of a signature is enough to protect the originality and integrity of the XML-RPC action requests. The home server knows the provider and the receiver names, supports a list of XML-RPC actions, and secretly stores the P-Key and T-Key. Thus, the home server can generate a signature and compare this with the received one. If the two signatures are not equivalent, the home server rejects the XML-RPC action request. The present prototype adopts the basic MD5 hash function, but other, more robust hashing candidates may be adopted in future designs.

The proposed scheme does not adopt sophisticated algorithms to generate a symmetric key or shared secret key. Instead, it uses symmetric-key algorithms which are generally much less computationally intensive than asymmetric key algorithms. One disadvantage of symmetric-key management is the requirement of a shared secret key, with one copy at each end. However, the effect of a potential discovery by a cryptographic adversary is minimal because there are only three participants in a transaction process. Next, a transaction key is transient, and different transactions are assigned different keys. Moreover, the home server can regularly change the provider key, ensuring security during distribution and service. If necessary, asymmetric algorithms can be used to distribute symmetric-keys at the start of a session. This simplifies the key distribution problem, as asymmetric keys only have to be distributed authentically, whereas symmetric keys must be distributed in an authentic and confidential manner.

Table 1
An initial set of XML-RPC actions.

XML-RPC action name	Description and usage
Browse	browse the content directory service provided on the indicated hostURLBase site /Browse?User = {(Provider|Receiver)}
Search	search any specific media object on the indicated site /Search?User = {(Provider|Receiver)}
Redirect	redirect the receiver to change the downloading method /Redirect?User = {Provider}&Signature = {MD5(DigestURL_Raw_0)}
ApplyForPermission	apply to home server for downloading permission /ApplyForPermission?File = {FileName}&User = {Receiver}&Signature = {MD5(DigestURL_Raw_1)}
ApplyForDownload	apply to provider for redirectly downloading a file /ApplyForDownload?File = {FileName}&User = {Receiver}
ApplyForBatchDownload	apply to provider for redirectly downloading files ApplyForDownload?File = {(FileName)$^{+}$}&User = {Receiver}
DownloadFile	download files via HTTP GET in redirect or continual mode /Download?File = {FileName}&User = {Receiver}&Signature = {MD5(DigestURL_Raw_2)}

3.6. Virtual file system and dynamic reference mapping

The home server employs a virtual file system with dynamic reference mapping to process directory access, file browsing, and retrieval operations through safe and secure content delivery services.

The proposed virtual file system framework uses a *directory-based* mapping approach. This virtual file system need not wholly imitate the native file system. Instead, a directory tree structure is enough to present the entities of all directories, sub-directories, and files in service (UPnP Forum, 2006). This system uses the analogous meta-data structure described in Section 3.4 with extended elements for file management. Every file or directory in the directory tree has a meta-data record. The home server marshals the meta-data records corresponding to file objects stored within a specific directory tree. The detailed structure of this directory tree (UPnP Forum, 2006) is omitted here to conserve space.

There are two variants of the URL reference element, actual and virtual URLs, depending on the design of the proposed asynchronous content delivery mechanism.

- The *actual URL* refers to the logic path in the underlying software system, e.g., URL-1 ≔ /MyHostURLBase/AV_Dir/Picture/My.jpeg. skip
- The *virtual URL* refers to a temporary, symbolic location reference, e.g., URL-2 ≔ /MyHostURLBase/MyPicXXX.jpeg, that the reference mapping service dynamically generates. The virtual file system resolves the symbolic mapping between virtual and actual URLs when it obtains a virtual URL.

This mechanism performs a two-phase access control on the asynchronous content delivery. As Fig. 6 illustrates, the mobile content provider provides the receiver with direct downloading during the initial phase. The provider lists the actual URLs in the working list. In redirect or continual downloading phase, the provider requests the home server to prepare a new working list whose meta-data records include virtual URLs. The provider then sends a newly modified working list to the receiver, which can then download the desired indicated content from the home server.

It is important to control the location reference. This is because the virtual file system does not have a pre-determined reference pool to avoid the potential security leak of the reference re-entry problem. Instead, every reference is generated dynamically and formed in a garbled string. The home server creates a location reference upon receiving a "permission request" for redirect downloading from the mobile content provider. The reference is sent to the provider and then forwarded to the receiver, which makes a connection to fetch the indicated file. The location reference is kept in a working list pertaining to the transaction. Every transaction is subject to transaction timing constraints. All location references associated with a transaction process are invalidated when the transaction is finished or terminated. This is known as *transaction-based validity*.

With its virtual file system and dynamic reference mapping designs, the proposed architecture provides a robust, secure, and friendly content directory service. The virtual file system provides a structured representation to ease content browsing and rendering. This allows the content directory service to catalog various media items, for instance, into audio, image, video, or document directories, in a directory tree. In addition, the dynamic mapping and transaction-based reference management scheme not only keeps the system secure from possible security threats, but places no additional burden on the receiver during `HTTP GET` downloading.

4. Prototype development and demonstration

This section describes the development and implementation of an experimental prototype. To ease exposition, this section presents two examples, involving direct and redirect downloadings, to demonstrate secure mobile content delivery services in the proposed architecture.

4.1. Development environment

The prototype development environment includes four architecture components and the XML-RPC messaging protocol. The UPnP stack was implemented in Java programming language. This allowed developers to avoid solving any porting and dependency issues on experimental platforms, including Linux, Windows XP, and Windows Mobile platforms, as they all support Java Virtual Machine (JVM). The implementation of other mechanism software and XML-RPC communication modules was based on Ruby 1.8.6 on a Rails 2.2 Web Framework. All XML-RPC interaction processes between the mobile content provider and its home server were developed using Web communication technologies. All interactive messages were represented in XML conventions. In addition, a mobile content receiver only required a little customization or program installed to interact with the home server over the HTTP communication protocols, depending on the specific conventions. Thus, the XML-RPC ran as a Web service. This allows networked devices with TCP/IP connectivity, Web server, or client, and XML functions to access the mobile content delivery service in the proposed framework.

4.2. XML-RPC primitives

The experimental prototype runs in XML-RPC logics when end devices communicate with each other across mobile and ubiquitous IP network environments. Specifically, a mobile content provider can send XML-RPC control actions to a mobile content receiver and later get response messages, and vice versa. A mobile content provider can also communicate with its home server outside a UPnP-based network domain. Considering the interactive behavior, this study defines an initial set of XML-RPC

operational primitives to control messaging among mobile content providers, receivers, and home servers, and to establish a secure mobile content delivery on the target device platforms. Table 1 lists the basic XML-RPC actions in this prototype. All actions have a uniform *in-line* expression as URL := "HostURLBase/XML-RPCAction?Parameters."

4.3. Interaction procedure

To clarify the prototype demonstration and usage of the XML-RPC actions in Table 1, Fig. 6 shows two interactive procedures for direct and redirect downloading cases. In the first phase, the mobile content provider and receiver execute the UPnP discovery to find each other and prepare for mobile content delivery. Refer to Section 3.3 for a description of this UPnP phase. After the provider and receiver are engaged, the receiver browses the content directory and downloads the desired media objects. In case of direct downloading, the receiver invokes a `Browse` action on the provider and receives the response of a working list of meta-data records, arranged according to the content directory service with actual URLs. The receiver applies `HTTP GET` iteratively to download separate media objects from the provider.

Browse:
< ItemList >
< Item Type="File" Name="Pic1.jpeg" Size="69200"
URL="/140.115.152.2/AV_Dir/Picture/MyPic1.jpeg"/ >
< Item Type="File" Name="Track1.mp3"
Size="3155202"
URL="/140.115.152.2/AV_Dir/Music/Track1.mp3"/ >
</ ItemList >
Download File:
HTTP GET /AV_Dir/Music/Track1.mp3 HTTP/1.1
HOST: 140.115.152.2:50988

For redirect downloading, the provider initially invokes a `Browse` action on the home server and receives a *directory list* of meta-data records in response. This directory list can be a super-set of all available media objects on the home server. The provider filters out irrelevant items and adds items that are not stored in its local storage. The provider then performs a `Redirect` action to notify the receiver to prepare the following redirect downloading *transaction*, and then forwards it a newly working list of indicated meta-data records. Every item in the new list is the same as the above, except for the value of the URL element that distinguishes the home server from the provider. The following gives an example: "Home_Dir" is dummy, not a real directory on the home server. The provider simply uses it to differentiate the sources from which an indicated media object should be downloaded from.

URL="/HomeServerURLBase/Home_Dir/Track1.mp3"

The receiver then invokes an `ApplyForDownload` or `ApplyForBatchDownload` action before retrieving any files from the home server. The receiver informs the provider of a list of indicated media objects it wants to access from the home server. After obtaining this list, the provider applies its signature `MD5(DigestURL_Raw_1)` to every URL and sends an `ApplyForPermission` request to the home server. If the provider's signature is authentic, the home server generates a transaction key and initializes a secure transaction process to reply to this request. The home server also assigns a virtual URL to every URL using dynamic reference mapping. The following is an example of a temporary URL.

URL="/140.115.152.4/XYZ123"

Note that a virtual file system uses a temporary location reference and file name to replace the previous one named by the provider. The transaction key and a list of temporary URLs are then sent to the provider. The provider modifies each temporary URL by appending a joint signature `MD5(DigestURL_Raw_2)` to the URL. The provider then sends the receiver a final working list in response to the `ApplyForDownload` action. This allows the receiver to perform `DownloadFile` actions to asynchronously retrieve indicated media objects before the transaction expires.

5. Performance results

This section evaluates the performance of the proposed user-provided multimedia content distribution architecture. From the viewpoint of system ability and agility, this study reports several experiments designed to determine the relative performance in terms of client request-response time, server processing time, and transaction time, compared with the conventional SOAP approach.

5.1. Experimental environments

The experiments on the developed prototype were conducted using off-the-shelf MHDs and PCs in real network environments. Mobile content receivers and providers ran in Apple iPhones (Samsung S3C6400 533 MHz CPU, Infineon UMTS/HSDPA, Wi-Fi 802.11b/g, and iPhone OS 2.2.1 version). The home server was installed on a general PC (Intel Core 2 Quad 2.66 GHz with 2 GB.RAM, and ArchLinux). The XML-RPC software module was implemented using Ruby 1.8.6 on a Rails 2.2 Web Framework. The mobile content receiver, provider, and home server supported XML-RPC functionality. The mobile content provider and home server ran a micro-Web server to process XML-RPC actions issued by any MHD through a simple Web client. The experiments adopted a control design with a SOAP 4R library using the same Ruby framework as a comparison baseline. Figure 7 shows the relative performance of the XML-RPC and SOAP models in two network environments, Wi-Fi 802.11b/g and 3G.

Figure 8 depicts the procedure of media file sharing that corresponds to a secure transaction over experimental network contexts. The measure includes three sequential phases: (1) browse, (2) ApplyForDownload and ApplyForPermission, and (3) DownloadFile phases. This figure ignores the preceding discovery phase because a mobile receiver finds a neighboring provider via UPnP discovery within its WLAN in both network environments. The time duration to transfer all data in a media file is not considered after an ordinary `HTTP GET` method is sent. Thus, the transaction time

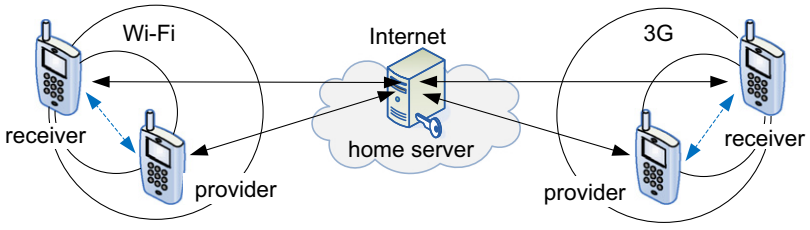

Fig. 7. Experimental environments.

indicates the aggregate of time intervals in these three phases. The following reports the average time to finish each phase and a transaction after one hundred testing runs.

5.2. Experiments in Wi-Fi and 3G networks

Given with the same experimental case of downloading a single file, this subsection examines the performance results under Wi-Fi and 3G networks. Figure 9 depicts the transaction times in contrast with those after using SOAP in lieu of XML-RPC. Figure 10 depicts the server processing times of phases 1, 2, and 3 in a transaction.

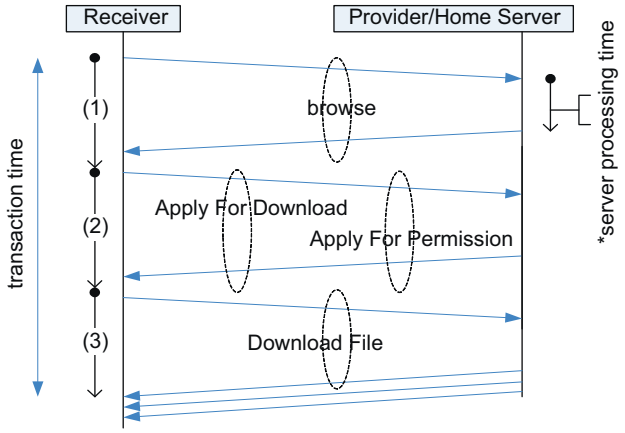

Fig. 8. Performance measure in three phases.

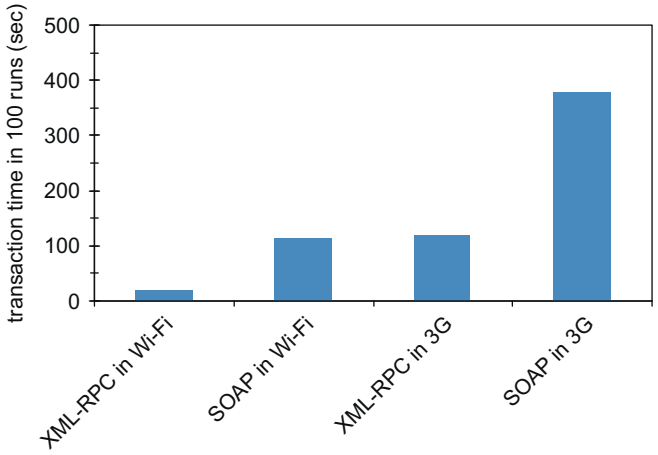

Fig. 9. Transaction time: an aggregate in 100 runs.

Results show that XML-RPC is much faster than SOAP in all cases in both networks. SOAP's transaction time is about six times that of XML-RPC under Wi-Fi. This falls to about three times when both SOAP and XML-RPC transfer data via longer connection distances in 3G. Nevertheless, the time difference between SOAP and XML-RPC remains significant. This is due to lengthy SOAP message payload, complicated message syntax, and extra messaging overhead, as described below.

The design of the XML-RPC message format and representation is simple. Compared with SOAP, XML-RPC is easy to parse and resolve. Two types of experimental results support this observation. First, parsing and resolving a SOAP message takes a lot of time due to its well-structured message format, strong type checking, and syntactical representation. Correspondingly, the design of a SOAP parser is quite complex. The results of phase 1 in Fig. 10 confirm this statement. Though transmission speed in the Wi-Fi context is much higher than that of 3G, SOAP's processing time in phase 1 is still long. This is because that both the receiver and server must handle SOAP's complicated message bodies and XML extensions. The second evidence is a comparison between XML-RPC and SOAP in terms of phases 2 and 3 processing times. The XML-RPC action specifications in Table 1 indicate that the processing times of XML-RPC's actions in phases 2 and 3 are related to the round-trip delay time, and incur no additional message payload. SOAP's processing times in these two phases are much longer than XML-RPC's, indicating that SOAP's messaging overhead is responsible for this time difference.

The percentage comparisons in Table 2 and Fig. 9 show that the XML-RPC messaging communication protocol is simplified and handled in a straightforward way with a fast response time. In the case of XML-RPC in 3G, all three phases have very short processing time, with an average of one third of the usual transaction time. Phase 1 achieves a round-trip time as short as phases 2 and 3 do even though phase 1 involves content directory services, i.e., meta-data

Table 2
Percentage of transaction time.

Method	Phase 1	Phase 2	Phase 3
XML-RPC in 3G	34.88	30.26	34.86
XML-RPC in Wi-Fi	46.58	24.17	29.25
SOAP in 3G	45.47	27.56	26.97
SOAP in Wi-Fi	65.53	17.40	17.07

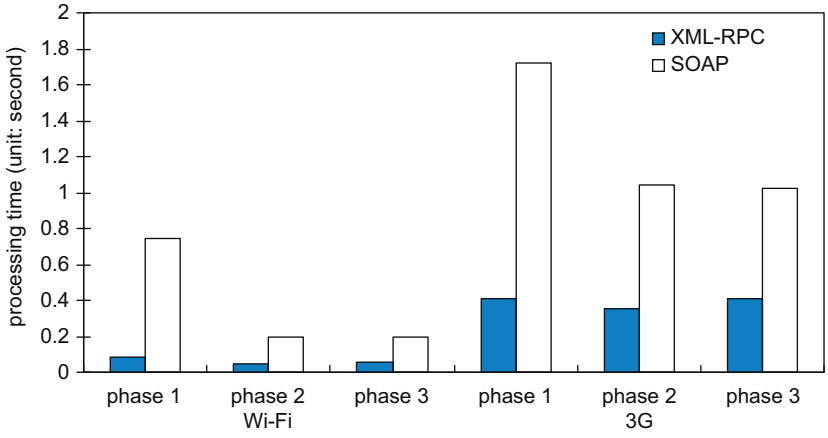

Fig. 10. Processing time: three phases in Wi-Fi and 3G network contexts.

composition and itemization. This indicates the effect of XML-RPC's *in-line* data representation without extra messaging overhead. Contrarily, the processing time of each phase in SOAP increases drastically. Due to SOAP's conventions, all phases take much more time to transmit lengthy SOAP messages. For example, phase 1 requires two-third the transaction time in a Wi-Fi environment. This phase unfortunately induces more computation cost than XML-RPC.

Energy drain is critical to mobile applications because mobile devices are often battery-powered. Thus, a key factor in running applications on mobile devices is to improve the energy utilization. A longer transaction time implies more energy consumption. As Fig. 9 shows, XML-RPC's average transaction time is about one third that of SOAP. In particular, Fig. 10 shows that XML-RPC achieves lower processing time than SOAP does in every phase. For instance, in phase 2 of requesting for downloading a single file, because the message body merely refers to a single item, the majority of the processing time involves message exchanging between the receiver and the provider/home server over the connection. Even so, SOAP's processing time remains about three to four times that of XML-RPC. Therefore, it is better to use XML-RPC because it significantly decreases the transmission duration.

5.3. Experiments on scalability against request workload

This subsection investigates server scalability against request workload in terms of server processing time. The server processing time indicates the time interval from the moment at which a server fetches a pending request from the input queue until it sends out its response. Consider that phase 1 takes a major percent of transaction time in content directory services. This study conducts experiments to measure the processing time of browsing a single item as a performance base.

The experimental setting contains a lighttpd HTTP Web server, which communicates with the back-end SOAP and XML-RPC handlers via FastCGI package and APIs (Lighttpd Web Server, 2009). To calculate server processing time accurately, only one FastCGI back-end is activated to avoid the multi-process concurrency issue. Then, experiments were run iteratively with incremental request workload. Figure 11 shows that both SOAP and XML-RPC approaches encountered a server bottleneck when request workload was about 130 requests per second. This is because the underlying target system with definite kernel resource, network I/O, and processor abilities was unable to deal with infinite request workload.

Handling an XML-RPC request is much faster than handling an SOAP request with a 135% increase in service processing time in the case of browsing a single file. The complicated structure, representation, and lengthy message payload in SOAP induce considerable processing time in parsing, resolving, and encapsulating SOAP messages in a request-response manner. Depending on the content directory service, the time difference can be enlarged, however, when downloading multiple files. Server scalability deteriorates as the request workload increases. Therefore, XML-RPC is able to expedite the server's processing, compared with SOAP.

5.4. Remarks and discussions

Unlike network applications running on desktops in fixed and broadband Internet environments, mobile applications suffer many restrictions, such as limited battery capacity, CPU capacity, memory space, Internet access speed, etc. Transmission speed is often the determinant factor in the success of a mobile transport service. However, this section addresses how application-level messaging representation and its communication protocol influence system performance to a considerable extent. The following accordingly summarizes several observations and remarks.

First, XML-RPC outperforms the SOAP in all phases during a transaction. Processing SOAP messages is expensive because it occupies much computational time and memory resource, especially for resource-limited mobile handheld devices. In contrast, XML-RPC features concise message format and composition, and rapid processing time in each phase. Thus, XML-RPC messaging can not only release strict energy and resource utilization, but also lower processing time to an acceptable level.

Second, the associated secure transaction model is lightweight, without any extra interactive messages among participant devices. The in-band and in-line authentication and access control mechanism simplifies the procedure of controlling security and safety for file sharing at the home server, e.g., redirect and continual downloading cases. In contrast, SOAP does not have in-band access control yet. Any out-of-band supplement may somehow burden mobile devices.

Third, during a secure transaction, a mobile content receiver can escape from negotiation and message exchange about authentication and access control. Rather, it appeals to its provider to assign an "opaque" signature, recognized by the provider and its home server. This specific design lets a thin receiver, with limited resource, be free from extra communication cost.

Fourth, XML-RPC is superior to SOAP from the viewpoint of service provision. The importance of scalability is noteworthy, since server loading is high against request workload. Correspondingly, a fast processing time in a request-response interaction can not only offer better user experience, but also shorten the network binding

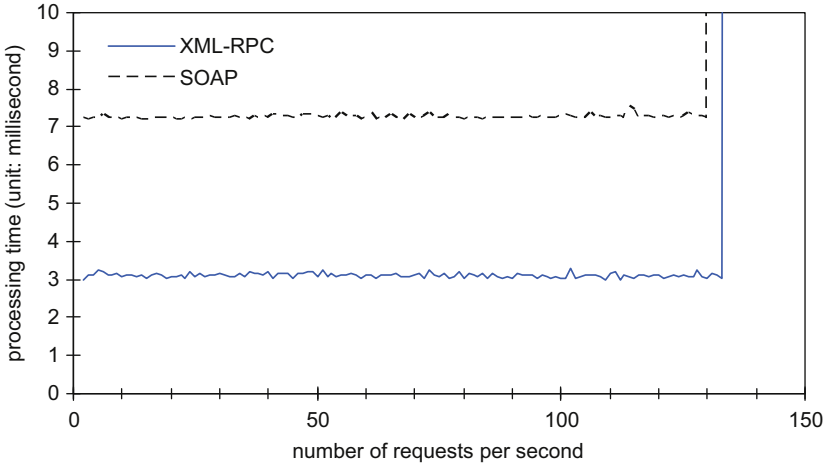

Fig. 11. Server scalability versus request workload.

time at a mobile receiver side to reduce the energy consumption.

Fifth, message representation and messaging protocols have a significant influence on system performance, especially to mobile applications. Though transmission speed is often concerned in wireless and mobile network contexts, it can be the case in some long-term applications, like large file delivery and streaming services; however, it may not be in many applications, for example, document-based information, instance messaging, Web surfing, and picture and music exchange services. The performance examination in this section addresses this implicit point: in fact, the process of parsing and resolving messages accounts for a significant part of transaction time, even more than transmission time.

The results above indicate that the XML-RPC mechanism is functional. Its design is secure, lightweight, and easy to implement with less system resource cost, energy consumption, message traffic, and overhead than conventional SOAP systems. Therefore, the proposed architecture is suitable for supporting network applications in resource-limited mobile handheld devices and telecommunication networks with lower transmission speeds.

6. Conclusion and future work

This paper proposes a user-provided multimedia content distribution software architecture for mobile and ubiquitous IP communication networks. The proposed architecture integrates several mechanisms, including UPnP discovery middleware, asynchronous content delivery service, secure transaction-based access control and management, and virtual file system with dynamic reference mapping. This architecture also adopts a fast and lightweight XML-RPC communication protocol for control messaging between remote networked devices. This integrated design allows networked devices to connect with each other, and enables mobile content delivery anywhere in a ubiquitous network playground. In addition, this study presents a proof-of-concept prototype. The prototype demonstration and performance results in practical experiments confirm that the proposed architecture is feasible and satisfactory.

We are currently investigating several design issues and development requirements for the proposed architecture. Future research will focus on three aspects. First, we will extend the set of XML-RPC actions to support publish/subscribe services with event notification. The goal here is to develop a ubiquitous content synchronization service. Second, we will design service management, storage I/O, manipulation, and local environment primitives to meet the basic requirements for developing a mobile device control paradigm. Finally, we will develop additional security packages and technologies to strengthen the security and robustness of the proposed architecture.

Appendix A. Supplementary data

Supplementary data associated with this article can be found in the online version at doi:10.1016/j.jnca.2010.08.010.

References

Al-Ali A, Al-Rousan M. Java-based home automation system. IEEE Transactions on Consumer Electronics 2004;50(2):498–504.

Belimpasakis P, Awan SA, Berki E. Mobile content sharing utilizing the home infrastructure. In: Proceedings of the 2nd international conference on next generation mobile applications, services and technologies (NGMAST'08); September 2008. p. 155–60.

Bieber G, Carpenter J. Introduction to service-oriented programming. Online available via 〈http://www.openwings.org/download.html〉; September 2001.

Choi J, Shin D. Research and implementation of the context-aware middleware for controlling home appliances. IEEE Transactions on Consumer Electronics 2005;51(1):301–6.

Delphinanto A, Hillen BAG, Passchier I, van Schoonhoven BHA, den Hartog FTH. Remote discovery and management of end-user devices in heterogeneous private networks. In: Proceedings of the 6th IEEE consumer communications and networking conference (CCNC'09), vol. 6; January 2009. p. 848–52.

Digital Living Network Alliance, December 2004. DLNA home networked device interoperability guidelines version 1.1 (rev 0.19).

Digital Living Network Alliance, March 2006. DLNA home networked device interoperability guidelines version 1.5.

Edwards WK. Discovery systems in ubiquitous computing. IEEE Pervasive Computing 2006;5(2):70–7.

Fielding R. Architectural styles and the design of network-based software architectures. online available via 〈http://www.ics.uci.edu/fiedling/pubs/dissertation/top.htm〉; alternatively via 〈http://en.wikipedia.org/wiki/Representational_State_Transfer〉; 2000.

Garcia-Reinoso J, Vidal I, Valera F, Azcorra A. Zero config residential gateway experiences for next generation smart homes. Computer Networks 2009;53(18):2967–84.

Home Gateway Initiative (HGI). Home gateway technical requirements: residential profile v1.01. Online available via 〈http://www.homegatewayinitiative.org/〉; 2008.

Hu C-L, Huang Y-J, Liao W-S. Multicast complement for efficient UPnP eventing in home computing network. In: Proceedings of IEEE international conference on portable information devices (Portable'07); February 2007.

Hu C-L, Liao W-S, Huang Y-J. Mobile media content sharing in UPnP-based home network environment. Journal of Information Science and Engineering 2008;24(6):1753–69.

Internet-Draft. Simple service discovery protocol/1.0 operating without an arbiter. IETF Internet-Draft draft-cai-ssdp-v1-03.txt; October 1999.

Internet-Draft. General event notification architecture base: client to arbiter. IETF Internet-Draft draft-cohen-gena-p-base-01.txt; September 2000.

Internet-Draft. Dynamic configuration of IPv4 link-local addresses. IETF draft-ietf-zeroconf-ipv4-linklocal-17.txt; July 2004.

Kawamoto E, Kadowaki K, Koita T, Sato K. Content sharing among UPnP gateways on unstructured P2P network using dynamic overlay topology optimization. In: Proceedings of the 6th IEEE consumer communications and networking conference (CCNC'09); January 2009. p. 1273–77.

Kim D-S, Lee J-M, Kwon WH, Yuh IK. Design and implementation of home network systems using UPnP middleware for networked appliances. IEEE Transactions on Consumer Electronics 2002;48(4):963–72.

Kim J-T, Oh Y-J, Lee H-K, Paik E-H, Park K-R. Implementation of the DLNA proxy system for sharing home media contents. IEEE Transactions on Consumer Electronics 2007;53(1):139–44.

Kim J-T, Park S, Lee J-H, Paik E-H, Park K-R. Provision of the multimedia service framework in the ubiquitous home network. IEEE Transactions on Consumer Electronics 2008;54(2):501–6.

Kim T, Bahn H. Implementation of the storage manager for an IPTV set-top box. IEEE Transactions on Consumer Electronics 2008;54(4):1770–5.

Kuriyama H, Mineno H, Seno Y, Furumura T, Mizuno T. Home appliance translator for remote control of conventional home appliance. In: Proceedings of the 20th international conference on advanced information networking and applications (AINA'06); April 2006.

Lawrence V, Zervos N, Zahariadis T, Meliones A. Digital gateways for multimedia home networks. Telecommunication Systems 2003;23(3–4):335–49.

Lee H, Kim J. An approach for content sharing among UPnP devices in different home networks. IEEE Transactions on Consumer Electronics 2007;53(4):1419–26.

Lighttpd Web Server. Online available via 〈http://www.lighttpd.net/〉; 2009.

Liong Y, Ye Y. Effect of UPnP advertisements on user experience and power consumption. In: Proceedings of the 2nd IEEE consumer communications and networking conference (CCNC'05); January 2005. p. 91–7.

Mazuryk Y, Lukkien JJ. Analysis and improvements of the eventing protocol for universal plug and play. In: Proceedings of the IASTED conference on communications, internet and information technology; November 2004.

Mills K, Dabrowski C. Adaptive jitter control for UPnP m-search. In: Proceedings of the 38th annual IEEE international conference on communications (ICC'03), vol. 2; May 2003. p. 1008–13.

Moriya T, Ohnishi H, Yoshida M, Ogawa T, Hirano M. A support system for designing ubiquitous service composition scenarios. In: Proceedings of the 2007 IEEE international conference on communications (ICC'07); June 2007. p. 1594–9.

Motegi S, Tasaka K, Idoue A, Horiuchi H. Proposal on wide area dlna communication system. In: Proceedings of the 5th IEEE consumer communications and networking conference (CCNC'08); January 2008. p. 233–7.

Newmarch J. A RESTful approach: clean UPnP without SOAP. In: Proceedings of the 2nd IEEE consumer communications and networking conference (CCNC'05); January 2005. p. 134–138.

Oh Y-J, Lee H-K, Kim J-T, Paik E-H, Park K-R. Design of an extended architecture for sharing DLNA compliant home media from outside the home. IEEE Transactions on Consumer Electronics 2007;53(2):542–7.

Ohnishi H, Yamato Y, Kaneko M, Moriya T, Hirano M, Sunaga H. Service delivery platform for telecom-enterprise-internet combined services. In: Proceedings of the 2007 IEEE global telecommunications conference (GLOBECOM'07); November 2007. p. 108–12.

Open Service Gateway Initiative (OSGi) Alliance. OSGi service platform specification release 4. Online available via 〈http://www.osgi.org〉; October 2005.
Rich C, Sidner C, Lesh N, Garland A, Booth S. Collaborative help for networked home products. In: Proceedings of IEEE international conference on consumer electronics; January 2005. p. 209–10.
Saiedian H, Mulkey S. Performance evaluation of eventing web services in real-time applications. IEEE Communications Magazine 2008;46(3):106–11.
Song T, Kawahara Y, Asami T. Using SNS as access control mechanism for dlna content sharing system. In: Proceedings of the 6th IEEE consumer communications and networking conference (CCNC'09); January 2009. p. 1386–1387.
UPnP Forum. UPnP device architecture 1.0 version 1.0.1; December 2003.
UPnP Forum. UPnP AV contentdirectory:2 service template version 1.01; May 2006.
UPnP Forum Standardization Device Control Protocols. 〈http://www.upnp.org/standardizeddcps/〉; 2010.
Vanthournout K, Deconinck G, Belmans R. A taxonomy for resource discovery. Personal and Ubiquitous Computing 2005;9(2):81–9.
Venkitaraman N. Wide-area media sharing with UPnP/DLNA. In: Proceedings of the 5th IEEE consumer communications and networking conference (CCNC'08); January 2008. p. 294–298.
W3C Consortium. Simple object access protocol. Online available via 〈http://www.w3.org/TR.2000/NOTE-SOPA-20000508〉; May 2000.
Zhu F, Mutka MW, Ni LM. Service discovery in pervasive computing environments. IEEE Pervasive Computing 2005;4(4):81–90.
Zilora SJ, Ketha SS. Think inside the box! Optimizing web services performance today. IEEE Communications Magazine 2008;46(3):112–7.