# On Accessing GSM-enabled Mobile Sensors

Zissis K. Plitsis, # Ioannis Fudos, Evaggelia Pitoura and Apostolos Zarras
*Department of Computer Science, University of Ioannina, Greece*
*{zplitsis, fudos, pitoura, zarras}@cs.uoi.gr*

**Abstract**

*In this paper, we propose a middleware framework that unifies access to GSM-enabled sensor devices in a global computing environment. Typically, communication with mobile sensors relies on proprietary protocols, involving the exchange of SMS and MMS messages. In the proposed framework, we use XML-based control descriptions that abstractly specify these protocols to generate proxies and corresponding WEB-based (HTML, WAP and WEB services) interfaces that realize them. Thus, we provide access transparency over different kinds of mobile sensors. Besides the overall architecture of the proposed framework, we discuss a particular instance where a GSM-enabled camera with temperature, and motion detection sensors is incorporated into our global computing environment. Finally, we assess the performance of the proposed framework by presenting experimental results.*

## 1. Introduction

The World-Wide Web has evolved into the major data structure for providing and accessing computer applications and other resources though well-defined WEB-enabled interfaces. Several emerging technologies exist for the development of such interfaces. In practice, we meet HTML-based interfaces that facilitate the communication between devices such as personal computers and laptops and WAP-based interfaces that support the communication in environments involving handheld devices such as PDAs and pocket PCs. Nowadays, we further have the ability to use programmable interfaces, relying on the standard WEB Services architecture [1], [2].

In this paper, we specifically focus on incorporating in such global computing environments [3] small GSM-enabled sensor devices, controlled by SMS messages. Typically, information gathering from mobile sensors is performed through either SMS messages (e.g. temperature, atmospheric pressure or humidity) or MMS messages (e.g. images, video or time varying signals of seismic or electromagnetic activity). SMS messages are traditionally used as means for controlling GSM-enabled devices and for logging data regarding their operation. A sensor-specific proxy server collects client requests for information and submits them to the sensor. Then, it collects the specified information and makes it available in client-compatible formats. The interaction between the proxy server and the mobile sensor is determined by the manufacturer's specifications regarding command sequences for initializing the sensor and for selecting amongst alternative delivery methods and data contents.

Hence, the initialization and the gathering of information provided by mobile sensors varies depending on the type of the sensor. In principle, a global computing environment, such as the ones we examine [3], shall comprise many different types of sensors. Consequently, the aim of this paper is to propose *a middleware framework that enables a uniform WEB-based access to mobile sensors*. To this end, each mobile sensor is accompanied by a description called *Mobile Sensor Control Description (MSCD)* that serves as input to the proposed framework. Based on the MSCD, we generate sensor-specific proxy servers and corresponding WEB interfaces. The generated proxy servers realize the necessary procedures for the sensor initialization and the gathering of information according to several sensor-specific parameters that can be customized by the clients through the WEB interfaces. The clients may use different devices such as personal computers, laptops or PDAs with Internet access. Depending on the client preferences, the sensor-acquired information may be delivered to an e-mail address, to a mobile phone or to a WEB page. In a sense, the proposed middleware framework is *reflective* [4] since it self-customizes its interfaces with respect to constraints imposed by each particular sensor that participates in the global computing environment.

To demonstrate our overall approach for unifying access to mobile sensors in global computing environments, we provide a specific instance of our architecture that allows accessing a mobile camera through multiple WEB-based interfaces. We report on various issues regarding the implementation of our framework and present related performance results.

The remaining of the paper is structured as follows. Section 2 discusses related work. Section 3, details the overall framework, while Section 4, focuses on the implementation of a prototype application that utilizes the framework. Section 5, presents a number of performance results. Finally, Section 6 concludes the paper.

## 2. Related Work

The work presented in this paper generally relates to the integration of devices that communicate through SMS and MMS in WEB-based global computing environments. Short Messaging Services or Short Message Sending (SMS) [5] is widely supported in mobile phones in most countries. It allows users to compose short textual messages using the telephone handset, and transmit them asynchronously. Thus, it is natural to bind together the pertinent telephony and computing protocols so that computers can originate and perhaps receive such messages. In that respect Short Messaging Services are

0-7803-9399-6/05/$20.00  

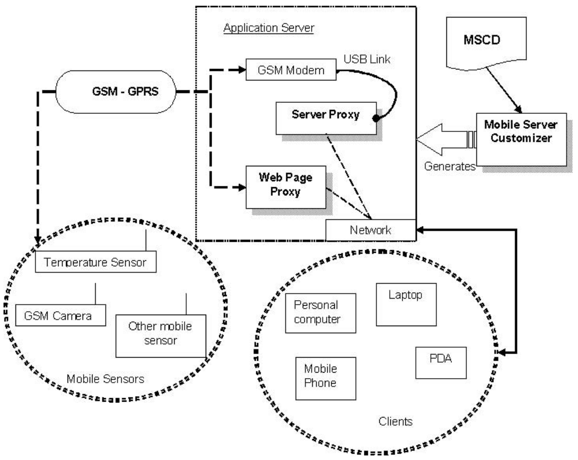



**Fig. 1:** System Architecture

offered by various cellular telephony providers through WEB interfaces.

In general, XML has been used for sending SMS messages over HTTP [6]. However, each vendor created its own implementation leading to interoperability problems. To solve such problems the SMS Forum [7] developed two relating standards: Short Messaging Application Part (SMAP), an XML format for the messages themselves, and Mobile Messaging Access Protocol (MMAP), a SOAP-based protocol for sending those messages. Simple Object Access Protocol (SOAP) is a simple XML protocol for exchanging structured information over the Internet and is amongst the core standards that formulate the overall Web Services architecture [1]. SOAP lies on top of a variety of transport protocols such as HTTP and SMTP.

The aforementioned standards constitute a foundation for communicating with mobile sensors using SOAP. An approach that actually realizes such communication capabilities is detailed in [8]. In particular, the authors propose a bi-directional SOAP/SMS gateway service. This approach bears some similarity with our framework. The gateway service gets SOAP requests from the client application, makes use of a database and a GSM modem to access mobile sensors and sends SOAP responses. Implementation-wise there are several common points between this approach and our framework. However, a major difference is that our approach unifies access to different types of mobile sensors through WEB-based interfaces generated automatically. The implementations of these interfaces translate client requests to sensor-specific sequences of SMS control messages. Our system further provides compatibility with approaches for accessing mobile devices through WAP [9]. WAP allows low-end devices with limited CPU power, memory and storage to access the wireless WEB.

As discussed in the introduction the proposed middleware framework is reflective as it self-customizes its interfaces with respect to constraints imposed by each particular sensor in the global computing environment. There are several middleware frameworks that expose the properties provided by the middleware services for introspection and change [10]. To our knowledge, none of the aforementioned particularly deals with the provision of WEB-based access transparency over mobile sensors.

## 3. System Architecture

An overview of our architecture is illustrated in Figure 1. The global computing environment we consider comprises clients, using different WEB-enabled devices such as personal computers, laptops and PDAs to access available resources. Mobile sensors communicating through GSM and GPRS are a particular kind of such resources. Our framework consists of three main components, namely a *mobile sensor customizer*, and different kinds of *server* and *WEB page proxies*. The server and the WEB page proxies are sensor-specific and establish communication between the clients and the sensors. On the other hand, the mobile sensor customizer serves for generating the aforementioned sensor-specific components, given the specification of *Mobile Sensor Control Descriptions (MSCDs)*. The rest of this section further discusses the main responsibilities of the components that constitute the proposed framework.

### A. Mobile Sensor Customizer

Currently, the interaction between clients and mobile sensors is determined by the manufacturers' specifications regarding command sequences for initializing a sensor, and for selecting delivery methods and data contents. Unifying the communication between clients and mobile sensors by providing appropriate WEB-based interfaces is a major issue in this context. Addressing this issue is the main responsibility of the sensor customizer. The customizer accepts as input an MSCD, provided by means of an XML file. Roughly, the MSCD specifies the type of information that can be delivered by the sensor and alternative delivery methods.

Following, the customizer generates appropriate WEB-based interfaces and corresponding implementations of server and WEB page proxies that mediate the interaction between clients and mobile servers. Different kinds of sensors have different descriptions and capabilities and so the behavior of the server and the WEB page specific proxies can vary. For instance, let us assume that a mobile sensor can send image, temperature or both, and this information can be delivered with an SMS or an MMS. The SMS control sequences that perform these operations on the mobile sensor is the information that the customizer wants to acquire from the MSCD, to generate a server proxy that actually realizes the operations which are exported by the server proxy in terms of a well-defined WEB interface. Specifically, the mobile sensor customizer we consider supports the generation of two different types of server proxies: (1) servlets providing HTML or WAP based interfaces, and (2) Web Services, providing WSDL compliant interfaces.

In detail, the MSCD of a mobile sensor consists of the following elements:

```xml
<?xml version="1.0" encoding="UTF-8" ?>
- <sensor xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="C:\Documents and
    Settings\zkp\Desktop\sensor.xsd" sensorName="Mobile Camera"
    sensorPhone='69471234...'>
    <name>sensorName</name>
    <phNumber>sensorPhone</phNumber>
  - <init>
    - <initialization>
      - <message>
        - <messageType>
          - <SMS>
              <body>" 25 " sensorName</body>
              <order />
            </SMS>
          </messageType>
          <receiver />
          <sender />
          <description>"Set a name for the camera"</description>
        </message>
        <type />
      - <delivery>
        - <message>
          - <messageType>
            - <SMS>
                <body>"ok"</body>
                <order />
              </SMS>
            </messageType>
            <receiver />
            <sender />
            <description>"receive ok"</description>
          </message>
          <description />
        </delivery>
    ...
```

```xml
+ <init>
- <info>
  - <infoType>
    - <message>
      - <messageType>
        - <SMS>
            <body>"1"</body>
            <order>"0"</order>
          </SMS>
        </messageType>
        <receiver />
        <sender />
        <description>"get image and temperature"</description>
      </message>
      <type />
    - <subTypes>
      - <subType>
        - <message>
          - <messageType>
            - <SMS>
                <body>"11 1"</body>
                <order>"1"</order>
              </SMS>
            </messageType>
            <receiver />
            <sender />
            <description>"change resolution to high"</description>
          </message>
          <order>"1"</order>
          <type />
        + <delivery>
        </subType>
      + <subType>
      + <defaultTypes>
      </subTypes>
    ...
```

**Fig. 2:** Example of a MSCD specification for a mobile camera.

- *Initialization information* (`init` tag in Figure 2), consisting of a set of alternative initialization protocols for the mobile sensor. An initialization protocol specifies an ordered collection of request and response messages that must be exchanged between the proxy server and the sensor towards the sensor's initialization.
- *Query delivery information* (`info` tag in Figure 2), comprising a set of alternative query protocols for the mobile sensor. A query protocol prescribes an ordered collection of request and response messages that must be exchanged between the proxy server and the sensor to obtain the information provided by the sensor.

The initialization and the query protocols customize the content type provided by the mobile sensor and several other content-dependent quality attributes that specify characteristics of the data type that will be delivered (`info` tag in Figure 2). For instance, the content types may be image, video or text and the attributes may specify characteristics such as image resolution, video compression or image format. The WEB interfaces generated by the customizer facilitate the selection between alternative initialization and query protocols, as they allow the clients to set their preferences regarding the various content types and attributes either graphically through HTML or WAP based pages, or though a programmable WSDL interface. Then, the client preferences are properly handled by the corresponding proxy servers.

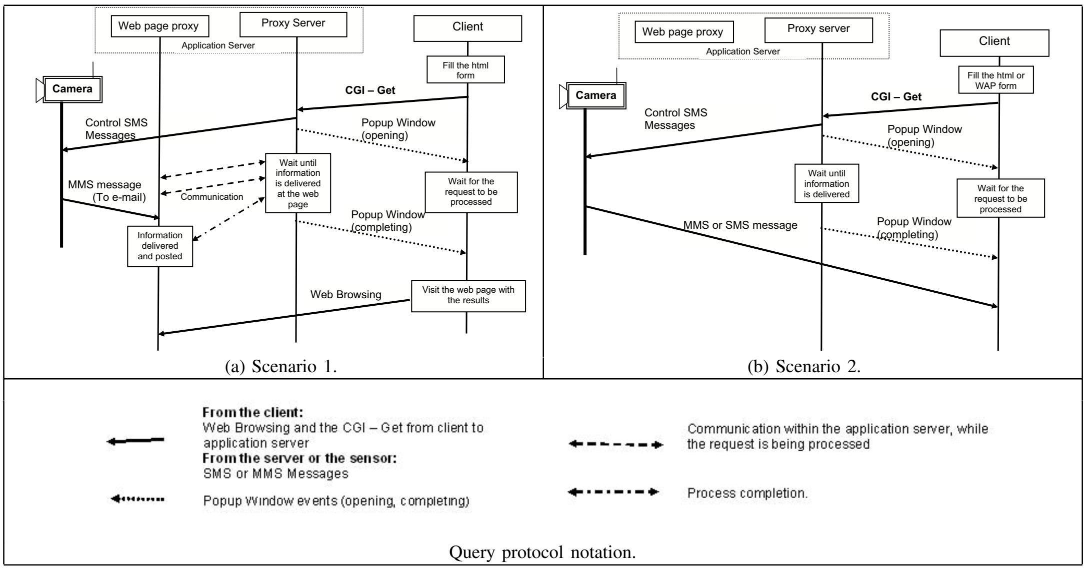
Hence, to integrate a mobile sensor in our global computing environment we define an XML scheme that describes the structure of MSCDs. Due to space limitations, the detailed description of the scheme can be found in the long version of this paper [11]. We can then describe all mobile sensors by providing MSCDs that comply to this scheme. A representative MSCD example is given in Figure 2, which is further detailed in Section 4.

### *B. Server and WEB Page Proxies*

The behavior of proxy servers materializes the alternative initialization and query protocols, specified in the MSCDs that were used for generating the servers. In particular, a proxy server collects requests for information issued by clients and translates them into sequences of sensor-specific requests such as SMS messages. Following, the proxy server receives the specified information and makes it available in client-compatible formats. The proxy server uses GSM to communicate with the mobile sensor and the mobile sensor responds by submitting appropriate SMS or MMS messages using GSM or GPRS, respectively. In our approach it is important to deal with the common scenario where a client executes on a low-end device with limited power, processing and storage capabilities. Such kind of devices may not efficiently support the reception of MMS messages. In this case our framework provides the option of building a WEB page that contains the results obtained by the sensor. The construction of this WEB page is a responsibility of the WEB page proxy component, which receives the MMS sent by the sensor in place of the client. The WEB page is created upon the arrival of the email message that contains the MMS built by the sensor. Synchronizing the client and the WEB page proxy is an issue, tackled by the proxy server. During the processing of a client request the proxy server waits for the creation of the result page at the WEB page proxy and then notifies the client. The proxy server uses polling to realize the previous task. While the client request is being processed a popup window is open at the client's browser, highlighting the progress of the client's request.

## 4. A Framework Instance

The application described in this section uses mobile sensors, which send and receive SMS messages through the GSM




(a) Scenario 1. (b) Scenario 2.

Query protocol notation.

**Fig. 3:** Query protocols for the two scenarios.

/GPRS network. The messages concern querying of information and controlling delivery parameters. The delivery is realized through either SMS or MMS messages. To provide a unified WEB-based interface for these sensors we use MSCD specifications as input to the mobile sensor customizer of the proposed framework. Specifically, Figure 2 gives the MSCD information for a mobile camera. The mobile camera is a stand-alone remote GSM-GPRS device with imaging hardware, motion detector, thermometer and microphone. The mobile camera needs no internet connection and it can be installed in any place where there is GSM coverage. The camera may send any of the information through an MMS or SMS message responding to a sequence of SMS control messages. Accounts can be created for users of the camera specifying e-mail or phone numbers. A master user is set with control privileges. For every account there are choices for the content of information and the type of delivery. Based on the MSCD, we generate a server and a WEB page proxy and corresponding HTML and WAP based interfaces.

The clients of our application may then execute several query scenarios involving information provided by the mobile camera simply through the use of the generated interfaces and without any particular knowledge of technicalities that relate to the particular camera. All the required expertise on using the mobile camera is encapsulated in the logic of the server and the WEB page proxies, generated by the mobile sensor customizer. Following we examine two possible scenarios which are further evaluated in Section 5:

1) A client uses the HTML interface of the camera to obtain image and temperature, delivered through a new page.
2) A client uses the WAP interface to acquire image and temperature, delivered through an e-mail message.

To realize the first scenario, the client has to fill up the options of the HTML forms given in Figure 4. In particular, the scenario proceeds as follows:

1) The client selects "Image and Temperature", a resolution and "To web page" in the delivery options (Figure 4(a), (b)).
2) After submitting the query a popup window appears and displays date and time asking the user to wait. At the same time the server proxy sends the client request to the camera and waits until the results web page is created (Figure 4(c)).
3) The camera receives the SMS message that encapsulates the client request and sends image and temperature data to the WEB page proxy through an MMS message.
4) Upon the reception of the MMS, the WEB page proxy uses a script to extract the data and creates the results page (Figure 4(d)).
5) After polling the WEB page proxy, the server proxy gets the notification that the results page is ready. Following, the server updates the popup window with the final form that displays the link to the results page.

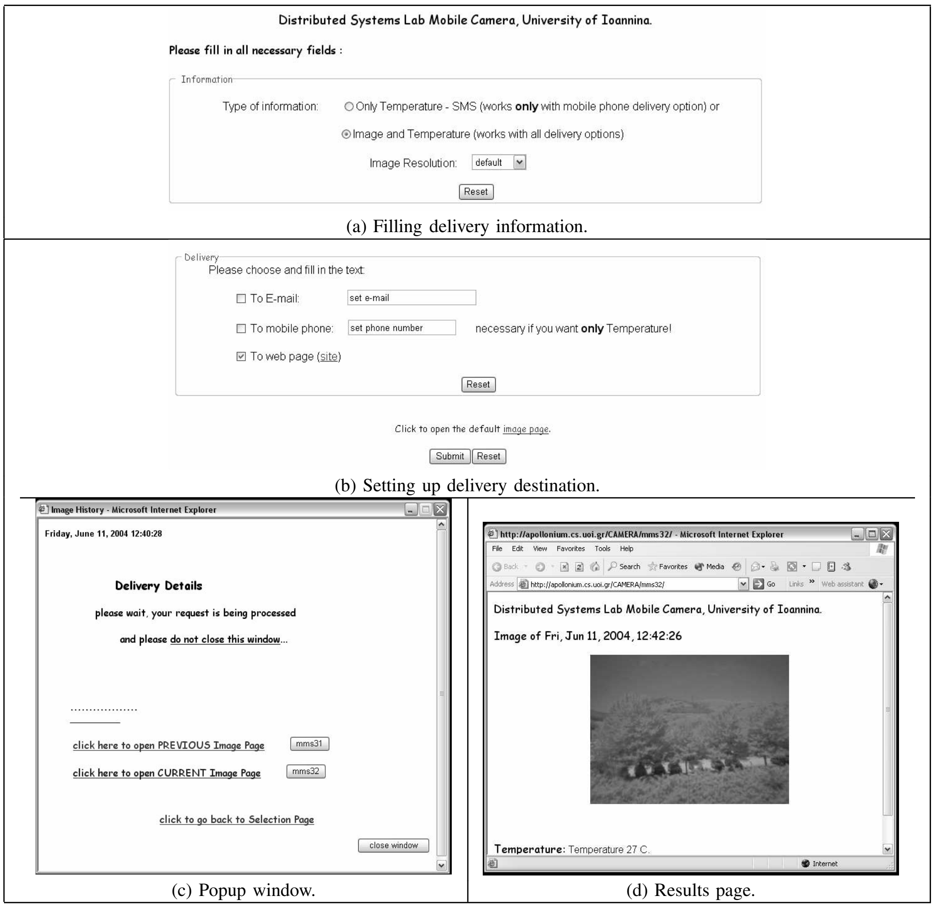
Similarly, to realize the second scenario the client has to use a generated WAP interface. Figure 3 gives further details regarding the query protocols executed during each of the scenarios. The query protocols and the WAP interface are not further discussed due to space limitations. More details can be found in the long version of this paper [11]. [1]

## 5. Performance Evaluation

We performed experiments for determining the average response time for the framework instance of Section 4 for common query requests in various configurations. The query experiments performed were the following:

(i) Requesting image and temperature with image resolution (a) default (b) high and (c) compact. The request is issued

[1]The interested reader may also test further scenarios involving the mobile camera and the HTML or WAP interfaces at `http://sensor-proxy.cs.uoi.gr/index_ds.htm` or `wap://sensor-proxy.cs.uoi.gr/index_ds.wml`, respectively.



(a) Filling delivery information.
(b) Setting up delivery destination.
(c) Popup window.
(d) Results page.

**Fig. 4:** Using the HTML interface in the first scenario.

**TABLE 1:** RESPONSE TIME AND MIDDLEWARE OVERHEAD FOR EXPERIMENT (I).

| Type of experiment | Average preparation time for sending message(s) (in sec) | Overall response time average (in sec) |
|---|---|---|
| Image with default resolution and temperature at web page | 31.6 | 64.5 |
| Image with high resolution and temperature at web page (two SMS messages) | 19.0 (1st SMS) 48.5 (2nd SMS) | 132.6 |
| Image with compact resolution and temperature at web page (two SMS messages) | 23.1 (1st SMS) 35.4 (2nd SMS) | 96.2 |

**TABLE 2:** RESPONSE TIME AND MIDDLEWARE OVERHEAD FOR EXPERIMENT (II).

| Type of experiment | Average preparation time for message sending (in sec) | Overall response time average (in sec) |
|---|---|---|
| Getting temperature at the mobile phone | 30.1 | 77.6 |

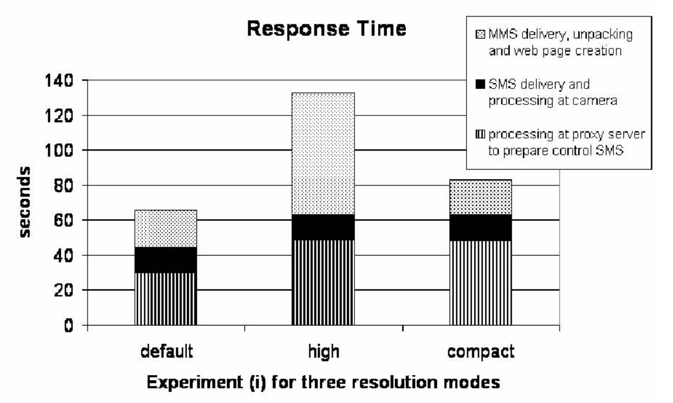
through the HTML-based interface and the results are delivered on a WEB page

(ii) Requesting temperature only through the HTML-based interface. The results are delivered by SMS to a mobile phone (note that even in that case a WEB page is created).

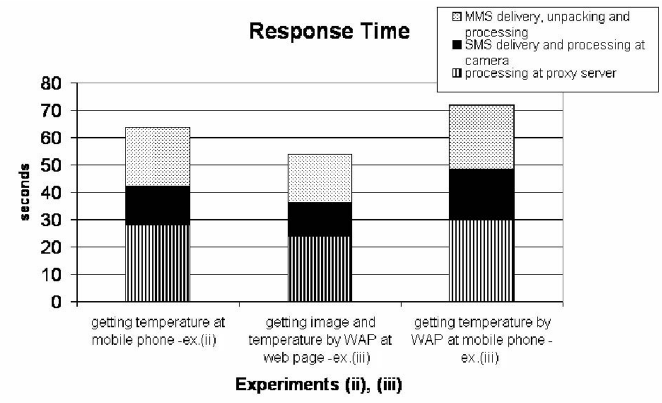
(iii) The queries of experiment (i) and (ii) submitted through the WAP-based interface.

Specifically, for the above scenarios we measured the average preparation time required by the server proxy for sending the SMS messages to the camera and the overall response time (measured from the moment that the user presses the submit button in the HTML or the WAP form, until the moment that he/she receives the corresponding results). Roughly, the average preparation time is the overhead introduced by the proposed middleware framework.

For the first experiment the results are shown in Table 1. For the default resolution the query protocol comprises a single SMS request message. The average response time was 64.5 sec with a standard deviation of 8.5 sec. When the resolution is set to high or compact an additional SMS request message is required in the query protocol so as to appropriately set up the corresponding quality attribute of the mobile camera. The




(a) Breakdown of overall response time for exp.(i)




(b) Breakdown of overall response time for exp.(ii),(iii)

**Fig. 5:** Results summary.

**TABLE 3:** RESPONSE TIME AND MIDDLEWARE OVERHEAD FOR EXPERIMENT (III).

| Type of experiments | Average response time for wap access (in sec) |
|---|---|
| Getting image and temperature at the web page | 42.5 |
| Getting temperature at a mobile phone | 74 |

need for this additional message almost doubles the overall response time.

For the second experiment the results are illustrated in Table 2. In this case, a results page is created with image and temperature info. Subsequently, the temperature info is extracted from the page and an SMS message is sent to the user's mobile phone with the temperature info only. The average time is 77.5 sec with 11.8 sec standard deviation.

Finally, Table 3 presents the results for the last experiment, i.e. accessing the camera from the WAP interface to get (a) image and temperature (with default resolution), delivered in a WEB page and (b) just temperature, delivered to a mobile phone.

Figure 5(a) and (b) summarize our results. Specifically, the overall response time is divided into the time required for the preparation of the SMS request messages at the proxy server (vertical lines), the time required for sending those requests to the camera (solid black) and the time required for the preparation and the delivery of the MMS reply from the camera to the client or to the WEB page proxy (gray). Observe that the processing time introduced by our framework at the proxy server is almost the same in every experiment. The remaining overhead depends on the network latency. The large standard deviation is due to the GSM/GPRS network traffic and communication parameters. In the case of multiple user requests at the same proxy server the response time could increase significantly. To resolve this bottleneck we may use more than one proxy servers and/or multiple sensors at the same point.

## 6. CONCLUSIONS

The main contribution of this paper is a middleware framework that provides WEB-based access transparency over different kinds of mobile sensors. Specifically, we use XML-based control descriptions to abstractly specify proprietary SMS/MMS-based communication protocols assumed by mobile sensors. Based on these specifications we generate proxies and corresponding WEB-based (HTML, WAP and WEB services) interfaces that realize the aforementioned protocols. In the paper we detailed the architecture of the proposed framework and we demonstrated its use in the particular case of a GSM-enabled camera with temperature, and motion detection sensors. Finally, we evaluated our framework and presented related performance results.

## 7. ACKNOWLEDGMENTS

This work is partial funded by a General Secretary of research and Technology grant for bilateral research cooperation between Greece and Australia (DECA).

## REFERENCES

[1] W3C. Web Services Architecture. Technical report, The World Wide Web Consortium (W3C), 2004. http://www.w3.org/TR/ws-arch/.

[2] C. Laird. SMS: Case study of a Web Services Deployment. developerWorks$_{(R)}$, IBM's resource for developers, August 2001. http://www-106.ibm.com/developerworks/webservices/library/ws-sms.html.

[3] E. Pitoura et. al. DBGlobe: a Service-Oriented P2P System for Global Computing. *SIGMOD Record*, 32(3):77–82, 2003.

[4] J. Malenfant, M. Jacques, and F.N. Demers. A Tutorial on Behavioral Reflection and its Implementation. In *Proceedings of REFLECTION '96*, 1996.

[5] GSM World . GSM - SMS Overview, a technical overview of SMS. The World Wide Web Site of the GSM Association, 2005. http://www.gsmworld.com/technology/sms/index.shtml.

[6] N. Chase. Tip: Use XML to Send SMS Messages; Reap the Benefits of MMAP and SMAP. developerWorks$_{(R)}$, IBM's resource for developers, 2004. http://www-106.ibm.com/developerworks/xml/library/x-tipsms1.html.

[7] The SOAP Forum. http://www.smsforum.net/.

[8] G. A. Halse and G. Wells. A Bi-directional SOAP/SMS Gateway Service. In *Proceedings of SATNAC'02*, 2002.

[9] V. Kumar, et. al. WAP: Present and Future. *Pervasive Computing*, 2(1), 2003.

[10] G.S. Blair, G. Goulson, and M. Papathomas. An Architecture for Next Generation Middleware. In *Proceedings of MIDDLEWARE '98*, 1998.

[11] Z. K. Plitsis, I. Fudos, E. Pitoura and A. Zarras. On Accessing GSM-enabled Mobile Sensors, 2005. http://softsys.cs.uoi.gr/dbglobe/publications/sensors_new.pdf.