Comparative Performance Evaluation of Web Services and JXTA for Embedded Environment Monitoring Systems

Jens Schmutzler, Andreas Wolff, Christian Wietfeld
Dortmund University of Technology
Communication Networks Institute
Otto-Hahn-Str. 6, 44227 Dortmund, Germany
{jens.schmutzler,andreas.wolff,christian.wietfeld}@tu-dortmund.de
http://www.cni.tu-dortmund.de

Abstract

Today’s broad coverage of mobile networks allows for large-scale environmental monitoring systems capable of warning citizens prior to an incident. In this paper, we present an embedded prototype system which collects environmental data and distributes them either using Web Service protocols or using the JXTA Peer-to-Peer protocol stack. The complete system was implemented based on both protocol stacks in order to prove their applicability for a middleware solution in the field of sensor networks. The performance of both stacks is analyzed, utilizing an embedded sensor gateway node with a wireless GPRS connection. In case of the Web Service approach standard XML-based SOAP messages as well as WAP Binary XML (WBXML) based SOAP messages are examined and compared against the JXTA Peer-to-Peer protocol. Both the processing power of the embedded sensor gateway node and the throughput of the GPRS-based wireless communication channel are the major bottlenecks focused in this performance evaluation. In our scenario WBXML is the favorable approach for small sets of data being transmitted over the wireless link whereas JXTA is the favorable approach for large sets of data.

1. Introduction

Recent developments in mobile computing and sensor networks allow for continuous monitoring solutions in multiple application areas. Autonomous embedded systems with small dimensions and dedicated sensor nodes are deployed in various areas of interest, like homeland security or pervasive healthcare. In this case embedded systems are utilized for surveillance applications or to create autonomous reliable networks [1]. Other application areas are environmental monitoring solutions which are the focus of this work. We created a testbed which allows us to add location information to measurement data sets from sensor nodes offering the functionality of a Geographical Information System (GIS). This application serves as a simplified demonstrator of complex environmental monitoring systems.

Given this scenario the idea of this work is to evaluate what kind of communication protocol stack offers a favorable approach for a middleware solution in this application area. The major focus of this work is the performance analysis of two variations of SOAP-based Web Services and JXTA communication protocols with respect to the given scenario.

Current middleware solutions are typically based on Web Services, which utilize the XML-based SOAP protocol for communication. But human readable XML has a bad degree of efficiency in terms of information per byte. In case of full blown Internet servers connected to high bandwidth networks this disadvantage is negligible, but it has a high impact on embedded systems and wireless communication links. To counter the overhead problem of XML, W3C specified the WAP binary XML (WBXML) [2] which is a binary representation of XML and therefore much more efficient with respect to the information per byte ratio. Both standard XML and WBXML are used for the performance evaluation of Web Services.

The system was also implemented on the basis of the Peer-to-Peer protocol JXTA and its embedded device binding JXTA Micro Edition (JXME) [3]. The JXME implementation utilizes a proxy service between the embedded device and the JXTA network, providing means to optimize the communication protocol and
processing tasks towards the embedded platform and at the same time enabling it to be perceived as full participant of the Peer-to-Peer network from the JXTA perspective.

This work respects both the processing constraints on the embedded platform as well as the energy constrains these systems are exposed to in deserted areas. They only provide limited communication means and therefore depend on efficient communication protocols. The system was designed considering both communication efficiency and interface usage efficiency in order to keep the overall power consumption of the embedded platform as low as possible. The evaluation was performed on public and simulated networks with a focus on the protocol stacks and not on the GPRS communication channel.

2. Environmental Monitoring Scenario

Increasing severity and numbers of natural disasters [4] have proven the importance of environmental monitoring and early-warning systems. In case of an alarm they help to reduce the time needed to inform and warn the concerned group of citizens. Typically early-warning systems continuously monitor a predefined area with the help of geographically distributed sensor nodes. Depending on the type of expected alarms for a given area of interest different kinds of sensors can be deployed. For our demonstrator we chose an exemplary sensor which continuously measures the temperature and both oxygen and methane content of the ambient air.

The system design corresponds to the use case diagram shown in figure 2. It conforms to generic monitoring purposes and can be easily adopted in previously mentioned application areas like pervasive healthcare or homeland security. A client (e.g. environmental authority) can subscribe to measurement or alarm events or explicitly request new measurement values. The sensor nodes on the other side push new measurements into the system which subsequently identifies alarming values in order to trigger alarm events or normal measurement events, respectively. Since the performance evaluation of the communication protocols with respect to the applicability for sensor network tailored middleware solutions is the major focus in this work, the system design was kept rather simple. [5] describes other systems which respect more complex use case scenarios in the context of environmental monitoring.
3. System Architecture

This section will describe the monitoring system architecture in more detail with respect to some technical aspects relevant to the performance evaluation. Besides the sensors, there are three functional entities involved in the system which are explained in detail in this section. Figure 1 gives an overview of the entire testbed incorporating the standard XML-based and WBXML-based SOAP approach and the JXTA Peer-to-Peer approach. Both Web Service implementations differ in the representation of the data on the wireless link. The WBXML proxy converts the binary data representation of WBXML into standard XML and vice versa. Hence, the Web Services themselves are not influenced by the protocol differences but the representation of data on the wireless link is different.

The architecture of the JXTA Peer-to-Peer protocol involves a much more complex proxy service which not only converts the data representation but offers a complete JXTA protocol binding for very limited JXME peers [3]. Additionally the JXTA network incorporates Rendezvous peers which are the infrastructure peers of the JXTA network. Each peer in the JXTA network registers with a Rendezvous peer by sending its peer and service advertisements to the Rendezvous peer. The Rendezvous peers organize all advertisement indices in a loosely coupled Distributed Hash Table (DHT). A much more detailed explanation of the JXTA protocol can be found in [6]. The embedded wireless gateway implements all three protocols.

3.1. Embedded Sensor Gateway

The embedded sensor gateway is based on the Siemens Wireless Module XT75. It offers quad-band GSM with support for GPRS class 12 and EDGE class 10. The platform is based on an ARM CPU with 400kB RAM and 1.2MB flash memory. For application development it is compliant to the Java Micro Edition (J2ME) Information Module Profile - Next Generation (IMP-NG) which is part of the Connected Limited Device Configuration (CLDC) of the J2ME specification. It offers numerous interfaces like a USB 2.0, Serial Port, Serial Peripheral Interface Bus, Inter-Integrated Circuit (I2C), multiple General Purpose Input/Output connectors, and many more for connecting various types of sensors. The gas sensor in our scenario is connected through a serial link to the XT75. All measurement data are transmitted through the GPRS link. In order to map the measurement values to an exact geographic location, the GPS functionality of the XT75 is used. Furthermore it supports TCP/IP and HTTP what makes it an ideal embedded platform for testing the Web Service and JXTA protocols. The XT75 offers a lot of other features which would be beneficial for early-warning systems (e.g. SMS Cell Broadcast) but these were further investigated due to the focus on the communication protocol stacks.

In order to optimize the energy efficiency of the device the GPRS interface is scheduled depending on the time interval between measurement transmissions. Hence, the measurement values are not necessarily sent one after another to the server but cached on the device’s memory and sent in sets of multiple measurements as long as no alarming measurement value is identified. The service on the embedded sensor gateway allows for defining several threshold values which conform to certain degrees of alarm situations. In case a measurement exceeds such threshold values all currently stored measurements and the critical measurement value are directly transmitted to the server. Additionally to the instant transmission on alarm events a maximum time interval between measurement readings can be defined. Due to the limited system memory of the XT75, 64 sensor measurement values were chosen as the maximum number of measurements being sent as one chunk.

3.2. Server-side data collection

The embedded sensor gateway connects and sends all measurement values to the server side of the system. The server primarily acts as a database collecting all sensor information and storing it into a database for later reference. It manages user subscriptions and forwards all sensor information to the list of subscribers and generates alarm messages if applicable. The server exposes interfaces for both JXTA and Web Services to the embedded sensor gateway as well as the client side of the system.

3.3. Geographic Information System Client

The client offers a graphical front-end to the Geographic Information System. The application provides a comfortable user interface and utilizes the JXMapViewer component which allows visualizing tile based maps. The user can subscribe to multiple areas of interests at the same time. The location information about the areas of interest are sent to the server side which processes the subscription request and henceforth forwards all events from sensors located within the specified area to the GIS client. The location
information is retrieved from the GPS subsystem on the XT75 and appended to each measurement message. The application also supports moving sensors and follows them as long as they are located in a predefined area of interest. Figure 3 shows a screenshot of the GIS client.

Figure 3: Screenshot of GIS client

4. Performance Evaluation

In order to compare the Web Service and JXTA approach with respect to the given scenario, a performance evaluation has been carried out which should indicate whether one of the approaches is favored to the other in terms of transmission efficiency and protocol characteristics. For that purpose the following measurements have been performed for each protocol:

- Data Traffic
- Transmission Delay
- Error Rate

The performance evaluation concentrates on the consequences of the bandwidth constraints of the GPRS-based communication between the embedded sensor node and the server side of the system and the processing constraints on the XT75. The following sections give an overview of the testbed and the obtained results of the performance evaluation.

4.1. Testbed Setup and Scenario Adaptations

The testbed being developed during the course of this work allows us to evaluate both approaches under the same preconditions. Figure 4 shows the entire test environment. The testbed considers tests for both protocols either over a public GPRS network or alternatively over a simulated GPRS network with the help of a mobile network emulator (Agilent Technologies 8960 Series 10 E5515C Test Set).

The mobile network emulator is used to verify all measurements performed via the public network against an encapsulated environment. Therefore the emulator is directly connected to a local router whereas in case of the public link all traffic is routed through the E-Plus GPRS gateway and the Internet to the server side. The mobile network of the German provider E-Plus Service GmbH & Co.KG has been chosen for all performance tests on the public link.

The server side of the system is deployed on two independent test sets. All Web Service related traffic is directly routed to the server side of the Web Service test set whereas all JXTA traffic is routed to the JXTA test set. In addition to the server side of the system, the JXTA test set incorporates the JXME Proxy, JXTA Rendezvous, and JXTA Relay which are necessary for the JXTA network setup.

For the performance evaluation of the communication protocols the original scenario setup has been adapted in order to avoid side effects from other operations in the scenario setup being not directly involved in the communication protocol flow. On the embedded platform the measurement values from the GPS and sensor subsystems were replaced by static values. The influence of the operations needed to measure the transmission delay on the embedded platform was less than 1ms which can be neglected with respect to the overall delay. On the server side the process of storing the measurement in the database was not performed in order to avoid any influences caused by accessing the database. On receiving new measurement values the server side directly replies with an acknowledgment and simply drops the retrieved message.

Figure 4: Testbed setup
4.2. Data Traffic

In order to evaluate the influence of the message size on the transmission delay, the size of the message was varied between 1 and 64 environmental measurement values per message. The exact number of bytes for each request or response is listed in Table 1. Each request incorporates:

- HTTP Header
- Device ID
- Localization Information (Longitude / Latitude)
- Time stamp
- Temperature of ambient air
- Methane and oxygen content of ambient air

The response incorporates the HTTP header and returns a “true”-symbol to the calling function to acknowledge the reception of the measurements. The response is therefore constant in size for all approaches.

**Table 1: Data traffic in dependency of protocol and number of measurement values**

<table>
<thead>
<tr>
<th>M</th>
<th>JXTA</th>
<th>JXME binary</th>
<th>Web Service</th>
<th>Web Service</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Req.</td>
<td>Rsp.</td>
<td>XML</td>
<td>WBXML</td>
</tr>
<tr>
<td>Req.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1.224</td>
<td>1.165</td>
<td>1.417</td>
<td>1.109</td>
</tr>
<tr>
<td>2</td>
<td>1.332</td>
<td>1.165</td>
<td>2.217</td>
<td>1.380</td>
</tr>
<tr>
<td>4</td>
<td>1.548</td>
<td>1.165</td>
<td>3.824</td>
<td>1.922</td>
</tr>
<tr>
<td>6</td>
<td>1.764</td>
<td>1.165</td>
<td>5.431</td>
<td>2.470</td>
</tr>
<tr>
<td>8</td>
<td>1.980</td>
<td>1.165</td>
<td>7.038</td>
<td>3.013</td>
</tr>
<tr>
<td>10</td>
<td>2.197</td>
<td>1.165</td>
<td>8.646</td>
<td>3.531</td>
</tr>
<tr>
<td>12</td>
<td>2.415</td>
<td>1.165</td>
<td>10.248</td>
<td>4.100</td>
</tr>
<tr>
<td>14</td>
<td>2.640</td>
<td>1.165</td>
<td>11.857</td>
<td>4.651</td>
</tr>
<tr>
<td>16</td>
<td>2.859</td>
<td>1.165</td>
<td>13.466</td>
<td>5.170</td>
</tr>
<tr>
<td>32</td>
<td>4.608</td>
<td>1.165</td>
<td>26.324</td>
<td>9.486</td>
</tr>
<tr>
<td>64</td>
<td>8.105</td>
<td>1.165</td>
<td>52.047</td>
<td>18.268</td>
</tr>
</tbody>
</table>

Table 1 highlights the differences between all three approaches. The standard Web Service approach suffers in terms of message size from the huge overhead of the textual representation of its XML-based protocols. Utilizing W3C’s binary representation of XML (WBXML) leads to a reduction in message size of nearly one third of the same message represented in XML. The JXME protocol which is also based on a binary representation becomes even more efficient for multiple measurement values.

4.3. Transmission Delay

In order to determine the coherence between the data traffic and the transmission delay the wireless GPRS link has been examined in more detail. Each test has been repeated 250 times for meaningful averaging of the observed processes. 5500 tests have been performed on both the public mobile network and the emulated network. The transmission delay was split into the following phases:

- Duration of the serialization process
- Duration of the transmission of the request (RTT)
- Duration for reading and processing the reply

Since the embedded platform and the server side have no synchronized clocks, the measurement had to be performed solely on the embedded platform. With respect to this constraint, the Round-Trip Time (RTT) consumed by sending the request until the retrieval of the response was measured.

![Figure 5: 250 Test runs for the transmission delay of JXTA with 16 measurement data sets](image-url)
The server side does not consume any processing time as it directly replies to the incoming measurement values without processing them.

Figure 5 shows as an example the result of 250 performed tests with 16 measurement data sets sent through JXTA. Even though a constant set of data is processed during serialization and deserialization the delay generated through these operations is quite volatile. This characteristic has already been observed and explained in [9]. The results of these measurements have been averaged and are the input of the following discussion for all three approaches.

In case of the standard XML-based Web Service approach with 64 measurement values transmitted (see fig. 6) it takes nearly 50 seconds for the whole process of serialization, transmission and deserialization of the reply, where 52,403 bytes are sent over the wireless link in total. The measurement clearly shows that the biggest part of the transmission delay is caused by the GPRS based communication channel during the send data phase.

Figure 6: Web Service Transmission Delay (XML)

The last series of measurements has been performed with the JXTA protocol (see figure 8). Compared to both Web Service approaches the delay of the entire transmission process evolves less critical for a high number of measurement values. The added overhead within the JXTA message to include further measurement values is very small compared to its XML counterparts. Even though the message size of the JXTA protocol is smaller than a WBXML-based message for more than 8 measurement values, the delay caused on the wireless link is still higher than for WBXML. This is due to the JXME proxy which has to be placed in between the embedded sensor gateway node and the server side system. For messages carrying more than 40 measurement values JXTA becomes more efficient with respect to the RTT even though the JXME proxy is embedded into the communication path. The same measurements performed on the emulated mobile network led to very similar results.

Figure 7: Web Service Transmission Delay (WBXML)

Figure 8: JXTA Transmission Delay
4.4. Error Rate

During the entire performance evaluation the number of failed requests were registered and logged. Using the emulated mobile network all 5500 tests succeeded without any failed requests. In case of the public mobile network the number of failed requests especially for the JXTA approach changed considerably. Figure 9 shows the distribution of failed service requests regarding all 5500 tests on the public network.

![Figure 9: Error Rate](image)

Reviewing the error log led to the finding that almost all of the errors occurred because the JXME response was not received by the XT75 in time. In order to improve the reliability of the JXME protocol the timeout between sending the request and receiving the response from the proxy has to be increased. The finding was that the JXTA protocol was much more volatile with respect to extreme outliers even though the averaged measurements were very good. This characteristic is also shown in figure 5 near the 20th and 225th test run.

5. Conclusion and Outlook

The measurements performed during the performance evaluation of our system have clearly shown that building the interface for embedded sensor nodes solely on the standard XML-based SOAP protocol is very inefficient. Utilizing the WBXML specification improves the Web Service performance in terms of the transmission delay due to smaller messages transmitted over the wireless link. The serialization and deserialization processes on the other side are only marginally improved considering the amount of time needed for these processes. The JXME protocol on the other hand benefits from its very efficient serialization phase and produces the lowest data traffic during the transmission of the measurement data sets. However, in JXTA a JXME peer has to connect and register with a Rendezvous peer on start-up which involves at least six message exchanges between the JXME peer and the JXME proxy service before being ready to send data to the data collection server. Considering only a small number of measurement values being transmitted during one connection session this complex connection setup seems to be not feasible. Hence, the JXTA protocol is only interesting for scenarios where a lot of data are transmitted during one connection session or where the connection remains active even if no data are transmitted.

The middleware developer therefore has to estimate the actual data traffic produced by the application carefully in order to determine which protocol allows for an efficient communication infrastructure of the middleware design. Additionally the WBXML-based Web Service approach benefits from its protocol specific wide acceptance of Web Services and its instant-on behavior. The JXTA approach on the other side has - next to its efficiency in case of large data sets - other P2P specific advantages like scalability and robustness which were not focused in this work.

In succession of this work we are currently investigating to wrap Web Services into JXTA. There has already been done some work in this field in [7] and [8] which we are investigating with respect to performance aspects on our existing testbed. We are furthermore working on the development of a binding between the Device Profile for Web Services and JXTA where JXTA is used as the backbone on the Wide Area Network in order to interconnect several DPWS enabled networks.

6. Acknowledgment

The work described in this paper was partly funded by the European Union through the MORE project in the 6th Framework Programme under the Reference-ID FP6-IST-032939. We would like to thank all project partners for their helpful contributions to this work.

7. References


