A WIRELESS PRODUCTION DATA VISUALIZATION SYSTEM FOR THE OIL AND GAS INDUSTRY

Guido Urdaneta[1], Juan Colmenares[1], Nelson Arapé[1], Nestor V. Queipo[1], Carlos Arévalo[2], Carlos González[3], José Quintero[4].

[1]*Applied Computing Institute. University of Zulia. Venezuela.*
[2]*Novatek Soluciones Tecnológicas C.A. Venezuela.*
[3]*Petróleos de Venezuela S.A. Venezuela.*
[4]*INTESA. Venezuela.*

Email: guidox@ica.luz.ve, juancol@luz.ve, narape@luz.ve, nqueipo@luz.ve, carevalo@novateknet.com, gonzalezcv@pdvsa.com, quinterojy@pdvsa.com

Keywords: oil production visualization, wireless applications, J2EE, web technology, industrial automation.

ABSTRACT

This paper discusses the design and implementation of a J2EE-based system for the visualization of production data using mobile personal wireless devices such as PDAs, cellular phones and portable computers. The design considers criteria such as interoperability, portability, scalability, performance, reliability, security, use of legacy systems and maintainability.

The system includes the following elements: i) an XML-based format for the description of a production data display, ii) a display generation application, iii) a specialized graphical display tool, iv) an interface for accessing real-time data, and v) a display design and management application.

The XML-based format for the description of a production data display is called SDDML (Simple Display Definition Markup Language). SDDML documents are generated by the Display Generation Application (DGA) which is based on Java Servlets and Enterprise JavaBeans. These documents may be transformed using XSL depending on the client nature. If the client is a WAP enabled phone they are transformed into WML, and if the client is an ordinary Web browser they are transformed into HTML. The specialized graphical display tool (SDDML Browser) takes a SDDML document generated by the DGA and displays it on a PDA or portable computer equipped with a Java Virtual Machine. The real-time data shown in displays is obtained through a standard interface called Real Time Data Source (RTDS). RTDS implementations provide decoupled access to constantly updated data identified with tags in real-time databases (e.g. PLC and SCADA systems). The Display Design and Management Application (DDMA) permits the creation and management of graphical displays. This application has server-side components responsible for the storage and retrieval of the display data in and from a relational database.

The system satisfies the aforementioned criteria and holds promise to be effective in the automation of Oil and Gas and other continuous process industries.

1. Introduction

Continuous processes such as those found in the Oil and Gas industry need to be constantly monitored. A delay in detecting or solving a problem in such processes could translate into unexpected financial expenditures or losses (e.g. delay of oil production) or the possibility of jeopardizing the health and safety of the workers. Furthermore, when problems arise the decision makers (e.g. supervisors) are not necessarily in the control rooms, and may not have available all the necessary information. An alternative to remotely supervise facilities while out of control rooms is the use of personal wireless devices with Internet access, such as Personal Digital Assistants (PDAs), mobile phones with data capabilities like the Wireless Access Protocol (WAP) or portable personal computers.

The authors are not aware of software solutions that will address the aforementioned problem while satisfying commonly required design criteria for enterprise applications (e.g. interoperability, scalability, reliability). In addition, even though there are Human Machine Interface products available for PDAs, they exhibit the following disadvantages:

- Provide their own security mechanism; as a result, it is not possible to authenticate and authorize users through existing corporate security infrastructures.

- Graphical displays of field information are stored in each PDA. This lack of centralized administration may be very difficult to deal with if the number of users is reasonably large and display definition change frequently.

- Usually require access to real time data using protocols not suitable for use in Internet applications, such as OPC (OLE for Process Control). It would be a serious security hole for companies to allow access to its real time data via OPC from the Internet.

- Only support certain PDA models, and do not support other wireless devices such as mobile phones. They do not even support desktop computers without the use of other products especially tuned for workstations.

This paper presents the design of a system based on Java 2 Platform, Enterprise Edition (J2EE) that addresses these issues. The system has been designed considering criteria such as maintainability, portability, interoperability, scalability, reliability and security. The system includes the following elements: i) real time data sources, ii) a simple display definition markup language, iii) a display definition database, iv) a display management and administration application, v) a display generation application and vi) a graphical visualization application. The remaining sections of the paper discuss the design criteria (Section 2), the system architecture (Section 3), the system implementation (Section 4) and some conclusions (Section 5).

2. Design Criteria

The design criteria imposed on the Wireless Production Data Visualization (WPDV) system are listed below.

<u>Maintainability and Functionality</u>

The system must support the display of production data on wireless devices connected via the Internet and be able to create graphical displays associated with different production facilities. In addition, to facilitate the system maintenance, the following criteria must be met:

- The definitions of the graphical displays must be stored on a central repository, such as a relational database and not on individual client devices.

- There must be an utility to manage the display creation, modification and removal.

- The graphical symbols used to define displays must be designed as an extensible library.

- The software running on client devices must be able to be upgraded automatically after being first installed.

- There must be a facility for client devices to search the displays by a name or ID.

Portability

The system must support several wireless platforms, including web enabled mobile phones (e.g. cellular phones equipped with Wireless Access Protocol), PDAs and personal computers.

Interoperability

The information transfer to wireless client devices must be made through standard Internet protocols such as HTTP or HTTPS. Moreover, the system must have potential access to several real time data sources (e.g. PLCs, RTUs and SCADA systems) and a mechanism for the transparent addition of new data sources.

Scalability and Performance

It should be possible to support an increasing number of clients by adding bandwidth and/or hardware without modifying the software.

Reliability

The system must not have single points of failure in its server side elements. This could be achieved using high availability techniques such as clustering and redundancy.

Security

The system must allow:

- Access to production data only to properly authenticated and authorized users. The authentication and authorization mechanism must be configurable, so that existing security infrastructures may be used.
- Read only access for Internet users.
- The encryption of production data as it is transferred to client devices.

3. System Architecture

The WPDV system is based on the Java 2 Platform, Enterprise Edition [1], which is an open platform that makes possible meeting the previously mentioned design criteria. The WPDV system architecture (see Figure 1) includes the following elements:

- Real Time Data Sources (RTDS)
- Simple Display Definition Markup Language (SDDML)
- Display Definition Database and Entity Components
- Display Definition and Management Application
- Display Generation Application
- SDDML Browser

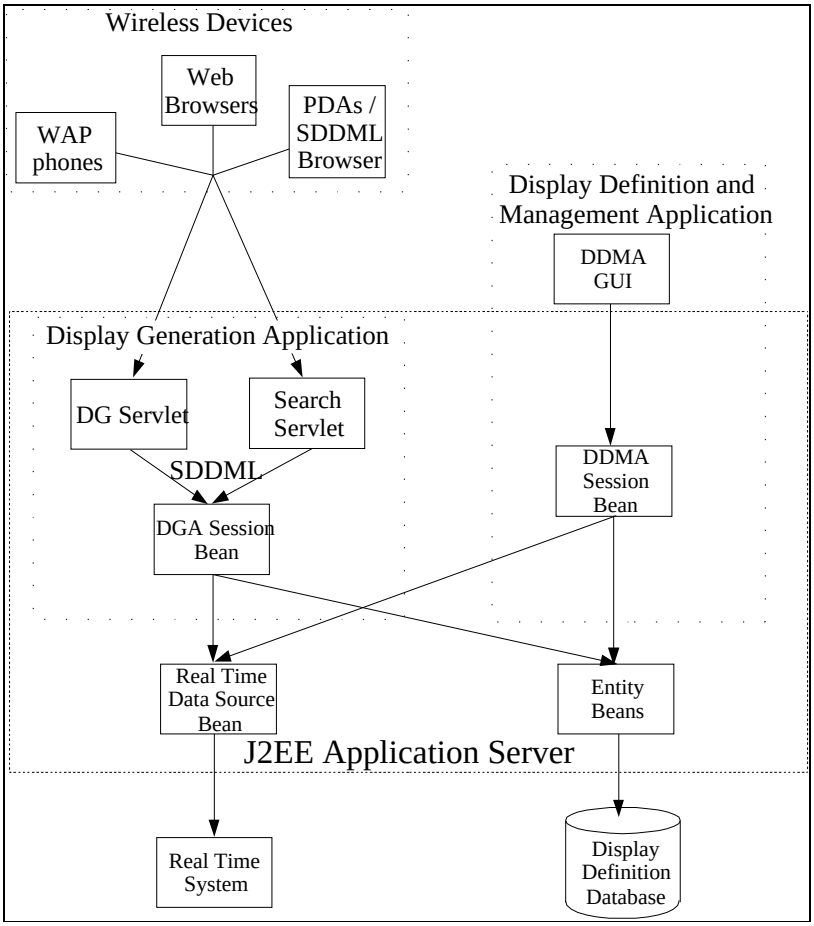

Figure 1. WPDV System Architecture

4. System Implementation

Real Time Data Sources (RTDS).

Real Time Data Sources define a standard interface to systems that provide continuously updated data identified with tags (e.g. SCADA systems). RTDS components are implemented as Stateless Session Enterprise JavaBeans (EJB) [2] and provide operations for manipulating present and historical production data.

The design of the RTDS interface is based on the functionality provided by OLE for Process Control (OPC)[3], CORBA Data Access to Industrial Systems (DAIS) [4], OSI Software Plant Information (PI) system and Metso Open Architecture Scada System (OASYS).

It should be noted that this interface is not specific to the WPDV system and is part of a previously reported architecture for the development and integration of high level industrial automation applications of continuous process industries [5].

Display Definition Database and Entity Components

The WPDV system stores all display definitions in a relational database. This provides several advantages:

    - Centralized administration: There is only one definition of a graphical display, so it is easy to make modifications and additions and make them available to all users.

    - Data integrity: The database design does not allow the storage of illegal display definitions.

- Security: The RDBMS restricts the operations users can perform on the display definitions.

All the information in the database is accessed from other parts of the system using Entity EJBs. The entity beans defined by the WPDV system are the following:

- *Display*: represents a graphical display.

- *Widget*: A widget represents a graphical symbol that is part of a Display. Widgets have a set of named attributes, which define the way a symbol is displayed in a client application.

- *WidgetType*: All widgets must have an associated WidgetType. A WidgetType defines the behavior of a Widget and its attributes. Widgets are instances of their respective WigetType. A WidgetType could be “Pump” and a Widget whose WidgetType is “Pump” could be “Pump 1 in Flow Station 1”.

There are several types of attributes: Numeric, String, Color, Font, Boolean and Tag. The Tag attribute type represents data obtained from an RTDS. The value of a Tag attribute is stored in the database as a character string that includes the name of the RTDS, the tag name, and the type of operation to be invoked on the RTDS interface when the display is retrieved (e.g. current or historical data access).

For example, the “Pump” WidgetType could specify that its attributes are:

- *tag* - type Tag: The RTDS tag that provides the data that indicates if the pump is on or off.

- *color_when_off* - type Color: The color to display the pump when it is off.

- *color_when_on* - type Color: The color to display the pump when it is on.

Apart from these attributes defined by the WidgetType, all Widgets have a number of common properties such as coordinates and size and a property called *ignorability* that specifies circumstances in which the attribute can be ignored (e.g. a color attribute may be ignored by WAP clients, since they do not have color capability).

<u>Simple Display Definition Markup Language (SDDML)</u>

The WPDV system supports several types of client devices including WAP enabled mobile phones, HTML browsers and a specialized client application that runs on PDAs and personal computers.

All data produced by the server side parts of the WPDV system are XML documents [6]. As a result, the data to be sent to the client is in an independent format and can be transformed into another format suitable to the client device, using XML Stylesheet Language Transformation (XSLT) [7]. For example, a WAP phone expects data in Wireless Markup Language (WML) format, while an ordinary web browser expects HTML.

The WPDV system defines three XML document types for describing the possible documents to be transformed and sent to client devices. These document types are defined using the XML Schema specification [8] and are called Simple Display Definition Markup Language (SDDML). These XML document types are Display Definition, Display Tag Values and Display List:

- *Display Definition*: Describes all the elements of a graphical display, including information about other related displays and graphical symbols (Widgets) included in the display, such as coordinates and attribute values. The Display Definition document for a pump and a pipe follows:

```
<?xml version="1.0"?>
<display id="1" name="display1" refresh="25">
   <description>Example Display 1</description>
   <display_link_list>
      <display_link id="2" name="display2"/>
   </display_link_list>
   <widget_list>
  <widget id="1" type_id="1" coord_x="10" coord_y="25" coord_z="1" width="20"
                        height="15" display_link_id="15" ignore="n">
         <description>Pump 1</description>
         <widget_attribute name="tag" ignore="n">
            <tag_value_list>
               <tag_value value="82.77765097871207" timestamp="648" quality="bad"/>
            </tag_value_list>
         </widget_attribute>
         <widget_attribute name="color_when_off" ignore="w">
            <numeric_value value="-16776214"/>
         </widget_attribute>
         <widget_attribute name="color_when_on" ignore="w">
            <numeric_value value="-65536"/>
         </widget_attribute>
      </widget>
   <widget id="2" type_id="2" coord_x="60" coord_y="10" coord_z="1" width="30"
                        height="30" display_link_id="14" ignore="h">
         <description>Pipe</description>
         <widget_attribute name="size" ignore="h">
            <numeric_value value="30"/>
         </widget_attribute>
         <widget_attribute name="color" ignore="h">
            <numeric_value value="-65536"/>
         </widget_attribute>
      </widget>
   </widget_list>

</display>
```


- *Display Tag Values*: This format is similar to the previous one, but only includes information corresponding to attribute values associated with data obtained from RTDSs. The Display Tag Values document for the pump and pipe display follows:


```
<?xml version="1.0"?>
<display_tag_values>
   <widget_tag_values id="1">
      <widget_tag_attribute name="tag">
         <tag_value_list>
            <tag_value value="82.77765097871207" timestamp="648" quality="bad"/>
         </tag_value_list>
      </widget_tag_attribute>
   </widget_tag_values>

</display_tag_values>
```


- *Display List*: Describes a display list that typically results from a search request. It includes the ID, name, description and creation date of each display. An example follows:


```
<?xml version="1.0"?>
<display_reference_list>
   <display_reference id="1">
      <name>display1</name>
      <description>Example Display 1</description>
      <creation_date>2001-11-30</creation_date>
   </display_reference>
   <display_reference id="2">
      <name>display2</name>
      <description>Example Display 2</description>
```

```
        <creation_date>2001-12-03</creation_date>
    </display_reference>
</display_reference_list>
```

Display Definition and Management Application (DDMA)

This application allows the creation and management of graphical displays and is divided in two parts: a client GUI and a server side component.

The client (DDMAGUI) is a Java application for the edition of displays using a GUI (see Figure 2). It communicates with a stateless session bean on the server side to retrieve and save displays. This application can be deployed using the Java Network Launching Protocol technology [9] (also known as Java WebStart). Every time the application changes, it is automatically downloaded by the clients from a web server, which allows a centralized administration of the application.

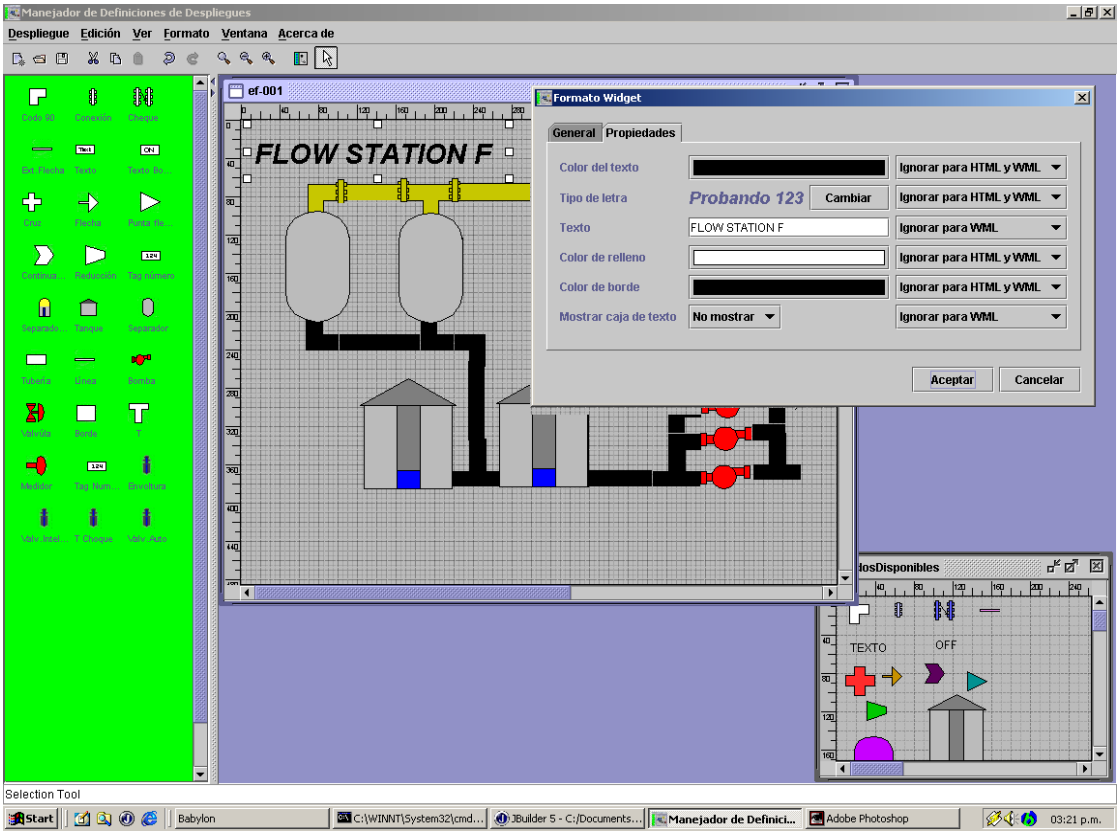

Figure 2 – DDMA GUI Screenshot

The server side session bean (DDMAServ) acts as a facade [10] to the entity beans, providing higher level operations to the client application, such as retrieving or deleting a display from the database.

Display Generation Application (DGA)

This application processes the requests made by the wireless client devices using HTTP or HTTPS and is implemented by a stateless session bean and two Java servlets. The DGA stateless session bean accesses the entity and RTDS components and, based on the information provided by them, constructs a SDDML document. This bean provides operations for creating all three types of SDDML documents. The servlets have the responsibility of determining the type of client device (e.g. WAP device, web browser), making an appropriate call to the DGA session bean and applying the suitable XSL transformation. One of the servlets processes display search requests while the other processes requests for Display Definition and Display Tag Values documents.

Figure 3 shows the output of the search servlet on a WAP phone emulator and an ordinary web browser.

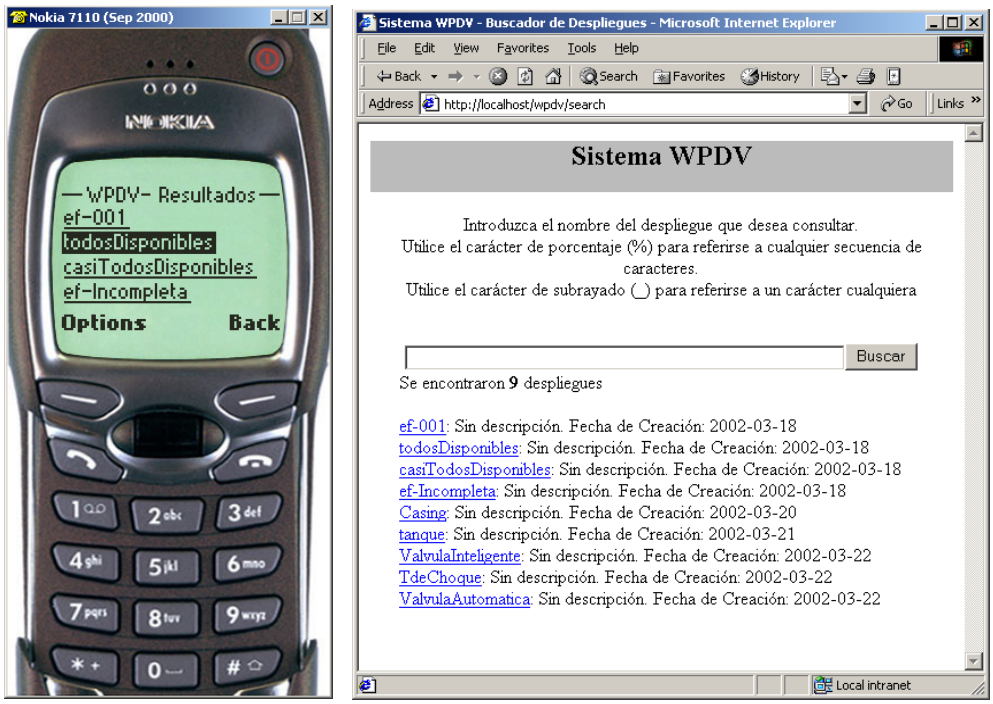

Figure 3. Output of the Search Servlet on a WAP phone emulator and an HTML browser

Figure 4 shows the output of the display generation servlet on a WAP phone emulator, web browser and the SDDML browser.

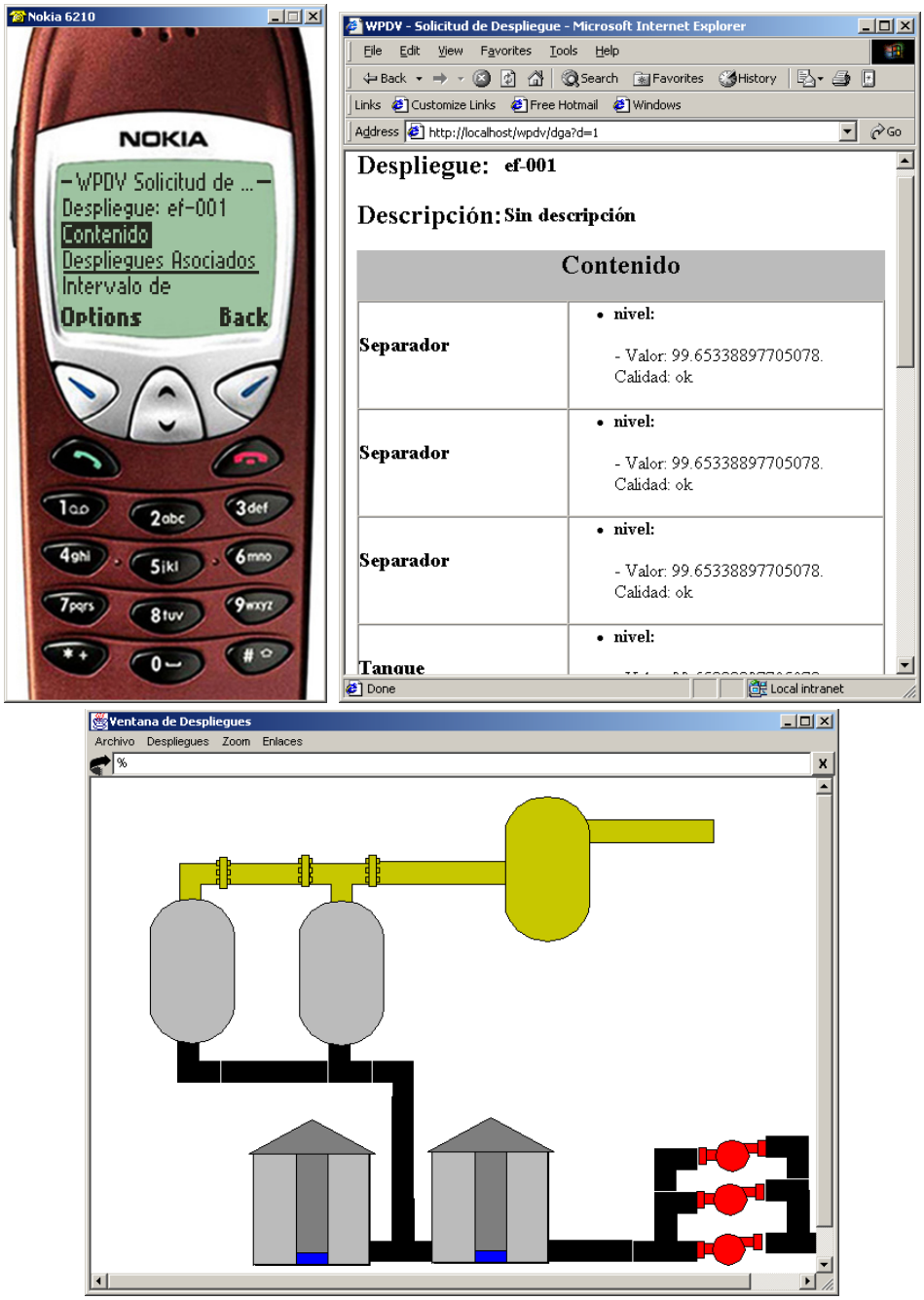

Figure 4. Output of the Display Generation Servlet on a WAP phone emulator, HTML browser and SDDML browser

It should be noted that the WML and HTML support of the WPDV system consists of text only information due to the limitations of the bandwidth available for wireless communications.

SDDML Browser

The SDDML browser is a client application developed according to the PersonalJava specification [11], which is a subset of the standard edition of the Java platform suitable for execution in resource constrained devices, such as PDAs. This application makes HTTP/HTTPS requests to the DGA and receives compressed SDDML documents without transformation. The SDDML browser is capable of showing a graphical representation of the information contained in the SDDML documents. Figures 4 and 5 show the SDDML browser running on personal computer and a PDA, respectively.

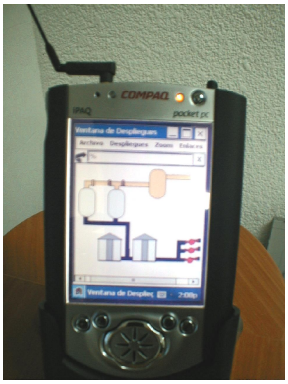

Figure 5. SDDML browser running on a PDA

When a user retrieves a display, the SDDML browser requests a Display Definition SDDML document and shows it graphically. Alternatively, for updating the real time information of an already retrieved document, the browser requests a Display Tag Values SDDML document.

While the SDDML documents provide information that defines the way a widget should be drawn in a display, they do not provide the machine code required to perform the actual drawing. The machine code is provided in a class library that has the routines required to draw each widget type and a configuration file. This library is extensible, so that new widget types can be added and is kept in a web server. Every time the SDDML browser starts, it updates the library if necessary. This feature facilitates the system maintenance, because the code that performs the widget drawing can be updated without manually reconfiguring any client device.

5. Conclusions

This paper discussed the design of a Wireless Data Production Visualization system for the Oil and Gas industry based on the Java 2 Platform, Enterprise Edition (J2EE) and the Extensible Markup Language (XML). The system has been designed considering criteria such as maintainability, portability, interoperability, scalability, reliability and security. The architecture of the system includes the following elements:

- *Real Time Data Source (RTDS)*: provides a unified interface for accessing systems that provide real time data, such as tag database and SCADA systems.

- *Simple Display Definition Markup Language (SDDML)*: Defines XML document types for transferring information from the server side components of the system to client applications. These documents describe displays definitions, real time data for updating a display and search results.

- *Display Definition Database and Entity Components*: The database is a centralized repository for all the graphical display definitions that can be visualized by the users of the system. The entity

components provide a high level object oriented abstraction of the information stored in the database.

- *Display Definition and Management Application (DDMA)*: Permits the creation of displays using a graphical editor.

- *Display Generation Application (DGA)*: Server side application that generates SDDML documents. The application transforms these documents to a format suitable for the client device making a request (e.g. WML for a WAP enabled phone).

- *SDDML Browser*: this is a Java application that runs on any device with an appropriate virtual machine, such as personal digital assistants or personal computers, and allows the graphical display of the SDDML documents produced by the DGA.

The system satisfies the aforementioned criteria and holds promise to be effective in the automation of Oil and Gas and other continuous process industries.

REFERENCES

[1] Sun Microsystems, Inc. (2001). Java 2 Platform, Enterprise Edition Specification, v1.3. Accessed online at http://java.sun.com.

[2] Sun Microsystems, Inc. (2001). Enterprise JavaBeans Specification, Version 2.0. Accessed online at http://java.sun.com.

[3] OPC Foundation. OLE for Process Control Overview. Accessed online at http://www.opcfoundation.org.

[4] ABB Automation Systems, et al. (2000). Data Acquisition from Industrial Systems. Accessed online at http://www.omg.org.

[5] Urdaneta, Guido et al. (2001). A Component Based Automation Architecture for Continuous Process Industries. Proceedings of the 5th International Conference on Engineering Design and Automation.

[6] W3C (2000). Extensible Markup Language (XML) 1.0 (Second Edition). Accessed online at http://www.w3c.org.

[7] W3C (2001). Extensible Stylesheet Language (XSL) Version 1.0. Accessed online at http://www.w3c.org.

[8] W3C (2001). XML Schema Part 0: Primer. Accessed online at http://www.w3c.org.

[9] Sun Microsystems, Inc. (2001). Java Network Launching Protocol & API Specification, Version 1.0.1. Accessed online at http://java.sun.com.

[10] Gamma, Erich et al. (1994). Design Patterns. Addison Wesley.

[11] Sun Microsystems, Inc. (2000). PersonalJava Application Environment Specification, Version1.2a. Accessed online at http://java.sun.com.