A Specification Approach to Compose Mobile Web Services Using Service Chart Diagrams

Zakaria Maamar and Mohammed Lahkim

College of Information Systems
Zayed University, Dubai, U.A.E
{zakaria.maamar,mohammed.lahkim}@zu.ac.ae

Abstract. We present our work on composing mobile Web services, denoted by M-services, in the wireless world. The wireless world has its own features that make it completely different from the wired world. For instance, new communication means need to be deployed, new user-friendly services need to be offered, and new types of specification techniques need to be provided. One of these techniques consists of using service chart diagrams in the specification of a composition process of M-services. A service chart diagram leverages the traditional state chart diagram of UML and specifies a service from five perspectives: state, flow, organization, information, and location.

1 Introduction

With the widespread and familiarity of the Web technology, an increasing number of businesses have decided to offer Web services over the Internet [10]. Those wishing to prepare their vacation can, for instance, access a travel web site, specify their needs (e.g., favorite airliner, accommodation type, and duration), and then trigger the appropriate Web services. Various technologies are behind the success of Web services such as WSDL, UDDI, and SOAP [2]. The advantages of Web services have mainly been highlighted by their capacity to be composed into high-level business processes.

Besides the Web expansion, we all witness the growth occurring in the wireless field. Telecom companies are offering new services to their customers over mobile devices. Reading emails and sending messages between cellular phones are becoming natural. Surfing the Web through WAP is another evidence of this growth. The next stage (if we are not already in it) for telecom companies in collaboration with businesses is to allow users to enact Web services from their mobile devices and possibly, to make these Web services runnable on these devices. M-services (M for mobile) denote this type of Web services.

It happens that a mobile user has to postpone her operations because she lacks appropriate facilities running on her mobile device (e.g., an application that converts a drawing file into a format that the user's cell-phone can display). In SAMOS (**S**oftware **A**gents for **MO**bile **S**ervices) project, we aim at supporting such users by allowing them: 1) to search for additional facilities, when needed; 2) to fetch these facilities to their mobile devices; and 3) to conduct the aforementioned operations in a transparent way. Different solutions are put forward to deal with those issues. A solution to 1) consists of devising brokering mechanisms. A solution to 2) consists of using wireless

communication channels. Finally, a solution to 3) consists of using Software Agents (SAs) to make searching for facilities and fetching them transparent to users [5]. In this paper, we focus on presenting the technique that is used for the specification of the composition of M-services.

Section 2 motivates our work on M-services. Section 3 defines Web services *vs.* M-services. Section 4 introduces the agent-based architecture of SAMOS. Section 5 discusses the composition of M-services using service chart diagrams. Section 6 overviews SAMOS related projects. Finally, Section 7 concludes the paper and presents our future work.

2 Motivation

Besides the new role of the Internet as a vehicle of delivering Web services, we expect that more Web services will be offered to users of mobile devices and particularly to those who are most of the time on the move (e.g., sales representatives). It is, also, expected that mobile devices will be enhanced with new advanced computing resources (some devices are already; since 2001 Java enabled I-mode phones are available on the Japanese market it is becoming possible to download Java applets from servers to be run on these phones). Unfortunately, the growth in the development and use of mobile devices does not come alone. It has its lot of obstacles and challenges. For instance, mobile devices are strictly bound to their batteries for operation leading to limit their computation power.

There exist several types of mobile devices, varying from cell-phones to personal digital assistants. These devices present several shortcomings that make their use requires specific arrangements. Among these arrangements, the services have to be tailored to each mobile device individually. Unfortunately, this contradicts the platform independence principle that Web services promote [1]. Therefore, leveraging Web services into M-services becomes a necessity. M-services will have to consider the characteristics of the mobile devices on which they will be running. For performance purposes, M-services will be fetched on-demand from provider sites to mobile devices of users.

3 Web services *vs.* M-services

A Web service is an accessible application that can be automatically discovered and invoked by other applications (and humans as well). An application is a Web service if it is [1]: 1) independent as much as possible from specific platforms and computing paradigms; 2) developed mainly for inter-organizational situations rather than for intra-organizational situations; and 3) easily composable.

Two definitions are suggested for an M-service [8]. The *weak* definition is to trigger remotely a Web service from a mobile device for execution. In that case, the Web service is an M-service. The *strong* definition is to transfer a Web service through a wireless channel from its hosting site to a mobile device where its execution takes place. In that case, the Web service is an M-service that is: 1) transportable through wireless networks; 2) composable with other M-services; 3) adaptable according to the computing

features of mobile devices; and 4) runnable on mobile devices. In this paper, we only consider the M-services that comply with the *strong* definition.

In the rest of this paper, a composite service denotes a list of component M-services.

4 Agent-based architecture of SAMOS Brokering

Brokering mechanisms and SAs are suggested as potential candidates in the design and development of a system offering services to users of mobile devices. In a nutshell, the salient features of the SAMOS architecture are:

- Use three types of SAs: user-agent, provider-agent, and device-agent.
- Deploy a Meeting Infrastructure (MI) to be headed by a supervisor-agent [7].
- Use user-delegate and provider-delegate. Delegates are installed in the MI, respectively acting on behalf of user-agents and provider-agents.
- Use storage servers in order to save the composite services to be submitted to the mobile devices for execution. These servers are spread across the network and storage-agents are responsible of their management.

Figure 1 illustrates the architecture of SAMOS. It consists of four parts: user, provider, MI, and storage. The MI and storage parts are linked to the user part in a wireless way. While, the MI and storage parts are linked to the provider-part in a wired way. To keep the paper self-contained, certain aspects of the SAMOS architecture are not detailed.

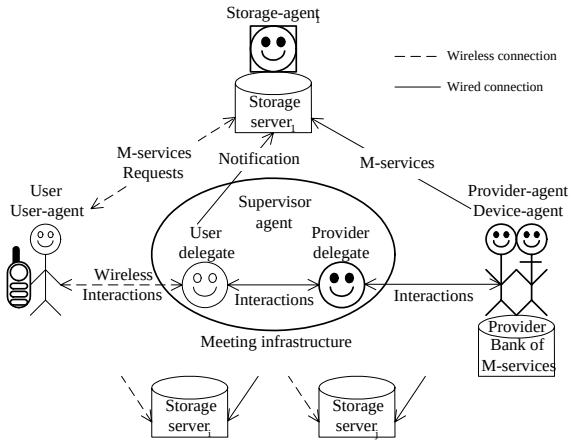

Fig. 1. Agent-based multi-domain architecture

User-agents act on users' behalf. They accept their needs, convert them into requests, and submit them to user-delegates. On a request basis, the MI supervisor-agent creates user-delegates that interact with provider-delegates.

The provider-part consists of providers, provider-agents, and device-agents. Provider-agents act on behalf of providers of M-services. They advertise their M-services to user-delegates, through provider-delegates. In addition, provider-agents monitor the behavior of providers in case of new M-services are being offered and thus, need to be announced. In Figure 1, M-services are gathered into a bank on top of which provider-agents and device-agents are located. Provider-agents create provider-delegates and transfer them to the MI. For device-agents, they support the work of provider-agents. The role of device-agents is to wrap the M-services before they are sent to the mobile devices of users for execution. Wrapping the services is due to the differences that exist between mobile devices (e.g., screen size, processor speed, storage capacity).

The MI part is a software platform within which user-delegates and provider-delegates reside and interact in a local and secure environment. In an open environment, initial interactions between consumers of services and providers of services are conducted through third parties, usually known as brokers. Despite its important role, a broker can easily become a bottleneck. To overcome this problem, consumers and providers have to bypass the broker. In fact, they need a common environment in which they meet. The MI plays the role of this environment. The MI approach has already shown its benefits in the research that was done in [3]. In this research, different interoperability environments based on the following architectures were compared: broker, matchmaker, and meeting infrastructure. Three criteria were used for comparing these three environments: number of messages exchanged, risk of intercepting the messages exchanged, and safety of the messages exchanged. The comparison has shown that the MI has a great potential to support brokering in distributed systems.

The storage part receives the composite services that will be submitted for execution on the mobile devices of users. According to the execution principles of SAMOS and the capabilities of mobile devices, M-services are sent to users' mobile devices one by one (this number can be adjusted based on the capabilities of these devices). Several advantages are obtained from the use of storage-servers. First, mobile devices have a "limited" storage capacity. Second, a user-agent does not have to deal with several providers from which it requests M-services for execution. Its unique point of contact for requesting the M-services is the storage-agent. The same thing applies to device-agents that will be interacting with few storage-agents instead of interacting with several user-agents. Third, security is increased for both users and providers. Storage-servers are independent platforms where security control can be conducted in a safe way.

5 Specification of a composite service of M-services

In SAMOS, a composite service of M-services is an outcome of the collaboration of providers. Each provider contributes to the composite service with one to several M-services. This is determined during the interaction session that occurs between user-delegates and provider-delegates in the MI (Figure 1). To specify a composite service of M-services, service chart diagrams are used [6].

Service chart diagrams are based on UML state chart diagrams [4]. This time, the focus is on the context surrounding the execution of a service rather than on the states that a service takes. Services are represented from five perspectives (Figure 2). Basically, the

state perspective is a state chart diagram. The organization perspective identifies both the entity that is charge of wrapping an M-service before it is added to a composite service and the entity that is in charge of submitting the M-service for execution. The information perspective identifies the data that are exchanged between the M-services of a composite service. Finally, the location perspective identifies the current place of an M-service. An M-service is in one the following places: 1) provider site waiting for selection and insertion into a composite service; 2) storage-server waiting for its turn to be submitted for execution; or 3) mobile device under execution. No more than one M-service can run on a mobile device (constraint to be relaxed based on the computing resources of mobile devices).

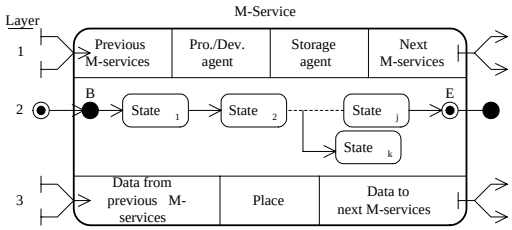

Fig. 2. Service chart diagram of an M-service

Figure 3 reports the details that enrich the service chart diagram of an M-service of Figure 2. These details are associated with layers 1, 2, and 3. Layer 2 is the most interesting; it contains the states that an M-service takes. These states constitute a state chart diagram that will be connected to the state chart diagrams of the M-services of the same composite service. Because a composite service consists of several M-services, the process model underlying that composite service is specified as a state chart diagram whose states are associated with service chart diagrams of the component services, and whose transitions are labelled with events, conditions, and variable assignment operations.

According to the location perspective, an M-service (i.e., instance) is in three exclusive places: provider site, storage server, and mobile device. These places influence the shape and content of the service chart diagram of that M-service. Below, we apply the service chart diagram of Figure 2 to each place.

1. Provider site place - An M-service is in a standby mode waiting to be inserted into a composite service. The provider-delegate takes cares of the insertion after interaction with user-delegates. Figure 4 is the service chart diagram of an M-service in ”provider site” place. In this diagram, all the fields of Figure 2 are filled in with Null, except ”provider-/device-agent” and ”place” fields. In addition, ”standby” is the only state that the M-service takes in ”provider site” place. No actions are undertaken in that state.

Layer	Field	Perspective
1	Previous services	Flow
	Next services	
	Provider -/Device -agent	Organization
	Storage -agent	
2	States	State (normal and ad -hoc operating)
3	Data from previous services	Information
	Data for next services	
	Place	Location

Fig. 3. Fields and perspectives of a service chart diagram

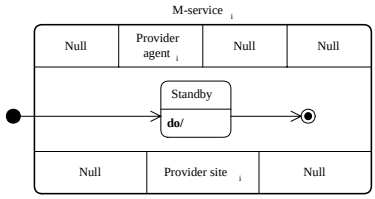

Fig. 4. Service chart diagram of an M-service in provider site place

2. Storage server place - A provider-delegate grants a user-delegate the right to use an M-service. The next step consists of transferring this M-service from the provider site to a storage server. Figure 5 is the service chart diagram of an M-service in ”storage server” place. Figure 6 illustrates how the fields of Figure 3 are instantiated according to that place.

3. Mobile device place - A storage-agent submits an M-service for execution to a user’s mobile device. Figure 7 is the service chart diagram of an M-service in ”mobile device” place. All the fields of Figure 2 are instantiated as indicated by Figure 8. In Figure 7, bold lines represent the links that exist between the M-services of the same composite service; ”in” corresponds to the data that are obtained from M-services. And, ”out” corresponds to the data that are left to the remaining M-services (due for execution after the current M-service is completed). In Figure 7, the dashed line with an arrow represents a wireless transition; an M-service is shipped from a storage server to a mobile device.

Running example - For illustration purposes on the way a composite service is devised, we suggest the following example. Let us assume a composite service CS of 3 component M-services: M-service$_1$, M-service$_2$, and M-service$_3$ (it is assumed that the component services are sequentially executed; concurrent execution is also doable but not discussed in the paper). Nine service chart diagrams are designed; three dia-

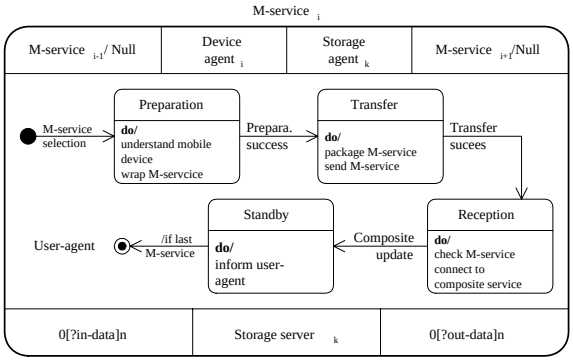

Fig. 5. Service chart diagram of an M-service in storage server place

Field	Value
Previous service	M-service $_{i-1}$
	Null (no predecessor)
Next service	M-service $_{i+1}$
	Null (no successor)
Device -agent	Device -agent $_{i}$
Storage -agent	Storage -agent $_{k}$
States	Preparation, Transfer, Reception, Standby
Data from previous service	0[?i n-data]n
Data for next service	0[?out -data]n
Place	Storage server $_{k}$

Fig. 6. Fields of a service chart diagram of an M-service in storage server place

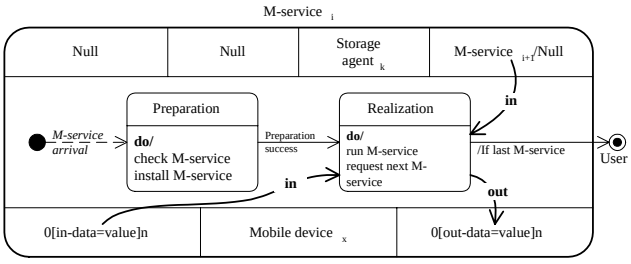

Fig. 7. Service chart diagram of an M-service in mobile device place

Field	Value
Previous service	Null
Next service	M-service $_{i+1}$
	Null (no successor): user
Device -agent	Null
Storage -agent	Storage -agent $_{k}$
States	Preparation, Realization
Data from previous service	0[?in -data=value]n
Data to next service	0[?out -data=val ue]n
Place	Mobile device $_{x}$

Fig. 8. Fields of a service chart diagram of an M-service in mobile device place

grams per an M-service. Figure 9 represents the composite service CS, the M-services that compose that composite service, and the execution chronology. Below each M-service, there are three boxes that constitute the service chart diagrams of that M-service. Letter **a** denotes the service chart diagram in provider site place (Figure 4). Letter **b** denotes the service chart diagram in storage server place (Figure 5). Finally, letter **c** denotes the service chart diagram in mobile device place (Figure 7). Based on the three places, the state chart diagram of the composite service CS combines the service chart diagrams of its component M-services.

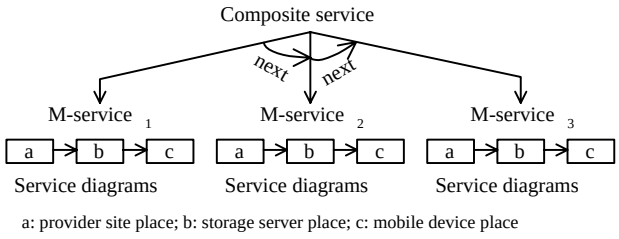

Fig. 9. State chart diagram of a composite service

Figure 10 helps in understanding how the state chart diagram of CS is obtained.

1. At time T, the state chart diagram of CS consists of the service chart diagrams of M-service$_1$ (a), M-service$_2$ (a), and M-service$_3$ (a).
2. At time T+1, the M-services of CS are now identified and their provider-agents and device-agents are asked to transfer them to a specific storage server. The state chart diagram of CS consists of the service chart diagrams of M-service$_1$ (b), M-service$_2$ (b), and M-service$_3$ (b).
3. At time T+2, CS is now established and its M-services are ready for submission to the user's mobile device for execution. Since the M-services are sent one by one, the

state chart diagram of CS consists of the service chart diagrams of M-service$_1$ (c), M-service$_2$ (b), and M-service$_3$ (b).

4. At Time T+3, M-service$_1$ has been executed with success and M-service$_2$ is due for execution. Now, the state chart diagram of CS consists of the service chart diagrams of M-service$_1$ (-), M-service$_2$ (c), and M-service$_3$ (b).
5. At Time T+4, M-service$_2$ has been executed with success and M-service$_3$ is due for execution. Now, the state chart diagram of CS consists of the service chart diagrams of M-service$_1$ (-), M-service$_2$ (-), and M-service$_3$ (c).
6. At Time T+5, the execution of CS is completed.

	Service chart diagram of a composite service CS		
Time	M-service $_1$ Service diagram	M-service $_2$ Service diagram	M-service $_3$ Service diagram
T	a	a	a
T+1	b	b	b
T+2	c	b	b
T+3	-	c	b
T+4	-	-	c
T+5	-	-	-

Fig. 10. Service chart diagrams of a composite service

6 Related work

There exist several research projects that aim at studying how mobile devices can change the way of doing business. In HP Laboratories, Milojicic et al. worked on delivering Internet services to mobile users [9]. This work was titled Pervasive Services Infrastructure (PSI). PSI's vision is "any service to any client (anytime, anywhere)". The project investigated how offloading parts of applications to mid-point servers can enable and enhance service execution on a resource-constrained device. In SAMOS, we are interested in the same issues. Furthermore, we consider how to support users in searching for M-services in an open wireless environment. The MI is part of the searching support that SAMOS offers. In [11], the Ninja project aimed at suggesting new types of robust and scalable distributed Internet services. Ninja's objective is to meet the requirements of an emerging class of extremely heterogeneous devices that would access these services in a transparent way. In Ninja, the architecture considered four elements: bases, units, active proxies, and paths. Similar to a composite service of M-services in SAMOS, a path is an abstraction through which units, services, and active proxies are composed. Proxies are transformational intermediaries put between devices and services to shield them from each other. Ninja proxies are similar to device-agents of SAMOS. Ninja also suggested a Service Discovery Service (SDS) for two reasons: enable services to announce their presence and enable users and programs to locate

these announced services. Comparable to the SDS, the MI in SAMOS has a brokering role. Moreover, the MI facilitates the direct interactions between providers of services and consumers of services, through delegates. In fact, the MI avoids bottleneck situations and ensures a better security to bother providers and consumers.

7 Conclusion and future work

In this paper, we presented the major characteristics of SAMOS in terms of architecture and specification technique of composite services. An overview of the implementation work can be found in [8]. Three major factors should boost the penetration and expansion of M-services in the market: personalization, time-sensitivity, and location-awareness. We are convinced that considering mobile devices as computing platforms will become a reality in the near future. Networks that make these devices reachable are in constant progress, offering more bandwidth and ensuring more reliability and efficiency. For instance, 3G systems with their data-transmission rate up to 384 Kbps for wide-area coverage and 2 Mbps for local-area coverage will provide high quality streamed Internet content. In addition to higher data rates, these systems will be the right support to new value-added services to users such as geographical positioning, user profiling, and mobile payment.

SAMOS is the object of continuing research. For instance, the creation of a user-delegate is currently tasked to the supervisor-agent. The creation of personalized user-delegates according to the profile of users is important. Therefore, our aim is to carry out the creation operation on the user's mobile-device and then, ship the user-delegate to the MI. As with provider-agents, user-delegates have to be checked before getting into the MI. Another initiative is related to security. It consists of enhancing the storage-agent with security mechanisms that will be applied to all the M-services despite their origin (i.e., device-agents) and destination (i.e., user-agents). In addition, specific security procedures that answer each user's priorities have to be done at the level of the mobile device. Since the resources of mobile devices need to be used in a rationale way, the security initiative deals with trust between user-agents and storage-agents. For instance, if a user-agent has had the opportunity to deal with the same storage-agent several times and based on the previous experiences, the user-agent could entrust to this storage-agent its security procedures.

References

1. B. Benatallah, Q. Z. Sheng, and M. Dumas. The Self-Serv Environment for Web Services Composition. *IEEE Internet Computing*, 7(1), January/February 2003.
2. F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana. Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI. *IEEE Internet Computing*, 6(2), March/April 2002.
3. A. Elkadhi, B. Moulin, and Z. Maamar. Vers une Approche d'Évaluation de l'Apport de la Mobilité du Code dans les Environnements Distribués. In *Journées Francophones en Intelligence Artificielle Distribuée et Systèmes Multiagent (JFIADSMA'2001)*, Montreal, Canada, 2001.