A SIP Servlet Framework for Service Provisioning in Stand-Alone Mobile Ad Hoc Networks

Slimane Bah¹, Basel Ahmad¹,², Roch Glitho¹,², Ferhat Khendek¹, Rachida Dssouli¹
{slima_ba, b_ahmad, glitho, khendek, dssouli}@ece.concordia.ca
{basel.ahmad, roch.glitho}@ericsson.com
¹ Concordia University, 1455 de Maisonneuve Blvd. W., Montreal Qc., H3G 1M8, Canada
² Ericsson Canada Inc., 8400 Decarie Blvd., Montreal Qc., H4P 2N2, Canada

Abstract - Mobile ad hoc networks (MANETs) are of primary interest as new networks because of their flexibility. The Session Initiation Protocol (SIP) is the main signaling system in 3G and SIP servlets the main service provisioning framework. SIP is now being considered as a basis for signaling in MANETs. This makes the SIP servlets framework a chief candidate for service provisioning in MANETs. In this paper we propose an SIP servlets framework for provisioning value added services in stand-alone MANETs. This framework consists of a business model and a distribution scheme for the SIP servlet engine. Scenarios and a proof-of-concept prototype are described along with preliminary performance measurements.

Keywords - MANETs; value-added services; SIP; SIP servlets; business model.

1. Introduction
Mobile ad hoc networks (MANETs) are now emerging and are gaining more and more momentum because of their flexibility and easy deployment. They are defined as transient networks formed dynamically by a collection of arbitrarily located wireless mobile nodes without relying on any existing infrastructure or centralized administration [1]. The Session Initiation Protocol (SIP) is the primary signaling protocol for 3G networks and SIP servlets represent the primary service provisioning framework. SIP has been proposed recently for session signaling in 4G both for stand alone MANETs [2], [3] and integrated 3G/MANETs [4]. This makes SIP servlets primary candidates for service provisioning in both stand-alone MANETs and integrated 3G/MANETs. This paper focuses on stand-alone MANETs. It proposes an SIP servlet framework made up of a business model and a distribution scheme for the servlet engine. Business models are critical to service provisioning. They describe the different roles involved in the process of service provisioning and their relationship to each other [5]. The rest of the paper is organized as follows: Section II provides background information on the SIP servlets framework for 3G. Section III presents the proposed business model and distribution scheme for the SIP servlet engine. The proof-of-concept prototype is presented in the fourth section. The fifth section discusses related work, and we conclude in the sixth section.

2. SIP Servlets Framework for 3G
2.1. SIP Servlets and IMS
Session Initiation Protocol (SIP) [6] is a signaling protocol for setting up, modifying and terminating multimedia sessions over IP networks. SIP servlets [7] are java-based applications performing SIP signaling. Servlets interact with SIP clients indirectly through a Servlet Engine (SE). The SE provides the execution environment and manages servlets throughout their lifecycles. The key component of 3G is the IP Multimedia Subsystem (IMS) [9]. IMS is organized in two layers: application and control. The application layer contains application servers (AS) and the control layer includes the core IMS network control nodes (i.e. CSCFs, HSS, MRF and MGW/MGCF). The IMS business model has three roles: the end-user, the service provider and the network operator. The end-user owns User Equipments (UE) --. The service provider owns the application server, as in most deployed IMS systems. The network operator is the owner of the network infrastructure (IMS control nodes). Basically, end-users invoke services through the 3G control nodes while the service providers deploy services in the AS. This later hosts the services, SIP servlets and SIP SE can be located either in a third party or in the network operator domain. The SIP servlets are building blocks that realize services and are executed in the SIP servlet engine. Figure 1 presents an SIP servlet framework with the IMS business model for 3G.

![Figure 1. An SIP servlet-based framework with IMS business model for 3G](image-url)
conference service by forwarding the received REFER message to the AS. Then the Servlets Engine (SE) hosted in the AS selects, loads and runs the appropriate servlet. The servlet ensures that the new participant shares the same interests with the conference members, then generates and sends an INVITE message to the SIP UA3. The RTP streams are sent by the Media Resource Function (MRF). In the rest of the paper we assume a combined AS/MRF entity.

In the rest of the paper we assume a combined AS/MRF streams are sent by the Media Resource Function (MRF). In the rest of the paper we assume a combined AS/MRF entity.

The subsequent messages are generated according to the SIP standard. Before it terminates, the servlet generates and sends a NOTIFY message to inform the UA1 about the progress of the REFER process.  

2.3. Why SIP Servlets for IMS Cannot Work in MANETs?  
SIP servlet business model for IMS as shown in Fig. 1 is not suitable for MANETs. Furthermore, today’s SIP SE implementations cannot be reused as such in MANETs. Indeed, MANETs do not rely on any infrastructure or central entity. Nodes have limited resources in terms of memory, processing, battery power and bandwidth. In addition, nodes have variable configurations and move frequently. These characteristics make the environment highly dynamic, heterogeneous, and with a continuously changing configuration. Service provisioning becomes a challenging task [10]. From the business model point of view, the IMS roles are supplying too many functions that are resource intensive. Indeed, the AS supplied by the service provider contains services/applications, SIP servlets and the SIP SE. The small handheld devices held by a service provider in a MANET cannot host all of these functional entities. Hence, we need a new business model with ‘lighter’ roles. Moreover, we have shown in reference [11] that the existing business models are not suitable for MANETs. IMS business model is not an exception since it relies on central units such as the CSCFs, MRFC and the servlet engine. The location of the functional entities is known beforehand and link characteristics are fairly stable and predictable. The application servers have no resource constraints. Furthermore, the network’s nodes belong to a controllable network where delays, packet loss and disconnections are rare and do not have any impact on the functionality of the SIP servlets framework.

On the other hand, SIP servlets have proven to be a valuable tool in creating and delivering SIP services in traditional networks with fixed infrastructures. With the introduction of MANETs it is worthwhile to extend the use of SIP servlets to provide value-added services in MANETs as well. The SE constitutes a central entity with centralized functionalities. As a consequence, the node hosting the SE should be powerful enough to support a heavy load. Moreover, the only SIP SE implementations available today are parts of a global SIP servlets server (e.g. Oracle Communications Convergent Application Server, WebSphere Application Server, Ubiquity Developer Edition and SIPMethod Application Server). These servers have stringent system requirements for a maximum usage such as a dual 3,6Ghz processor, 4GB of RAM and a Gigabit Ethernet for the Oracle SIP server. This makes MANET nodes unsuitable for hosting the SE as a whole. Therefore, the SIP servlet model for IMS is unsuitable for MANETs. A new business model that takes into account MANET characteristics is required. Moreover, distributing the servlets engine is a promising alternative for using SIP servlets for service provisioning in MANETs.

3. Novel Business Model and Distribution Scheme for the SIP Servlet Engine

3.1. Novel Business Model  
The first requirement on the model is that the functional entities provided by the business roles should not be infrastructure-based since MANETs are infrastructure-less by definition. Second, the business model should be flexible: it should be possible to discover dynamically not only roles, but also functional entities. This will address the dynamic aspect of MANETs where nodes can join and leave anytime. Third, the business model should rely on individuals rather than on organizations. By individuals we refer to anybody present in a MANET at any given time and by organizations we mean business entities such as network operators that own or (have control over) the network. In addition, communication between business roles should be done in a pure peer-to-peer fashion because no central entities are allowed in MANETs. Finally, the mechanism for the publication/discovery of the business roles and the functional entities they provide should take into account MANET characteristics.

We propose a business model with four roles: the End-User (EU), the Service Provider (SP), the SIP Servlets Provider (SSP) and the SIP Servlet Engine Sub-Part Provider (SESPP). The end-user is the entity that makes use of services; the service provider proposes its services to end-users while the SIP servlets provider offers servlets to service providers in order to build and run services. The SIP servlet engine sub-part provider is the owner of a part of the SIP SE function. Due to resource constraints in MANETs, this role will allow MANET nodes to provide a tiny functionality of the SIP SE. Several SIP SE Sub-Part providers should collaborate to offer the overall execution environment for services. The EU role provides the functionality that enables the discovery and the invocation of the value-added services that are available in the MANET at any given time. The SP offers the functionality that maintains and publishes the list of services available in the MANET. Another functional entity is needed to discover any required resources (e.g. servlets, servlets...
3.2. Requirements for Distributing the SIP SE

Building an SIP SE suitable for MANETs requires either designing an adequate SIP SE from scratch (revolution) or distributing the existing one across multiple nodes (evolution). We opt for the evolutionary approach since the current SIP SE is widely used and then the developers are more familiar with it. In addition, by distributing the SIP SE we ensure a backward compatibility. Once distributed, the SIP SE would cease being a heavy-weighted and centralized entity. Instead, its functionalities will be distributed over several nodes that discover each other and establish a scheme to collectively act as an SIP SE. This section presents a set of high-level requirements for a distributed SIP SE in MANETs.

First of all, the components of the distributed SIP SE should be able to run on limited-resource devices. Obviously, the components should have small footprints and should not perform extra processing because of the resource constraints of MANET devices. Second, the communication between the SIP SE components should not introduce too much bandwidth overhead. Third, since a high level of scalability is expected from MANETs, the distributed engine should be scalable. The scalability should be carried out in terms of the number of clients using the distributed SIP SE, the number of requests handled by the distributed SIP SE and the number of servlets deployed in the distributed SIP SE. Fourth, several instances of the same component should be able to co-exist in the same MANET for higher reliability and flexibility. Finally, the components of the distributed engine should not provide overlapping functionality for resources optimization and saving.

3.3. Distribution Scheme

In this paper we only consider the existence of a unique SIP SE. However, the replication of the distributed component (the fourth requirement) is of utmost importance. The replication and the self-organization issues will be addressed in future work. We propose to break up the SIP servlet engine functionality into four independent functional units: the connector, the wrapper, the session repository and the controller. The connector is mainly responsible for receiving, decoding, encoding and sending SIP messages. The wrapper is the entity that loads and runs servlets. The session repository stores application and session information. The controller is the most important component--it handles SIP protocol transactions and orchestrates the other SIP SE components. Consequently, an SIP servlet engine is defined when the four components that constitute it are available in the network.

The motivation behind this distribution is first to make the components as independent as possible so as the replication can be achieved easily. Furthermore, it reduces the coordination between the components for bandwidth saving. The second motivation is to provide a high level of granularity for the distributed functions. It is unsuitable for MANETs to have too many distributed functions closely coupled. From the business model viewpoint, four SIP SE sub-part providers are derived and can be mapped to the SIP SE distributed units. Thus, we define the controller provider, the connector provider, the wrapper provider and the session repository provider. These providers together provide the SIP SE where each part is to be provided by an independent entity. The new business model for service provisioning in MANETs is illustrated in Figure 3.

![Figure 3. The business model for service provisioning in MANETs](image-url)
of servlet, the wrapper can access the session repository to update session information. SIP responses from the wrapper go back to the EUA through the controller and then the connector. Distributing servlet engines brings new challenges. Indeed, the different components have to cooperate to fulfill the SIP SE goal. The communication model between these components should take into account MANET characteristics. In particular it should be light-weighted and should not be bandwidth demanding. Moreover, the communication mechanism should be distributed, peer-to-peer oriented, scalable to the number of involved nodes and allow the asynchronous communication mode. Therefore, we select LIME [12] (LINDA in Mobile Environment) middleware. LIME was chosen because it is distributed, lightweight, written in Java and targeted towards mobile ad-hoc environments with limited resource devices. Furthermore, LIME carries a low overhead because it is based on the tuple space and reactions concepts. Tuples are small messages that contain either values or formals (wildcards). Reactions allow a node to perform some processing in reaction to the presence of a given tuple in the tuple space. Figure 4 presents the communication model of the distributed SIP SE.

Figure 4. The distributed SIP SE communication model

The proposed distribution scheme meets our requirements and takes MANET characteristics into account. The SE functionalities are distributed and no centralized entity exists. The distribution allows small devices to hold and run the distributed functionalities. In fact, in our prototype the controller, the wrapper, the connector and the session repository have, respectively, the following footprint (the jar files): 462KB, 462KB, 417KB and 340 KB. The characteristics of LIME as a communication model between the SE components allow for reducing the overhead. Finally, with several SIP SE components and an adequate mechanism for self-organization and recovery, scalability can be achieved.

4. Validation

4.1. General Description

The interest-based conference service has been implemented twice: with a centralized and with a distributed SIP SE. The conferencing issues in MANETs are solved in [2] and [13] and are out of the scope of this paper. The distributed SIP SE prototype implementation pursues two goals. First, it demonstrates that an SIP SE distributed according to the business model described in previous sections can provide services that a traditional SIP SE currently provides. The distribution of the SIP SE is transparent to the SIP entities that use the services. Second, the prototype aims at establishing that acceptable response times can be achieved with a distributed SIP SE (less than two times the response time of the centralized model). However, this validation is only partial. A more thorough analysis should be conducted, using simulation tools for instance.

As a first step, the selected service is implemented as a full mesh conference. With scalability limitations this scheme can be enhanced using clusters. Creation of an IMS network is not feasible in our lab environment at this time. Instead, a simplified architecture is used, where end-users communicate directly with the SIP SE. We use JAVA as the programming language, JAIN SIP [14] for signaling, Linda In Mobile Environment (LIME) [12] as a middleware for SIP SE components’ message exchange and Apache Tomcat [15] for hosting servlets (Servlets provider). The distributed SIP SE is based on the reference implementation of SIP SE. The EODV ad-hoc routing protocol is used.

We implemented the communication model between the SIP SE components and the interfaces with the nodes representing the other business roles. For inter-node communication we use LIME. Special SIP headers are also used to pass information between the nodes of the distributed SIP SE. Communication with the external nodes is done as follows: SIP protocol is used by the end users to contact the Connector. LIME is used by the Service Provider to issue a "deploy" command to the SIP SE. Finally, the Servlets provider is an HTTP server and, therefore, HTTP is used to download SIP application archives.

4.2. Scenario and Results

Figure 5 shows how the distributed SIP SE handles an initial SIP request. The implementation uses LIME’s primitive Out() and reactions concept for message exchange.

Figure 5. Distributed SIP SE handling an initial SIP request

When the connector receives an SIP request from the EUA it forwards it to the Controller. This triggers a reaction on the controller side. The controller interacts with the session repository to manage the session ID and includes it in an SIP request object passed to the Wrapper, which runs the
appropriate servlet and sends back the response to the EUA.

Three Windows XP laptops with 802.11g adapting cards configured in the ad-hoc mode are used for the distributed scenario. The machines are mobile Pentium 4’s with 512 MB of RAM.

We focus on the conference setup phase. Therefore, the components that do not interact during the conference setup are hosted in the same machine. Thus, the controller and the servlets provider are hosted in one machine. The session repository, the connector and the service provider are hosted in a second machine and the wrapper with the clients is in the third machine. For the centralized SIP SE all the parties run in the same Windows XP machine. It is a 3Ghz Pentium 4 with 1GB of RAM.

The scenario establishes a conference between two clients, and then invites one by one, three other clients to join the ongoing conference. In fact, the scenario contains four phases. The first phase invites two clients A and B, to establish a conference. The second phase invites a third client C. The next phase invites a fourth client D and finally the last phase invites the fifth client E to join the full mesh conference.

A scenario is successful when all its phases are successful. For each phase, the response time from the service provider side is collected. Each scenario was executed ten times and an average of the response times has been calculated. Table 1 summarizes the results for the distributed SE.

Table 1. The average response time and standard deviation of the distributed SE scenario

<table>
<thead>
<tr>
<th>Scenario’s phases</th>
<th>Average response time (sec)</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.75</td>
<td>1.13</td>
</tr>
<tr>
<td>2</td>
<td>6.96</td>
<td>2.45</td>
</tr>
<tr>
<td>3</td>
<td>9.97</td>
<td>3.62</td>
</tr>
<tr>
<td>4</td>
<td>14.05</td>
<td>4.81</td>
</tr>
</tbody>
</table>

The results show a normal behavior: the response time grows with the number of clients. However, the standard deviation is also higher showing that the response time can be sometimes quite lower than the average.

Table 2 presents the results for the centralized SE. We notice that clients are talking directly to the SE instead of going through the IMS core network nodes, which decreases the response time.

Table 2. The average response time and standard deviation of the centralized SE scenario

<table>
<thead>
<tr>
<th>Scenario’s phases</th>
<th>Average response time (sec)</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6.91</td>
<td>0.21</td>
</tr>
<tr>
<td>2</td>
<td>3.57</td>
<td>1.20</td>
</tr>
<tr>
<td>3</td>
<td>5.96</td>
<td>1.00</td>
</tr>
<tr>
<td>4</td>
<td>14.57</td>
<td>3.79</td>
</tr>
</tbody>
</table>

This experiment has shown that the average response time with a distributed SIP SE is 3.39 to 4.84 seconds longer than the average response time with a centralized SIP SE for the three first phases. Except the first phase where initializations take place, the response time in the distributed scheme is less than two times the one in the centralized scheme. The additional time is due to the messages exchanged between the distributed SIP SE components, but we expected that running each component in a separate node and using clusters rather than a full mesh scheme for the conference would result in a shorter response time.

5. Related Work

Reference [11] pinpoints the drawbacks of the ‘classical’ business models and proposes a new model for MANETs. It is made up of four roles: the end-user, the service provider, the capabilities provider and the execution environment provider. The end-user uses the services offered by the service providers. While the capabilities provider makes available building blocks for service creation the execution environment provider offers an environment for service execution.

In an SIP servlets setting these roles can be mapped to end-user, service provider, SIP servlets provider and SIP servlets engine provider, respectively. Nevertheless, the business model assumes that the execution environment runs on one node. In MANETs, devices are usually of a small size with very limited capabilities and features. The business model proposed in this paper is a refinement of the business model discussed in [11]. Indeed, the execution environment provider is now decomposed to several execution environment sub-part providers.

The Java Servlet specification [8] allows the servlet engine to be distributed across several Java Virtual Machines (JVM). JVMs may be running on the same host or on several remote hosts. In reference [16] session management is separated from servlet hosting. Then, multiple hosts running instances of the same servlet may access the session management point to retrieve or modify session information. Two schemes are proposed for this distribution. However, both approaches assume the presence of a central entity, which is unrealistic in MANETs. Furthermore, the goal behind this distribution is load balancing with no consideration for MANETs’ characteristics.

6. Conclusion

This paper proposed an SIP servlets-based framework for service provisioning in stand alone MANETs. It defines a novel business model and describes the roles and their interactions. We have proposed a scheme for distributing the SIP servlet engine. The engine is decomposed into four components to be easily deployed in MANETs. A mapping between the distributed SIP SE components and the business roles has been presented. Finally, a prototype as a proof-of-concept for both the business model and SIP servlet engine distribution has been developed, and a partial validation has been conducted and discussed.

7. References

[2]. C. Fu, R. Giltho and R. Dssouli, A Novel Signaling System for Multiparty Sessions in Peer-to-Peer Ad Hoc Networks,


