A Framework for Efficient Web Services Provisioning in Mobile Environments

Khalid Elgazzar, Patrick Martin, and Hossam Hassanein

School of Computing, Queen's University, Canada
Kingston, Ontario, K7L 3N6
{elgazzar,martin,hossam}@cs.queensu.ca

Abstract. Advancements in wireless networks and mobile device capabilities enable ubiquitous access for Web resources anywhere anytime. Web service technologies enable rapid and low-cost development of networked and portable applications. The successful convergence of these technologies produces mobile Web services provisioning, where mobile devices can host and provide Web services. However, the resource constraints of mobile devices and the characteristics of wireless networks pose key challenges to mobile Web services provisioning. Several research efforts have studied the Web services provisioning from mobile devices, however, they address specific aspects in isolation which may yield inefficient Web service provisioning. This paper proposes a generic framework for efficient Web services provisioning in mobile heterogeneous environments with resource-constrained mobile devices. We demonstrate the proposed framework using a use-case scenario and a sample prototype. A preliminary provisioning prototype shows that the framework is able to provide reliable and personalized Web services while maintaining the service availability.

Keywords: mobile devices, Web services, service provisioning, mobile computing.

1 Introduction

Ubiquitous information access through mobile devices has become a typical practice in everyday life. Mobile users naturally expect that their mobile devices support "anywhere, anytime" communications and computing while on-the-move. Indeed, recent advancements in mobile device manufacturing coupled with the rapid evolution of wireless technologies offer the promise of a new prosperous era of mobile computing paradigms and seamless data access. However, the error-prone communications that characterize wireless networks and the resource limitations of mobile devices present challenges for leveraging this type of convenient access of information.

Web services, on the other hand, allow applications written in different programming languages for different platforms to interact seamlessly through standard protocols. These protocols are independent from any platform and underlying operating system. While mobile devices span a wide range of device form factors with a variety of different capabilities and different platforms, applications that are developed for a specific mobile platform are not fully compatible, in most cases, with other mobile platforms. Web services are then a method of enabling interoperable mobile applications through standard message exchange communications.

With the ever-expanding mobile-customer base and the growing consumer interest of ubiquitous Internet access from mobile devices, people tend to use their mobile devices to access the Web and carry out their daily life. Hence, mobile Web services again become an interesting approach. The idea of providing Web services from mobile devices facilitates the integration of Web service offerings into mobile applications while putting the flexibility of managing and administrating self-offered Web services at hand. Mobile devices are typically equipped with multihoming capabilities, i.e. multiple network interfaces with different wireless technologies, which means mobile providers can select the best network interface for provisioning or switch to another interface in case of primary network failure.

Mobile web services offer the great potential of a multitude of services and opportunities; marking a shift in how everyday businesses interact and perform. Real-time access to context information would facilitate time-critical applications, such as healthcare and location-based services. Emergency disaster situations form another rich domain in which mobile Web services may play a critical role. More importantly, mobile service provisioning would expand on devices embedded on mobile devices, such as cameras or GPS units.

Several important considerations should be taken into account when designing mobile Web services to cope with both mobile and wireless environment constraints, yet provide Web services efficiently. Much effort has been put forward to investigate the efficient access of Web services by wireless mobile devices, however, less attention has been given for mobile Web service providers. Although there have been several research efforts concerning mobile services provisioning, these research efforts address a specific aspect, such as asynchronous invocation [1] or service replication [2], and are done solely in isolation of other issues. The research presented in this paper is an attempt to bring all aspects together to form a big picture of the efficient Web service provisioning from mobile devices.

This paper proposes a generic framework for efficient Web service provisioning from mobile devices. The framework aims at exploiting the features of mobile devices including their ubiquitous wireless access, multihomed interfaces, their attachment to specific context information such as user and location, and their capability to behave in an ad-hoc manner. At the same time the proposed framework tackles most of the mobile environment constraints such as intermittent connectivity and diversity of mobile device form factors. The framework, hence, satisfies the essential requirements for service provisioning in mobile wireless domains such as ubiquitous service access by supporting different wireless access technologies, persistent service availability by supporting provider/customer disconnection-handling, and adaptability for different device form factors.

The remainder of this paper is organized as follows. Section 2 outlines some of the related works. Section 3 presents the essential requirements for efficient mobile Web services provisioning. A brief overview of how mobile devices are capable of achieving ubiquitous connectivity is given in Section 4. The description of the framework components is introduced in Section 5. Section 6 presents a use-case scenario and a preliminary prototype implementation and results. Lastly, Section 7 concludes the paper and outlines future research directions.

2 Related Work

Advancements in mobile device manufacturing, in addition to wireless technologies, triggered a great interest in the research community in adopting mobile Web services. Several research efforts have studied Web services provisioning from resource-constrained providers, however, they address specific aspects in isolation which may yield inefficient provisioning.

El-Masri [3] addresses enabling hosting capabilities in mobile devices. The author presents a framework for a "Mobile Web Server" that supports Web services provisioning and Web applications from mobile devices. Mobile providers and clients communicate with the mobile Web server and each other through their respective network operators. Services that live on mobile devices are registered in a public UDDI registry and are made accessible for both mobile and Web applications through the mobile Web server.

Singh et al. [1] introduce a framework that supports asynchronous invocation of mobile Web services with the integration of telecommunication services such as SMS messaging. It supports disconnected operations and releases the client device once the service operation is invoked. Then, the SOAP response is dispatched to the terminal when it's ready via the SMS protocol.

A recent direction on reducing the overhead of services provisioning in resource-constrained providers has been explored by Hassan et al. [4]. They propose offloading some of the workload of complex services from mobile devices to the Cloud. Their framework proposes a distributed SOAP engine that runs a group of tasks on mobile devices and others on a static server. A partitioning strategy is presented to arbitrate the split of service tasks.

In an effort to cater for the constrained resources of mobile devices, Kim et al. [5] propose a lightweight framework for hosting Web services on mobile devices. The core of their work is improving service availability via service migration. The migration is initiated by the mobile service provider or in response to a context change. The migration *manager* selects the candidate new host based on a context-suitability function that comprises host capabilities and service requirements. Although the framework supports only SOAP-based Web services the main idea is still valid for REST-based Web services.

Service replication is another way to approach the service availability problem in dynamic mobile environments due to providers' mobility, battery outage, or environmental changes. A framework that supports Web service migration for achieving seamless provisioning is proposed in [5]. The migration takes place as per the provider's request or a change in the context information. Candidate hosts are chosen from neighbor devices based on a suitability function.

Sheng et al. [2] propose an on-demand replication approach to reduce the risk of Web service unavailability under large volume of service requests. Their approach deploys services, on selected idle hosts based on a multi-criteria utility function that takes into consideration service requirements. The replication model calculates the number of required replicas to maintain a predefined level of service availability.

In the case of multihomed mobile devices, Meads et al. [6] discuss Web services provisioning from devices with multiple wireless interfaces. The authors present a middleware infrastructure that relies on *Jini* surrogate architecture and supports vertical handover between Bluetooth and HTTP. The handoff mechanism between different *interconnect channels* can be initiated by the user (proactive) or in response to a context change (reactive). Enabling HTTP connections over Bluetooth channels is also investigated by Auletta et al. [7]. The authors introduced a Bluetooth Http Server Proxy (BHSP) that handles the interface between Bluetooth-enabled clients and HTTP Web server. Their implementation of this interface extends the J2ME standard class `HttpConnection` to `BtHttpConnection` class that supports the underlying communication on a Bluetooth channel transparently.

Despite current research efforts on facilitating service provisioning on mobile devices, a holistic approach that provides Web services from resource-constrained providers (i.e. mobile devices) is still lacking. Specifically, one that takes into account their specific features and limitations. In this paper, we address this void by proposing a framework for Web services provisioning from mobile devices in mobile heterogeneous environments; adopting reliability and efficiency in provisioning as core metrics. We provide a high level description of the framework components and describe the applicability of related approaches in it.

3 Requirement for Efficient Mobile Web Services Provisioning

Resource limitations of mobile devices, such as memory, computational power, battery life, and wireless network characteristics, such as intermittent connectivity and limited bandwidth pose several challenges for mobile services provisioning. An efficient service provisioning framework takes advantage of the unique features of mobile devices and reduces the impact on performance that stems from providing services from resource-constrained providers attached to wireless networks.

A successful framework should satisfy the following essential requirements:

1. Efficient provisioning must adopt existing Web service and wireless networking standards. The framework should be flexible, interoperable and independent of any underlying platform.
2. Since wireless communications are often unreliable and mobile providers may change their point of attachment to the network as they move, full support

of wireless intermittent connection-handling, and seamless handoff for both service providers and customers, are a must.

3. Availability of Web services is an important QoS parameter; especially in mobile domains where the constraints of mobile devices and unreliable wireless connections may diminish the possibility of service availability. Mobile Web services may become inaccessible, mainly due to: host battery outage, connection loss, system crash either of mobile devices or of Web service container, large number of concurrent service requests, and bandwidth limitations. To maintain a specific threshold of service availability and reliable delivery, mechanisms for service replication and asynchronous service invocation should be supported.
4. Service provisioning should enable ubiquitous access by utilizing the multihoming feature; especially that it exists in most current mobile devices.
5. Flexible service provisioning must provide the necessary adaptation mechanisms to accommodate a variety of consumer device form factors .
6. A chief benefit of mobile services is the provisioning of personalized services, which are services tailored specifically to best fit within a particular context. Providing personalized services, hence, incorporates user preferences, device profile, and other context information in service discovery and execution.
7. Web services, generally speaking, can be designed using SOAP messaging protocol to exchange messages (SOAP-based), or conforming to the REST principles (REST-based). Although REST-based approach is lightweight and suitable for resource-constrained providers, an efficient provisioning must handle requests from both SOAP-based and REST-based Web services.

The mobile services provisioning framework proposed in this paper satisfies all the above requirements, and paves the road for a better understanding of efficient services provisioning in dynamic mobile environments.

4 Mobile Devices Ubiquitous Connectivity

Mobile devices, specifically smartphones, have become the most convenient ubiquitous computing interface due to the advancements in their capabilities and functionalities, which have gone far beyond just providing traditional telephony services. Nowadays, mobile users can access a wide range of data services and Web applications over multiple wireless technologies including 3G, WiFi, Bluetooth, and more recently ZigBee [8]. These wireless interfaces differ in their capacity, transmission range, and power consumption. Mobile devices can handover between different access technologies not only to achieve ubiquitous connectivity transparently to the running applications, but also to enhance the quality of services they receive or deliver.

Most of contemporary mobile devices are equipped with short range/low-power wireless connectivity, such as Bluetooth, to establish connectivity with one another in an ad-hoc manner. Bluetooth provides free wireless access to mobile users in the unlicensed frequency band with no cost. Bluetooth standards enable automatic discovery of mobile devices and network services in the device transmission range. Therefore, Web resources such as Web services can be provided over Bluetooth technology [6]. Communication over Bluetooth could be useful when providers and customers are within proximity of each other to reduce the cost and save battery power for both of them.

ZigBee offers greater coverage compared to Bluetooth and consumes less energy compared to WiFi, while having relatively limited data rates. ZigBee technology particularly targets applications that demand low bandwidth. Jin et al. [8] designed and developed a WiZi-Cloud system with a ZigBee interface for mobile devices to overcome some of the constraints that exist with other wireless technologies such as the short range of Bluetooth and the high energy consumption of WiFi. Their system supports seamless handover between WiFi and ZigBee technologies, easy integration with existing HW/SW, flexibility in determining which interface is appropriate to use, and efficient energy consumption.

5 Service Provisioning Framework

Mobile environments are characterized by intermittent connections and constantly changing signal quality. Mobile devices have limited-resources and change their point of attachment to the network or their access technology frequently. Our framework aims to provide seamless and reliable mobile Web services in dynamic mobile environments. Figure 1 illustrates the framework components. The framework comprises nine essential components: request/response handler, service execution engine, response representation, context manager, directory manager, performance monitor, disconnection support, publication/discovery, and user feedback manager.

5.1 Request/Response Handler

Generally speaking, users can request access to Web services or other Web contents. A Web service request could be SOAP/XML for SOAP-based Web services or HTTP request for REST-based Web services. The *Request/Response Handler* plays the role of a multiplexer, it receives users' requests and differentiates between "Web service" requests and "Web content" requests. The handler forwards the latter directly to the Web server whereas the former is sent to the service *Execution Engine* for processing. SOAP/XML service requests are handled by *SOAP Manager* before they sent down to the Web server, whereas HTTP requests for Web services are analyzed directly by a servlet which selects the appropriate class and method to respond to these requests based on the class and method annotations. The proposed *Request/Response Handler* supports HTTP and other protocols such as Bluetooth and ZigBee. If a service request is received over a Bluetooth or ZigBee channel the message will be forwarded to the *Bluetooth/ZigBee Protocol Handler*. Likewise, responses for Web service requests are treated the same except that they are put in a device-compatible form by the *Response Representation* before sending them to the requester.

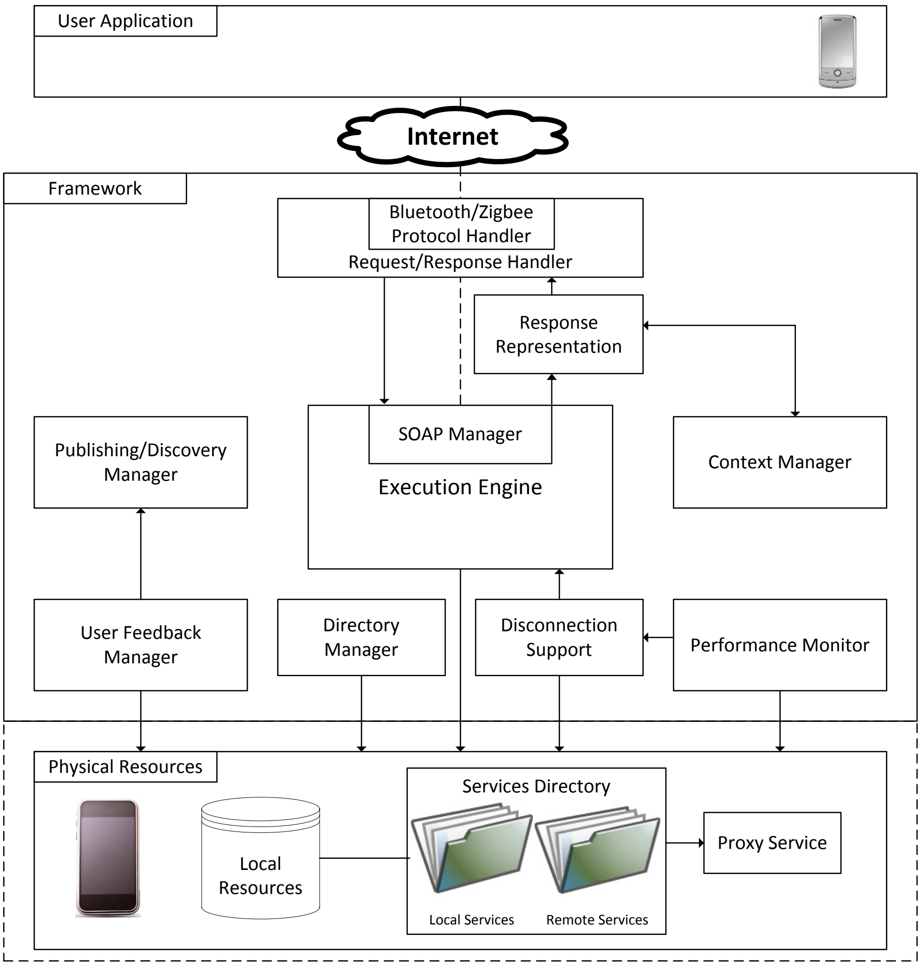


Fig. 1. Illustration of the service provisioning framework components

5.2 Bluetooth/ZigBee Protocol Handler

Most of the current mobile devices and smartphones are equipped with a sort of short range wireless connectivity interface such as WiFi and Bluetooth (ZigBee technology is envisioned to be embedded soon [8]). The *Bluetooth/ZigBee Protocol Handler* then provides the essential mapping between the used communication channel format (channel-specific logic [6]) and HTTP format to be accepted at the Web server interface (satisfying requirement #4). In this regard, an infrastructure that supports channel switching between Bluetooth and HTTP seamlessly while provisioning Web services from multihomed mobile devices is presented in [6]. The handler can also implement a protocol-specific proxy server to deal with a request/response at one side and forward it for processing to the Web server at the other side [7].

5.3 Execution Engine

After the *Request/Response Handler* de-serializes the XML data that is associated with the service request and converts it to a Java object, the service *Execution Engine* invokes the corresponding method for execution. The execution engine can handle both SOAP-based and RESTful Web services (requirement #7). The execution engine includes a *SOAP Manager* [5] to handle SOAP-based Web service communications and SOAP messages. In contrast, RESTful Web service requests are executed directly by the servlet container. Once the execution is done, the response is sent to the response representation, which then performs the proper response optimization to fit within the requester's device before forwarding it to the *Request/Response Handler*.

5.4 Response Representation

While most of today's mobile devices feature some sort of Web browsing capabilities, Web navigation by mobile devices has not become as convenient as it is on their desktop counterparts. However, some research efforts have been put forward to take advantage of the device profile to support device-related adaptations for Web content delivery [9]. Generally, mobile devices differ in their display features and screen capability support. Some devices have small screens with limited colors while others have relatively large displays with high resolutions and full color support. Web service responses could be sent as simple as text or rendered with graphic demonstrations and/or multimedia entities. The *Response Representation* receives the device information from the *Context Manager* and adapts the service response accordingly (requirement #5). The *Response Representation* can also take into account the available transmission rate and choose to send simple but representable format from the response to cope with the limited transmission rate despite the fact that the requester's mobile device is powerful and features a large display. From that perspective, Ortiz et al. [10] employed the Aspect-Oriented Programming and model-driven developments to develop flexible Web services that can adapt to mobile devices and produce appropriate responses according to the device capabilities.

5.5 Context Manager

The *Context Manager* collects context information about the customer's device (device profile), customer preferences (customer profile), running environment status (i.e. transmission rate, available bandwidth). The objective of collecting these contexts is to use them for Web service discovery and personalized service provisioning, which satisfies requirement #6 of our defined requirements for efficient service provisioning in mobile environments that we stated earlier. Using the device profile in Web service discovery ensures that the discovered/selected service will function properly with in the mobile device [11]. Service provisioning, especially in mobile domains, can take advantage of the knowledge about service requesters to provide services that best match with their preferences, hobbies, habits, and maybe current location [12]. The environment context can be used for service selection/ranking to ensure a reliable execution for the selected Web service given a certain environment status. For example, at the discovery time, services with multiple behaviors or a service that can adapt to the environment context [13,14] would be ranked first or given a higher priority, whereas at the running time, a switch to a service that is more appropriate within the new environment context could be performed.

5.6 Directory Manager

The *Directory Manager* is responsible for managing the services offered by mobile devices. They fall under two categories: *local services*, which are hosted and provided by the mobile device, and *remote services*, which are "active and running" services that are hosted on other mobile devices but provided by a local proxy service; or simply announced to others through local publishing mechanism/broker.

5.7 Performance Monitor

The *Performance Monitor* component aims at managing the behavior of Web services to ensure that the performance meets the requirements of Service Level Agreements (SLA). Tracking the performance of mobile Web services enables mobile providers to decide on a proper action in response. A proper action could be blocking any upcoming service requests to keep services up and running and their performance meets the requirements' threshold until more resources become available. Another option for mobile providers if they become overloaded during peak demand periods is to offload the service to the cloud or to replicate it to some idle hosts as suggested by Sheng et al. [2]. Performance monitoring reports the status of specified requirements that may include response time, latency, and bandwidth availability. Some commercial products exist that perform service monitoring using API's and can send alerts to identify and resolve performance issues based on predefined configurations.

5.8 Disconnection Support

Mobile users typically suffer from intermittent connectivity. Consuming or providing mobile Web services must efficiently handle this property of wireless communications. The traditional Web services-based communication assumes that the service consumer and the service provider maintain an active connection until a response is received by the consumer or a connection time out occurs. The *Disconnection Support* manager provides mechanisms to support disconnected consumers as well as disconnected providers, which satisfies requirement #3.

Supporting Disconnected Consumers. Reliable provisioning of Web services in mobile domains means that services should function properly during the absence of network connection. This means that the consumer could be blocked as long as the service is executing, which is not be feasible in mobile wireless networks, especially for services that could have a lengthy execution [15]. Asynchronous invocation then is a viable alternative that can be used for Web services based communication in dynamic mobile environments, where consumers may disconnect after invoking the required Web service and interaction techniques such as "Callback" and "Polling" can be used to communicate the response. *Asynchronous Service Access Protocol (ASAP)* [16] is specifically designed by OASIS targeting the service interactions in Web services that require complex processing or lengthy execution. ASAP enables services to run asynchronously (in terms of response handling) and independently from their caller. The service consumer invokes the service and the response is sent back whenever it is available or a later separate request can be made by the consumer to communicate the results of the Web service operations. ASAP enables consumers to send update messages and change previously sent parameters or information if the consumer requirements change during the service execution. The asynchronous Web service should then have the ability to update itself accordingly at runtime.

Supporting Disconnected Providers. Mobile providers similarly suffer from intermittent connections which could make their Web services temporarily inaccessible. For a reliable and seamless provisioning, mobile providers in such cases should provide users with service access alternatives to ensure continuous service availability. The issue of providers' connectivity outage can be approached in three ways, caching Web services, service replication, and using the Cloud to host the services of the disconnected providers. Caching Web services relies mainly on the existence of a proxy that can take over the service provisioning whenever the original provider experiences a network disconnection. However, this approach requires additions to WSDL specifications. The proxy may partially or fully support the service operations depending on the nature of the service and whether its operations need access to a real time data from the mobile provider or not. Furthermore, Web services are typically active entities and need access to live data most of the time, hence HTTP passive caching may not be an appropriate approach for them. Service replication then is another alternative to tackle this problem, where providers that experience frequent connection loss may deploy or replicate their services on other available nearby hosts. The deployment of replicas could be transparent to service consumers, i.e. they need not to be aware of the underlying replication [2]. The third option supports robust service provisioning, where providers may resort to hosting Web services on the Cloud not only when they experience or expect connection loss, but also when their services become inaccessible due to the large amount of invocations. Clouds usually perform elastic resource assignments, that is more resources could be acquired on demand to cope with the high volume of service requests and released when these resources are not needed anymore (resources can grow up and shrink on demand).

5.9 Publishing and Discovery Manager

Publishing is carried out by service providers to inform potential customers of the existence of Web services. Discovery is the process of finding a Web service that fulfills a particular consumer objective. There could be a mediator (service broker) that facilitates the link between service providers and consumers. Providers typically have two options to publish their services; either by registering the services with a service broker in a public service repository, using the UDDI standard, or publishing the services in a local service directory. Also, providers may publish two types of services, services that they host and provide locally (local services) and services that they know about from other providers (remote services) and both are managed by *Services Directory Manager*. Providers may also provide a proxy service for remote services. Local services typically have access to locally managed resources such as databases or real time data collectors (i.e. sensors and embedded devices). In P2P overlay networks, Web services are often published as JXTA modules, each module includes a module class, module specification, and module implementation [17]. The module class basically represents the information needed to declare the services. The module specification contains the information required for the potential consumer to access the service, whereas implementation indicates the methods (possibly across different platforms) of the advertised specifications. A high level description of a distributed publishing of Web services in P2P networks is proposed by Sun and Hao [18]. In their framework, Web services are registered on a specific peer according to their contained ontologies. In contrast, publishing and discovery in client/server based interactions follow the traditional Web service standards (i.e. UDDI).

5.10 User Feedback Manager

Service providers may dishonestly claim unrealistic QoS capabilities for their advertised services to attract interested consumers. The QoS is primarily of particular interest to distinguish services with equivalent functional properties. *User Feedback Manager* collects the perceived QoS by consumers to reflect reality and users' satisfaction, which can be then exploited to improve the service delivery and draw providers' attention to issues that concern service consumers. Although, the previous research work done on this perspective [19] targets the improvements of service discovery and ranking, the same concepts could be applied to enhance the service delivery by collecting consumers feedback. However, mechanisms that prevents or at least reduces false ratings and misleading feedback should be incorporated in collecting users' feedback for credibility.

6 The Egyptian Revolution, 2011: A Use-case Scenario

In the Egyptian revolution of 2011, people protested to get rid of a dictatorship. The call for the revolution began on social networks and people organized themselves and communicated mainly over Facebook. Some essential communications are required in such scenarios, as those calling for leaders to share some information, coordinate between groups, communicate messages to protesters or arrange for medical emergencies, etc. In such a scenario, we look at leveraging communication capabilities with mobile Web services provisioning, aiding protesters to communicate with each other at their best; especially for carrying out crucial tasks like tending to medical emergencies.

To provide medical emergency services, health professional volunteers created a fixed medical emergency facility at each gathering square in each city where the vast majority of protesters exist, as well as some mobile medical emergency units that can promptly respond to injured persons at their locations. These emergency units should be able to announce their locations to the crowds and receive injury reports from protesters. The service provisioning, at the same time, should maintain a specific level of reliability, availability, and adaptability.

According to our framework we show how to provide reliable mobile services from mobile devices in such a scenario taking advantage of mobile phones available with the protesters and the P2P overlay network that they can form together. A Web service can be developed and deployed on the mobile device of the medical emergency volunteers. The *Publication/Discovery* component registers the Web services on the local services directory and advertises the services to all peers in the transmission range, the same way Web service requests are handled in P2P overlay networks. Peers, consequently, cooperate to disseminate the service advertisement. The advertisement in this case could be as simple as a URL that could be used to access the Web service interface associated with some human readable service description. Whoever receives the advertisement originates a copy to its neighbors after reducing the Time To Live (TTL), that is associated with the advertisement, by 1. Peers stop forwarding the service advertisement if $(TTL = 0)$. This is marginally different than the traditional service publications, where users initiate a Web service request to discover relevant services that fulfill their objectives. The TTL value controls how far a service advertisement can go.People then can access these services and submit reports of injuries to the medical emergency volunteers who would respond appropriately.

The *Disconnection Support* component provides the required mechanisms for mobile devices that lose the connectivity due to becoming temporarily out-of-range of any supporting peer, whether they forward a response or a service call. The type of support depends on whether the disconnected peer is a service provider or a service consumer. In case of service providers the support action could be a service replication to a capable connected peer [13], while in service consumer it could be integrating some telecommunication services such as SMS [1]. In both cases the asynchronous service invocation is a viable option [1]. The *Context Manager* in this scenario provides network status reports and maintains a reference for all the available network connectivity. In case the active connection goes down for any reason, a seamless switching procedure to one of available networks can be issued preserving the status of all active connections [6,7].

Our framework can also handle the heterogeneity of client device form factors by adapting the service response to cope with their capabilities. The *Response*

Representation module changes the format of the service response to best fit within the mobile device's profile parameters. If the client's mobile device supports only text mode, a simple text that represents the service response is sent to the client. Otherwise, a more appealing graphical representation is sent to capable terminals.

On January 28, 2011, the Egyptian government shut down the Internet and cellular networks to isolate protesters and prevent them from communicating. Our framework would have enabled service requests and invocation of Web service functionalities over Bluetooth or any other short range wireless connectivity (that already exist on most of mobile devices) via the *Bluetooth/ZigBee Protocol Handler*. At the same time, when better connectivity (e.g. 3G or WiFi) becomes available, the transition will be seamless and transparent with no interruption to the provisioned services [6].

Figure 2 illustrates a sample behavior of our framework and actions taken to maintain the service availability in this scenario. We assume that the mobile service provider has access and is connected to WiFi and 3G networks. In such cases, WiFi is more preferable due to the higher bandwidth availability in contrast with the 3G. The *Request/response Handler* receives the service requests from different users over different wireless channels and provides the proper protocol handling. The handled request is then forwarded to the service *Execution Engine* which invokes the appropriate method. At the same time the *Context Manager* takes advantage of HTTP sessions and collects the device profile parameters. The *Response Representation* component, which is an application specific module, receives both the device profile and the raw service response to reformat the response to best fit within the device profile. At some point of time, suppose that WiFi connectivity is lost due to the provider's mobility or any other reason. The *context Manager*, therefore, reports this network loss to the *Disconnection Support* component, which in turn replicates the service on another available candidate host as a protective action (set by the service administrator) to maintain service availability and reliable provisioning. The candidate host of the replica and it's location are out of this paper's scope, however, these issues are studied in [20,2]. We also assume that the consistency between replicas is handled by underlying Web server/system. When the service experience a large number of requests which violate the performance goals (such as a response time threshold set by the service administrator), the *Performance Monitor* reports this violation and another replica is triggered to keep the service performance within certain limits. During the performance violation interval, any upcoming requests are temporally rejected until another replica becomes up and running or the violated parameters fall down under their threshold.

6.1 Implementation

We have implemented a preliminary prototype system to test and validate our framework for the above scenario. Our prototype focuses on the reliability and adaptability issues (i.e. how to handle service availability and provide different formats of same service response to different customers).We conducted several

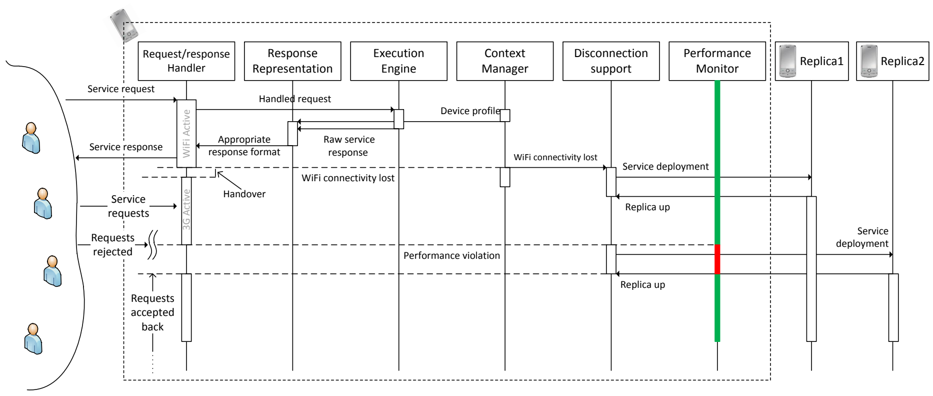
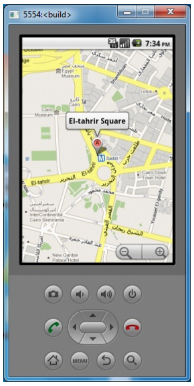


Fig. 2. A sample behavior of the mobile services provisioning framework and actions taken to maintain the service availability

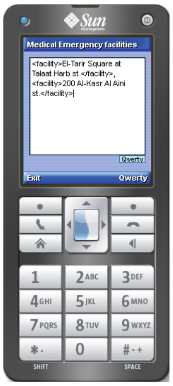
experiments to demonstrate the effectiveness and reliability of the service provisioning framework, where phone-based users can mainly depend on their mobile devices to seek and receive help in emergency situations. We developed a RESTful Web service that sends the locations of field medical assistance facilities and accepts medical emergency requests. We chose the REST-based design for Web services because of its lightness and scalability. RESTful Web services typically communicate over HTTP using standard HTTP methods such as “GET” and “POST”. The Web service is deployed on a mobile device using Apache Tomcat 7.0 as a servlet container.

Our prototype demonstrates three aspects: the first is how the framework supports providers who experience or expect frequent disconnection and when to maintain the service availability; the second is how the framework provides reliable Web services and responds to performance violations, and the third is how the framework handles the heterogeneity of mobile device form factors. We emulate concurrent users using Java mulithreading. The *Performance Manager* observes the average response time to maintain a certain level of QoS. When the response time exceeds the predetermined threshold, the service request handler rejects any upcoming requests to keep services alive and issues a service replication command. For the sake of simplicity we perform the service replication using the remote deployment feature that Apache Tomcat 7.0 provides.

In this prototype, service replication is performed if the *Context Manager* reports a network loss (as long as an alternative connection is available) or the threshold of the average response time exceeded. We set the response time threshold, perceived at the server side, to 80 ms (as per the SLA for example). We remark here that we are not testing the network performance, but rather evaluate how our framework responds to context changes and QoS violations to maintain a certain level of service reliability and availability. Any other set of criteria could be set.



(a) Large and high resolution display


Medical Emergency facilities

```
<facility>El-Tarir Square at
Talaat Harb st.</facility>,
<facility>200 Al-Kasr Al Aini
st.</facility>
```

(b) Limited display capabilities

Fig. 3. Service response adaptation based on device profile

After successful deployment of a replica, the framework updates the service endpoints by adding the new service access point to the alternative endpoints in the description file, so that customers may try alternative endpoints if the primary host does not respond, if the replication strategy allows such communication. The replication strategy [20], which regulates whether customers are allowed to communicate with replicas directly or through the primary host, is out of this paper's scope. However, we update the service description file with the alternative access points to make it ready for any replication strategy.

To demonstrate how the framework adapts the service response according to the client's device profile, we conducted an experiment to access the Web service from different devices with different capabilities (more specifically in terms of display size and graphics support). In this experiment, we obtain device profiles from the emulator specifications. It worth noting here that this experiment is application dependent and yields application specific service response formats. Figure 3 shows different response formats for different mobile devices despite the same response content. Figure 3a shows a map view of the locations of the medical emergency facilities that protesters created at El-Tahrir square, Cairo. Although, we generate the map at the client side using location information, it may be generated at the server side and sent as an image. This response viewed on a standard Android emulator. Figure 3b shows the same information formatted as a simple XML string (as it is extracted from the HTTP-GET response) viewed on the default CLDC "DefaultCldcJtwiPhone" emulator that is embedded with Java ME SDK 3.0 [21]. This CLDC emulator has a limited display size of 180x208px.

7 Conclusion

Mobile devices have become the most convenient interface for pervasive and ubiquitous computing as they offer “anywhere, anytime” communication and computing while on-the-move. The ever-increasing numbers of mobile customers are becoming more interested in accessing ubiquitous information from their mobile devices. Mobile Web services provisioning is an approach in which mobile devices play the role of service providers. In this paper, we introduce a holistic approach for effective Web services provisioning from mobile devices in wireless mobile environments. We present the essential requirements of efficient provisioning and propose, based on these requirements, a generic framework that takes advantage of mobile device and wireless network features and hide their limitations. The framework proposes disconnection support for mobile service providers through service replication and service consumer through asynchronous service invocation. A preliminary prototype has been developed to evaluate the proposed framework. Testing the response adaptation yields different response formats for different customers based on their device profile.

Our ongoing research will extend the prototype implementation of the proposed framework to include the efficient service discovery and maintain the service continuity during the handover process.

References

1. Singh, R., Mishra, S., Kushwaha, D.S.: An efficient asynchronous mobile web service framework. SIGSOFT Software Engineering Notes 34, 1–7 (2009)
2. Sheng, Q.Z., Maamar, Z., Yu, J., Ngu, A.H.H.: Robust Web Services Provisioning through On-Demand Replication. In: Yang, J., Ginige, A., Mayr, H.C., Kutsche, R.-D. (eds.) UNISCON 2009. LNBIP, vol. 20, pp. 4–16. Springer, Heidelberg (2009)
3. El-Masri, S.: A framework for provising mobile web services. In: The Second International Conference on Innovations in Information Technology, IIT 2005 (2005)
4. Hassan, M., Zhao, W., Yang, J.: Provisioning web services from resource constrained mobile devices. In: Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing, CLOUD 2010, pp. 490–497 (2010)
5. Kim, Y.-S., Lee, K.-H.: A lightweight framework for mobile web services. Computer Science - Research and Development 24(4), 199–209 (2009)
6. Meads, A., Roughton, A., Warren, I., Weerasinghe, T.: Mobile service provisioning middleware for multihomed devices. In: Proceedings of the 2009 IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, pp. 67–72. IEEE Computer Society, Washington, DC (2009)
7. Auletta, V., Blundo, C., De Cristofaro, E.: A j2me transparent middleware to support http connections over bluetooth. In: Proceedings of the Second International Conference on Systems and Networks Communications, pp. 3–9. IEEE Computer Society, Washington, DC (2007)
8. Jin, T., Noubir, G., Sheng, B.: Wizi-cloud: Application-transparent dual zigbee-wifi radios for low power internet access. In: The 30th Conference on Computer Communications, INFOCOM 2011, Shanghai, China (2011)
9. Zhang, D.: Web content adaptation for mobile handheld devices. Commun. ACM 50, 75–79 (2007)

10. Ortiz, G., Prado, A.G.D.: Improving device-aware web services and their mobile clients through an aspect-oriented, model-driven approach. Information and Software Technology 52(10), 1080–1093 (2010)
11. Al-Masri, E., Mahmoud, Q.H.: Mobieureka: an approach for enhancing the discovery of mobile web services. Personal Ubiquitous Computing 14, 609–620 (2010)
12. García, J.M., Ruiz, D., Ruiz-Cortés, A.: A Model of User Preferences for Semantic Services Discovery and Ranking. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part II. LNCS, vol. 6089, pp. 1–14. Springer, Heidelberg (2010)
13. Maamar, Z., Tata, S., Belaid, D., Boukadi, K.: Towards an approach to defining capacity-driven web service. In: International Conference on Advanced Information Networking and Applications (AINA 2009), pp. 403–410 (2009)
14. Tao, A., Yang, J.: Context aware di erentiated services development with congurable business processes. In: Proceedings of the 11th IEEE International Enterprise Distributed Object Computing Conference, pp. 241–252. IEEE Computer Society, Washington, DC (2007)
15. Aijaz, F., Hameed, B., Walke, B.: Towards peer-to-peer long lived mobile web services. In: Proceedings of the 4th International Conference on Innovations in Information Technology, pp. 571–575. IEEE, Dubai (2007)
16. Fuller, J., Krishnan, M., Swenson, K., Ricker, J.: Oasis asynchronous service access protocol, asap, May 18 (2005), http://www.oasis-open.org/committees/documents.php?wg_abbrev=asap
17. Gong, L.: Jxta: a network programming environment. IEEE Internet Computing 8, 88–95 (2001)
18. Zhen Sun, F., Gang Hao, S.: A discovery framework for semantic web services in p2p environment. In: 2010 International Conference on Electrical and Control Engineering (ICECE), pp. 44–47 (June 2010)
19. Averbakh, A., Krause, D., Skoutas, D.: Exploiting User Feedback to Improve Semantic Web Service Discovery. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 33–48. Springer, Heidelberg (2009)
20. Juszczyk, L., Lazowski, J., Dustdar, S.: Web service discovery, replication, and synchronization in ad-hoc networks. In: Proceedings of the First International Conference on Availability, Reliability and Security, pp. 847–854 (2006)
21. Java (tm) platform, micro edition software development kit 3.0 (June 2011), http://www.oracle.com/technetwork/java/javame/javamobile/download/sdk/index.html