



Installation and Configuration Manual

DataMóvil

Version 3.1

Satec DataMovil Installation and Configuration Manual

The software described in this Manual is furnished under a license agreement and may be used only in accordance with the terms of the agreement.

Copyright Notice

Copyright © 2004-2007 SATEC S.A.

All rights Reserved

This document may not, in whole or in part, be copied, photocopied, reproduced, translated, or reduce to any electronic medium or machine-readable form without prior consent in writing by SATEC, S.A. Av. Europa 34, Madrid, Spain.

ALL EXAMPLES WITH NAMES, COMPANY NAMES, OR COMPANIES THAT APPEAR IN THIS MANUAL ARE IMAGINARY AND DO NOT REFER TO, OR PORTRAY, IN NAME OR SUBSTANCE, ANY ACTUAL NAMES, COMPANIES, ENTITIES, OR INSTITUTIONS. ANY RESEMBLANCE TO ANY REAL PERSON, COMPANY, ENTITY, OR INSTITUTION IS PURELY COINCIDENTAL.

Every effort has been made to ensure the accuracy of this manual. However, SATEC makes no warranties with respect to this documentation and disclaims any implied warranties of merchantability and fitness for a particular purpose. SATEC shall not be liable for any errors or for incidental or consequential damages in connection with the furnishing, performance or use of this manual or the examples herein. The information in this document is subject to change without notice.

Trademarks

DataMovil, SATEC, and the SATEC logo are trademarks of SATEC, S.A.

Other products names mentioned in this manual may be trademarks or registered trademarks of their respective companies and are the sole property of their respective manufacturers.

SATEC, S.A.

Avda de Europa 34 A, 28023 Aravaca - Madrid (España) Tel.: (+34) 91 708 90 00

Index

1. Introduction	1
1.1. What is DataMovil?	1
1.2. What is DataMovil composed of?	1
1.3. How does DataMovil work?	2
2. Installation Steps Summary.	4
3. Contents of the DataMovil CD.	5
3.1. Root	6
3.2. DataMovilServer	6
3.2.1. War	6
3.2.2. License	6
3.2.3. XML-DBMS	6
3.3. DataMovilBrowser	6
3.3.1. PocketPC	6
3.3.2. MIDP	7
3.3.3. BlackBerry	7
3.4. DataMovilEditor	7
3.5. DesktopEvaluation	7
3.6. Documentation	7
3.7. Samples	7
4. Installation Prerequisites.	8
5. Installation and configuration of the DataMovil server.	9
5.1. Relational Database	9
5.1.1. Applications Table	9
5.1.2. Versions Table	10
5.1.3. Sequences Table	11
5.1.4. Resources Table	12
5.1.5. Associations Table	13
5.1.6. Assoc_Res Table	14
5.2. Administration	15
5.2.1. Administration Configuration	15
5.2.2. Using custom plugins	16
5.2.3. The “received XML” directory	17
5.3. Version Control and data reception	17
5.3.1. Configuration	18
5.3.2. Installation of the license file	20

5.3.3.	Using server custom plugins.....	20
5.3.4.	The “received XML” directory.....	21
5.4.	Users and groups management.....	21
5.4.1.	Creation of the tables.....	22
5.4.2.	Installation of the web application.....	23
5.4.3.	Configuration of the application.....	24
6.	Installation and configuration of the DataMovil smart client.	26
6.1.	Installation in PocketPC.....	26
6.2.	Installation in MIDP devices.....	26
6.3.	Installation in BlackBerry devices.....	27
6.4.	Configuration files for POCKETPC devices.....	27
6.4.1.	Data.ini.....	27
6.4.2.	Variable Names.....	32
6.4.3.	Messages_es.ini / Messages_en.ini.....	32
6.4.4.	DMLauncher.ini.....	36
6.5.	Configuration files for MIDP/BlackBerry devices.	37
6.5.1.	Data.ini.....	37
6.5.2.	Variable Names.....	37
6.5.3.	Messages_es.ini / Messages_en.ini.....	37
7.	Installation of the applications editor.....	39
8.	Installation of the evaluation version.	40
9.	Installation and execution of the sample applications.	42
9.1.	PetStore.....	42
9.2.	SmokingHabit.....	43

1. Introduction

1.1. What is DataMovil?

DataMovil is a platform for the rapid development and deployment of applications for mobile devices like PDAs. It mainly addresses applications with wireless network connections like GPRS, CDMA, 3G or Wi-Fi (802.11x), though it can be also used to execute applications local to the device.

It is a platform originally designed for PDAs, based on standards and with utilities that facilitate the integration with corporate backoffice systems. Since 3.0 version, DataMovil is also available for MIDP 2.0 devices.

1.2. What is DataMovil composed of?

DataMovil is composed of the following modules:

DataMovil smart client

It is an application executed in the device (PC, PDA, mobile phone...). This application is executed based on data which is received from a server or loaded from a local file, in XML format. This XML incorporates all the data relevant and the actions to be executed over the data, by the end-user or automatically by the client itself.

As a result of the execution of an XML received by the client, DataMovil will send an XML to the server or, alternatively, it may be stored locally. The server may answer sending another XML to the client.

The XML handled by the DataMovil smart client is based on the XForms 1.0 W3C Recommendation of October 14th 2003. The small differences between the standard and DataMovil arises from the computing resources limitations of a mobile device where DataMovil will be executed and to provide with some extra functionality in some aspects related to communications with peripheral devices.

DataMovil Server:

The DataMovil server includes several components that provide different services:

Application downloading and version control. This functionality is provided by a servlet that it is invoked by the client in order to discover what new applications or new versions of already downloaded applications are available for the user in the server.

Engine to include the data sent by DataMovil, in a relational database. Through the edition of a template that links specific XML tags to tables and fields of any relational database, the engine may incorporate all the data received by the client.

Additionally, the developers may program their own servlets to send a DataMovil-XForms answer to the client at any time.

Administration and development of DataMovil applications:

DataMovil includes several tools as a help to the development of applications. A web based administration is responsible for the registry of the applications created and the proper maintenance of the different versions of an application, including their current status.

For the application development, there is a Java “swing” based interface, executed at the desktop by the developer. This tool provides the following improvements:

- Based on Java, improves the developer productivity.
- Supports the whole XML included in DataMovil.
- Quick preview of individual elements or a group of elements for standard and mobile devices.
- Includes the new published DataMovil XML Schema.
- Synchronization with the administration in order to import/export applications.

1.3. How does DataMovil work?

In order to understand how DataMovil works, we may think about DataMovil comparing it with the traditional web applications, realizing this way its differences and advantages.

In the first place a web application is based upon a web browser that interprets the HTML language composed of tags focused on the presentation of documents on a computer screen. In addition, the web browser incorporates scripting languages like JavaScript to include some processing logic. The data, the presentation and the processing logic are all mixed up in a web application.

In the case of DataMovil, the client is based on XForms; that is an XML language that has a clear separation between the data model and the processing logic. It does not include either presentation logic or scripting languages. The processing logic is included in the XML but the presentation logic is not incorporated to the XForms standard. In the current version of DataMovil, the presentation is automatically handled by the client though it may be configured through the “data.ini” file. From version 2.1 on, the presentation and layout of the different elements of each application may be customized at will by the developer.

These DataMovil features greatly diminish the development time and the maintenance costs as it eliminates the well known problems associated to the maintenance of scripting code and, largely due to the separation of the data model and the processing logic.

The DataMovil smart client communicates with the server in a similar way to a web browser sending and receiving data. The main difference is that DataMovil always sends data in XML format and receives data in XML according to the DataMovil-XForms language based on XForms. An application

may be built by means of a sequence of exchanges with one or several servers, the same way as we build applications based on a web browser.

However, the DataMovil smart client presents a number of differences with a web browser, in addition to the already mentioned, that makes DataMovil more flexible and adapted to the mobile environment. A user as he/she starts DataMovil in the mobile device, has the option to execute already loaded applications (files in DataMovil-XForms format) or to connect to a server and download the applications. The user may save the result data locally in the device or send them to a server. In case the user has data saved in the device, he/she can send them to a server at any time a connection to the server is available. In this way, the user is protected against possible lack of connectivity at any time. In the same way, a user can save the state of an application locally, in the device memory, at any time (for example, saving the result of an intermediate state of an application composed by multiple interactions with a server) and to recover that state later on to continue the execution of the application.

Another differential feature of the DataMovil smart client is that the client is able to handle a single DataMovil-XForms in a sequence of device screens, minimizing the number on interactions with the server. On the other hand, this makes more friendly the user experience with the device, reducing the annoying scroll bar manipulations. The DataMovil smart client shows automatically the buttons that allow the navigation through the different screens.

The DataMovil server provides the management of the files created in DataMovil-XForms format that allows a user to list and select the applications available for him and their different versions. This feature is included in a servlet that the client invokes to access to “remote applications”. This servlet returns the updated list of applications. Once the user has chosen one application, it can be stored in the device and, then, executed. If the application has several interactions with the server, these interactions require the development of the corresponding servlets the same way as we would proceed in the case of a web application, taking into account that the servlet must return a DataMovil-XForms.

This is the very summarized functioning of DataMovil, which must be taken into account to implement a mobility solution.

2. Installation Steps Summary.

1. **Check** that the prerequisites of paragraph 4 are satisfied. In case that any of the prerequisites is not satisfied, it will be needed to proceed to the installation of the prescribed software before starting the DataMovil installation.
2. **Create** the **tables** of the **database**.
3. **Install** the **version control servlet** in the web server, realizing the deployment of the war file provided.
4. **Install** the **administration** in the web server, realizing the deployment of the war file provided.
5. **Install** the **users administration** in the web server, realizing the deployment of the war file provided.
6. **Configure** the version control modifying the **config.properties** file to adapt it to the platform where the software has been installed.
7. **Configure** the administration modifying the **config.properties** file to adapt it to the platform where the software has been installed.
8. **Configure** the user administration modifying the **config.properties** file to adapt it to the platform where the software has been installed.
9. **Install** the **DataMovil smart client** in the devices (PDAs, mobile phones, tablet pcs...).
10. **Install** the applications editor.

3. Contents of the DataMovil CD.

The directory and files structure of the DataMovil CD is shown in the following figure:

```

\
|--- DataMovilServer
|   |--- war
|       |--- adminDataMovil3.1.war
|       |--- controlDataMovil3.1.war
|       |--- usersDataMovil3.1.war
|   |--- license
|       |--- license.jar
|   |--- XML-DBMS
|       |--- xmldbms101.zip
|
|--- DataMovilEditor
|   |--- EditorDM.exe
|
|--- DesktopEvaluation
|   |--- DataMovilDesktopEvaluation.exe
|
|--- DataMovilBrowser
|   |--- PocketPC
|       |--- DataMovil3.1.exe
|   |--- MIDP
|       |--- DataMovilME.jar
|       |--- DataMovilME.jad
|   |--- BlackBerry
|       |--- 4.1/
|           |--- DataMovilME.cod
|           |--- deploy.bat
|       |--- 4.2/
|           |--- DataMovilME.cod
|           |--- deploy.bat
|
|--- Documentation
|   |--- Installation and Configuration Manual.pdf
|   |--- Programming Manual.pdf
|   |--- Client User Manual.pdf
|
|--- Samples
|   |--- PetStore
|       |--- Server
|           |--- PetStore.war
|       |--- DataMovil-XForms
|           |--- PetStore_1_1
|   |--- Smoking Habits
|       |--- SmokingHabits_1_1
|       |--- SmokingHabitsTitle.jpg
|
|--- readme.txt

```

--- license.pdf

In case DataMovil is downloaded from Internet, the compressed file in zip format will contain a directory file structure identical to the previous one.

A description of the contents of the different directories follows:

3.1. Root

This directory contains the readme.txt and license.txt files. The last file contains the Terms of Use of the product.

3.2. DataMovilServer

This directory contains the subdirectories with all the files related to de DataMovil server.

3.2.1. War

It includes the files ready to make the deployment in the web server. The file for the administration of the product and the file for the version control of applications.

- adminDataMovil3.1.war
- controlDataMovil3.1.war
- usersDataMovil3.1.war

3.2.2. License

It includes the **license.jar** file that contains the license of the product. In case this file were absent, contact the DataMovil support service. In case the file were provided by different means, it must be copied to this location to ensure the correct functioning of the product.

3.2.3. XML-DBMS

It includes the software package that allows to integrate XML data into relational databases.

3.3. DataMovilBrowser

This directory contains all the subdirectories and files related to the DataMovil smart client.

3.3.1. PocketPC

Install file of the DataMovil smart client for PocketPC platforms and configuration editing utility.

3.3.2. MIDP

Install files of the DataMovil smart client for MIDP enabled devices.

3.3.3. BlackBerry

Install files of the DataMovil smart client for BlackBerry devices.

3.4. DataMovilEditor

This directory contains the executable file with the assistant to install the DataMovil editor in a computer with Windows operating system.

3.5. DesktopEvaluation

The DataMovil evaluation version includes, in a single package, all what is needed to test the functionalities of the product.

3.6. Documentation

This directory contains the manuals of the product in pdf format.

- Installation and Configuration Manual.
- Programming Manual.
- Client User Manual.

3.7. Samples

This directory contains all the files needed to execute two sample applications created with DataMovil.

The SmokingHabit application consists in the introduction in the device of the answers to questions for a survey about smoking habits in the sanitary sector. It is addressed to health professionals and the questions are related to tobacco consumption.

The PetStore application is a DataMovil-XForms version of the well-known application used many times to illustrate J2EE issues. In this version we show two things: the first one is how we can have many steps of the application in a single interaction with the server and, secondly, how we can have DataMovil applications with dynamic DataMovil-XForms “pages”. For this application, in addition to the initial XML file we need to deploy the war file provided in the web server.

4. Installation Prerequisites.

In order to install DataMovil 3.1, it is necessary to have in place some software packages prior to start the installation. These packages are:

- Web server enabled to execute Java servlets and JSP pages (i.e. Tomcat from Apache or SunOne Web Server)
- A relational database (i.e. Oracle, MySQL, SQL Server)
- A Java Virtual Machine in the PDA compatible with PersonalJava 1.2 or Personal Profile (i.e. JBed from Esmertec or J9 from IBM).
- A Java Virtual Machine in the MIDP devices compatible with MIDP 2.0.
- A personal computer with Windows operating system and a JVM 1.5 or higher.

The web server will be the repository where the administration pages will be located. These pages will allow us validating, invalidating or deleting the applications. Moreover, the servlets for the version control and the data reception, with which the DataMovil smart client will communicate to update applications or to send data, are to be located in this web server as well

All these data (applications and application generated data) will be saved in the relational database, that will be the central information repository.

Lastly, as the client is coded in Java language, providing portability between different environments, we need that the PDA to be used is provided with a Java Virtual Machine (JVM), in this case PersonalJava 1.2 or Personal Profile compatible.

5. Installation and configuration of the DataMovil server.

5.1. Relational Database.

The DataMovil server makes use of a relational database, where the application data is stored to be used by the clients later on. This database can be any of the relational databases most commonly found in the market like Oracle, MySQL or SQLServer.

A database must, therefore, be installed or an existing one could be used instead. Upon the database, a data model must be created according to the following directions.

The data model of the system is composed of six tables:

- Applications
- Versions
- Sequences
- Resources
- Associations
- Assoc_Res

Now, we describe each table and the fields in each one of them.

5.1.1. Applications Table

This table will save the information of the applications created with the administration. The script to create this table is the following:

```
CREATE TABLE APPLICATIONS (
  IDXFORM decimal(8) default '0' NOT NULL,
  IDAPPLICATION decimal(8) default '0' NOT NULL,
  APPLICATIONNAME VARCHAR (100) NOT NULL,
  TITLE VARCHAR (100) NOT NULL,
```

```
DESCRIPTION VARCHAR (255) NOT NULL,
XFORM LONG NOT NULL,
PRIMARY KEY (IDXFORM));
```

The description of the fields of this table is the following:

- **IDXFORM:** Unique identifier for each DataMovil-XForms file
- **IDAPPLICATION:** Unique identifier of each application.
- **APPLICATIONNAME:** Name of the application.
- **TITLE:** Title of the application to be shown to the user.
- **DESCRIPTION:** Description of the application.
- **XFORM:** This field keeps the DataMovil-XForms that defines the application. The type of this field depends on the database to be used. In the example, the script for the Oracle database is shown.

To create the table, the tools provided by the database we are working with should be used (i.e. SQLPlus for Oracle).

5.1.2. Versions Table

This table stores the information about the different versions of the applications created with the administration. Furthermore, this table will be accessed when a DataMovil smart client connects to the server in order to check whether there are new versions of the applications already installed in the PDA.

The script to create this table is the following:

```
CREATE TABLE VERSIONS (
IDVERSION decimal(8) default '0' NOT NULL,
IDXFORM decimal(8) default '0' NOT NULL,
VERSION decimal(3) default '0' NOT NULL,
VALIDATED char(1) NOT NULL,
VALIDATIONDATE DATE,
PRIMARY KEY (IDVERSION));
```

The description of the fields of this table is the following:

- IDVERSION: Unique identifier for the version.
- IDXFORM: Identifier of the application associated to the version.
- VERSION: Updated version number of the application.
- VALIDATED: Flag to indicate whether the version of the application is validated or not.
- VALIDATIONDATE: Date when the version is validated.

To create the table, the tools provided by the database we are working with should be used (i.e. SQLPlus for Oracle).

5.1.3. Sequences Table

This table is used to obtain unique identifiers associated with the fields of the previous tables. In this way we are able to provide compatibility with different database management systems.

The creation of the table 'SEQUENCES' is made with the following SQL sentence:

```
CREATE TABLE SEQUENCES (
  ID_SEQUENCE decimal(8) default '0' NOT NULL,
  TABLENAME varchar(250) default ' ' NOT NULL,
  FIELD varchar(250) default ' ' NOT NULL,
  SEQ_VALUE decimal(10) default '0',
  INCREMENTVALUE decimal(10) default '0' NOT NULL,
  MINIMUMVALUE decimal(10) default '0' NOT NULL,
  MAXIMUMVALUE decimal(10) default '0' NOT NULL,
  CIRCULAR char(1) default ' ' NOT NULL,
  PRIMARY KEY (ID_SEQUENCE),
  UNIQUE (TABLENAME, FIELD))
```

The description of the fields of this table is the following:

- ID_SEQUENCE: Unique identifier of the sequence.
- TABLENAME: Name of the table that this sequence belongs to.

- FIELD: Name of the field associated to this sequence within the table.
- SEQ_VALUE: Next value of the sequence.
- INCREMENTVALUE: Value of the number of units in which the field VALUE is incremented each time we request a new sequence number.
- MINIMUMVALUE: Minimum value of the VALUE field.
- MAXIMUMVALUE: Maximum value for the VALUE field.
- CIRCULAR: Flag describing whether a sequence is circular or not. That is, a sequence is circular if when it reaches the maximum value, the sequence starts over with the minimum value as the next value. Its value is “T” if the sequence is circular. For any other value it is assumed not to be circular.

To create the table, the tools provided by the database we are working with should be used (i.e. SQLPlus for Oracle).

Five registers in this table must be created through the following sentences:

```

INSERT INTO SEQUENCES (ID_SEQUENCE, TABLENAME, FIELD, SEQ_VALUE, INCREMENTVALUE,
MINIMUMVALUE, MAXIMUMVALUE, CIRCULAR) VALUES ( 1, 'APPLICATIONS', 'IDXFORM', 0, 1, 1,
999999999, 'T');

INSERT INTO SEQUENCES (ID_SEQUENCE, TABLENAME, FIELD, SEQ_VALUE, INCREMENTVALUE,
MINIMUMVALUE, MAXIMUMVALUE, CIRCULAR) VALUES ( 2, 'APPLICATIONS', 'IDAPPLICATION', 0,
1, 1, 999999999, 'T');

INSERT INTO SEQUENCES (ID_SEQUENCE, TABLENAME, FIELD, SEQ_VALUE, INCREMENTVALUE,
MINIMUMVALUE, MAXIMUMVALUE, CIRCULAR) VALUES ( 3, 'VERSIONS', 'IDVERSION', 0, 1, 1,
999999999, 'T');

INSERT INTO SEQUENCES (ID_SEQUENCE, TABLENAME, FIELD, SEQ_VALUE, INCREMENTVALUE,
MINIMUMVALUE, MAXIMUMVALUE, CIRCULAR) VALUES ( 4, 'USERS', 'IDUSER', 0, 1, 1,
999999999, 'T');

INSERT INTO SEQUENCES (ID_SEQUENCE, TABLENAME, FIELD, SEQ_VALUE, INCREMENTVALUE,
MINIMUMVALUE, MAXIMUMVALUE, CIRCULAR) VALUES ( 5, 'GROUPS', 'IDGROUP', 0, 1, 1,
999999999, 'T');

```

To introduce these registers, again, we must use database manufacturer provided tools.

Once the database is installed and the tables created, according to the previous directions, we can proceed to the next steps of the installation.

5.1.4. Resources Table

This table stores the data of the available resources in the system. Resources are XML data files which are incorporated to the DataMovil smart client through the ‘src’ attribute of the ‘instance’ element. The

applications which have associated resources will be downloaded to the DataMovil smart client with all of its resources.

```
CREATE TABLE RESOURCES(
  ID VARCHAR(50) DEFAULT '' NOT NULL PRIMARY KEY,
  PATHTOSAVE VARCHAR(200) DEFAULT '' NOT NULL,
  NAME VARCHAR(200) DEFAULT '' NOT NULL,
  CODE LONG NOT NULL,
  EXPIRATIONDATE VARCHAR(50) DEFAULT NULL,
  FORMAT VARCHAR(50) DEFAULT NULL
)
```

The description of the fields of this table is the following:

- ID: Unique identifier of the resource.
- PATHTOSAVE: Absolute path to the directory where the resource will be stored in the device. This parameter may use the location variables defined in the data.ini file of the DataMovil smart client. For example, %LOCAL_USER_PATH%, %RESOURCE_HOME%, %ROM_STORAGE%, %SD_CARD%.
- NAME: Name of the file (in the previous directory) where the resource will be saved.
- CODE: Content of the resource. It must be a well-formed XML document.
- EXPIRATIONDATE: After this date, the resource will not be valid.
- FORMAT: Pattern expressing the format of the previous field. The pattern is the same as the pattern for dates in the Java language.

To create the table, the tools provided by the database we are working with should be used (i.e. SQLPlus for Oracle).

5.1.5. Associations Table

This table contains all the data required for the associations between applications and resources: UriDownload and CodeUpload.

```
CREATE TABLE ASSOCIATIONS(
  IDXFORM DECIMAL(8) DEFAULT '0' NOT NULL PRIMARY KEY,
```

```

URLDOWNLOAD VARCHAR(100) DEFAULT " NOT NULL,
CODEUPLOAD DECIMAL(1) DEFAULT '0' NOT NULL
)

```

The description of the fields of this table is the following:

- IDIFORM: The value of the field 'IDIFORM' in the table 'Applications'.
- URLDOWNLOAD: Url where the associated resources can be downloaded.
- CODEUPLOAD: Boolean property to indicate whether the old code of a resource must be uploaded before a new download of the resource.

To create the table, the tools provided by the database we are working with should be used (i.e. SQLPlus for Oracle).

5.1.6. Assoc_Res Table

This table contains the relationship between applications and resources. One application can have zero or more associated resources.

```

CREATE TABLE ASSOC_RES(
IDIFORM DECIMAL(8) DEFAULT '0' NOT NULL,
IDRESOURCE VARCHAR(50) DEFAULT " NOT NULL,
CONSTRAINT SYS_PK_ASSOC_RES PRIMARY KEY(IDIFORM,IDRESOURCE)
)

```

The description of the fields of this table is the following:

- IDIFORM: The primary key in the table "Associations".
- URLDOWNLOAD: The primery key in the table "Resources".

To create the table, the tools provided by the database we are working with should be used (i.e. SQLPlus for Oracle).

5.2. Administration

In order to install the DataMovil administration, it is needed to realize the deployment of the `adminDataMovil3.1.war` located in the directory `/DataMovilServer/war` of the DataMovil CD in the web or application server where the installation will be done.

As an example, assume that an Apache Tomcat server is installed in the following path: “C:\Program Files\Apache Group\Tomcat 4.1”

In this case to make the deployment, the only action needed is to copy the `adminDataMovil3.1.war` file in the directory “C:\Program Files\Apache Group\Tomcat 4.1\webapps”. The server itself will uncompress the file and install the application.

Once the deployment is done, a new directory called **adminDataMovil3.1** will be created under “C:\Program Files\Apache Group\Tomcat 4.1\webapps” where all the elements needed by the administration are located. A short description of the directories content follows:

- `css`: this directory contains the stylesheets for the administration pages.
- `js`: this directory contains the files with the JavaScript code needed for the correct functioning of the administration.
- `img`: this directory contains all the images used by the administration.
- `WEB-INF`: this directory contains the `web.xml` file that describes the administration application to the web server.
- `WEB-INF/lib`: this directory contains all the libraries used by the administration. **It is necessary to copy the JDBC driver library** of the database used with the administration (not included in the DataMovil CD). It is required to restart the web server after the library was copied.
- `WEB-INF/config`: this directory contains the `config.properties` file containing the configurations parameter of the administration.

It is necessary to adapt the initial configuration of the application to adjust it to the system where the DataMovil 3.1 server has been installed. This configuration is described in more detail in paragraph 5.2.1.

Lastly, restart the server so that all changes are loaded.

5.2.1. Administration Configuration.

Let us see a sample of a configuration file and we will describe the options based on this file.

```
#####
#          SERVER CONFIGURATION FILE 3.1          #
```

```
#####
# Version number.
version.number=3.1

# Country and language of administration

country=EN
language=en

# It specifies if version control will check the user identity to access to the
# downloading of applications.
main.users.active=true

##
### DB for storing applications
###
# Url and JDBC driver
main.database.url=jdbc:oracle:thin:datamovil/satec@claudia.malab.satec.es:1521:DBNRED
main.database.driver=oracle.jdbc.driver.OracleDriver
```

- version.number: specifies the version number.
- country: .specifies the country.
- language: specifies the language to show the texts in the administration interface.
- main.users.active: specifies whether the administration will use user and groups control. The possible values are: “true”, “false”
- main.database.url: string to connect to the database used to store the applications.
- main.database.driver: Java class implementing the JDBC driver.

5.2.2. Using custom plugins.

As described in the “DataMovil Programming Manual”, custom processors can be developed in order to handle the received data in a specific way.

To configure this feature follow these steps:

- Add a new app.receive.plugins:
 - Increase the value of app.receive.plugins.size.
 - Add a new block of classname – key. The key can have two kind of values, the beginning of a filename or the string appID#versID.
- Locate your compiled class/jar into the server classpath.
- Restart the server.

This plugin will execute when one of these conditions is true:

- There isn't a BD mapping already defined for this idApp#version. Mappings to BD have always preference over plugins.
- A received STORE_ITEM (see the Programming Manual for information) has the filename attribute and the value of this attribute starts with the value defined in the *key*.
- A received STORE_ITEM (see the Programming Manual for information) has the idXForm and idVersion attributes and the union (with the # separator) of the values of this attribute match with the value defined in the *key*.

5.2.3. The “received XML” directory.

This directory must be defined in the **app.receive.map.dir** parameter. The name of the files in this directory has this pattern:

```
tip_idApp_idVers_filename_randomNumber
```

The **tip** is the cause to write the file. Its values are:

- ok → When there is no plugin or mapping for the received data.
- errSQL or errMAP → When there is a mapping to automatically insert data into a database and an error appears.
- errPRO → When there is a plugin and an unrecoverable error appears.

The **idApp** and **idVers** are the received values for the application-id and version (primary key of each DataMóvil-xforms application).

The **filename** is the name of the file where data was previously stored into the DataMóvil device.

The **randomNumber** is a selfgenerated number. Filename for older versions of DataMovil included only this value.

5.3. Version Control and data reception.

The installation of the version control and data reception from the client are realized in the same way as the administration.

We make the deployment of the controlDataMovil3.1.war file that is located in the **/DataMovilServer/war** CD directory. Following the same steps as in the previous paragraph, we copy the file in the “C:\Program Files\Apache Group\Tomcat 4.1\webapps” directory what produces the

automatic decompression of the war file in a directory called controlDataMovil3.1. This directory is composed by the following subdirectories:

- WEB-INF\config: this directory contains the config.properties file containing the configurations parameters.
- \WEB-INF\lib: this directory contains all the libraries used by the application. **It is necessary to copy the JDBC driver library** of the database used with the administration (not included in the DataMovil CD). It is required to restart the web server after the library was copied.
- \WEB-INF\mappings: this directory contains the files used to map the data sent by the client to fields in tables of the database where the data will be saved.

Lastly, restart the server so that all changes are loaded.

5.3.1. Configuration

Let us see a sample of a configuration file and we will describe the options based on this file.

```
#####
#      VERSION CONTROL CONFIGURATION FILE DATAMOVI 3.1      #
#####
##
## DB for storing applications
##

# It specifies if version control will check the user identity to access to the
# downloading of applications.
main.users.active=false

# Url y driver JDBC
main.database.url=jdbc:oracle:thin:j2me/j2me@claudia.malab.satec.es:1521:DBNRED
main.database.driver=oracle.jdbc.driver.OracleDriver

##
## Plugins for specific handle of XMLs
##
app.receive.plugins.size=1

app.receive.plugins[0].classname=es.satec.xforms.ProcessorTest
app.receive.plugins[0].filenameKey=AminimalPetStore
app.receive.plugins[0].app#VersKey=0#0

##
## Mappings for the data reception servlet
##

# DataBase connection Url where the XML will be stored and the JDBC driver
app.receive.db.url=jdbc:oracle:thin:j2me/j2me@claudia.malab.satec.es:1521:DBNRED
app.receive.db.driver=oracle.jdbc.driver.OracleDriver

# Number of mappings
app.receive.map.size=1

# Directory where the 'map` files are located under the WEB-INF directory
app.receive.map.dir=mappings

# Directory where the XML files are stored if there is not an assigned mapping
app.receive.store.dir=/opt/adminDM2/xmls

app.receive.map[0].idApplication=0
```

```

app.receive.map[0].version=0
app.receive.map[0].name=0.map

#####

```

- `main.users.active`: specifies whether the administration will use user and groups control. The possible values are: “true”, “false”
- `main.database.url`: string to connect to the database where the applications will be stored.
- `main.database.driver`: Java class implementing the JDBC driver.
- `app.receive.db.url`: indicates the connection string to the database where the results sent by the client will be stored and there is a mapping file giving directions about how these data must be saved.
- `app.receive.db.driver`: Java class implementing the JDBC driver.
- `app.receive.plugins.size`: indicates the size of the mapping list.
- `app.receive.plugins[].classname`: full qualified name of the plugin class.
- `app.receive.plugins[].filenameKey`: beginning of the filename attribute of each received item.
- `app.receive.plugins[].app#VersKey`: union of the `idApp` and the version of the received item.
- `app.receive.map.size`: indicates the size of the mapping list.
- `app.receive.map.dir`: indicates the path where the data sent by the smart client will be saved if there is not a defined mapping for the application.
- `app.receive.map[].idApplication`: indicates the identifier of the application for which the mapping is valid.
- `app.receive.map[].version`: indicates the version of the application for which the mapping is established.
- `app.receive.map[].name`: indicates the name of the file containing the XML describing how the data sent by the client is mapped on the database.

5.3.2. Installation of the license file

The license file of DataMovil 3.1 must be installed in the libraries directory of the application called controlDataMovil3.1. Following the installation process described before, this directory can be found in controlDataMovil3.1/WEB-INF/lib from the directory for the web applications of the Tomcat server (in our example, the complete route is the following: C:\program files\Apache Group\Tomcat 4.1\webapps\controlDataMovil3.1\WEB-INF\lib)

The license file is found in the **/DataMovilServer/license** directory of the installation CD. It is the **license.jar file**. In case this file is absent, contact the DataMovil support to obtain one and copy it in the right location.

Once the license file is copied in the corresponding directory, it is necessary to **restart** the web server for the changes to take into effect.

In case the license file is downloaded through the web or received by any other electronic means, the installation procedure is the same as described before.

5.3.3. Using server custom plugins.

As described in the “DataMovil Programming Manual”, custom processors can be developed in order to handle the received data in a specific way.

To configure this feature follow these steps:

- Add a new app.receive.plugins:
 - Increase the value of app.receive.plugins.size.
 - Add a new block of classname – filenameKey – app#VersKey. At least one of filenameKey or app#versKey is required. The purpose of having two kind of keys is to have one kind of mapping to process both “live-submissions” and “deferred-submissions”.
- Locate your compiled class/jar into the server classpath.
- Restart the server.

This plugin will execute when one of these conditions is true:

- There isn't a BD mapping already defined for this idApp#version. Mappings to BD have always precedence over plugins.
- A received STORE_ITEM (see the Programming Manual for information) has the filename attribute and the value of this attribute starts with the value defined in the *filenameKey*.
- A received STORE_ITEM (see the Programming Manual for information) has the idXForm and idVersion attributes and the union (with the # separator) of the values of this attribute match the value defined in the *app#VersKey*.

5.3.4. The “received XML” directory.

This directory must be defined in the **app.receive.map.dir** parameter. The name of the files in this directory has this pattern:

```
tip_idApp_idVers_filename_randomNumber
```

The **tip** indicates the result status of the submission operation. Its values are:

- ok → When there is no plugin or mapping for the received data.
- errSQL or errMAP → When there is a mapping to automatically insert data into a database and an error appears.
- errPRO → When there is a plugin and an unrecoverable error appears.

When the file is correctly inserted into a database or correctly managed by a plugin, no file will be written into the “received XML” directory.

The **idApp** and **idVers** are the received values for the application-id and version (primary key of each DataMovil-xforms application).

The **filename** is the name of the file where data was previously stored into the DataMovil device.

The **randomNumber** is a selfgenerated number. Filename for older versions of DataMovil included only this value.

5.4. Users and groups management

DataMovil allows us to install a users management tool. This tool offers functionality to control the access of users to the applications created with the administration. The users are organized into groups and the groups of users are granted execution rights to applications. In this manner we can define which users belong to a group and which applications can be executed by a group. The user must be identified by the web server, therefore, **the web server users must be the same as the DataMovil users**. The synchronization between both users tables can be done manually or automatically with tools that will depend in great extent on the existing infrastructure where DataMovil will be installed (text files, LDAP directory, Active directory). This is the reason for not providing a particular solution to this problem.

The installation of the users and groups management tool is done in three steps:

1. Creation of the needed tables in the database.
2. Deploy of the web application in the web or application server.
3. Configuration of the application.

5.4.1. Creation of the tables

We need to create four tables in the database to support the users and groups management tool. We show the scripts for the creation of the four tables.

Table USERS

```
CREATE TABLE USERS (  
  IDUSER decimal(8) default '0' NOT NULL,  
  LOGIN VARCHAR(10) NOT NULL,  
  NAME VARCHAR(255) NOT NULL,  
  LASTNAME VARCHAR(255) NOT NULL,  
  PRIMARY KEY (IDUSER));
```

The description of the fields is:

- IDUSER: unique identifier of each user.
- LOGIN: username to access the web server
- NAME: name of the user
- LASTNAME: family name of the user.

Table GROUPS:

```
CREATE TABLE GROUPS (  
  IDGROUP decimal(8) default '0' NOT NULL,  
  NAME VARCHAR (255) NOT NULL,  
  PRIMARY KEY (IDGROUP));
```

The description of the fields is:

- IDGROUP: unique identifier of each group.

- NAME: name of the group.

Table USERGROUPS:

```
CREATE TABLE USERGROUPS (
  IDUSER decimal(8) default '0' NOT NULL,
  IDGROUP decimal(8) default '0' NOT NULL,
  PRIMARY KEY (IDUSER, IDGROUP));
```

The description of the fields is:

- IDUSER: user identifier.
- IDGROUP: group identifier.

Table GROUPSAPP

```
CREATE TABLE GROUPSAPP (
  IDGROUP decimal(8) default '0' NOT NULL,
  IDAPP decimal(8) default '0' NOT NULL,
  PRIMARY KEY (IDGROUP, IDAPP));
```

The description of the fields is:

- IDGROUP: group identifier
- IDAPP: application identifier.

5.4.2. Installation of the web application

in order to install the users management tool, it is necessary to make the deployment of the file usersDataMovil3.1.war located in the directory **/DataMovilServer/war** of the distribution CD in the web server or application server where the application will be installed.

As an example we assume that the installation will be done in an Apache Tomcat server installed at the path: “C:\Archivos de programa\Apache Group\Tomcat 4.1”

In this case to make the deploy, we only need to copy the usersDataMovil3.1.war file to the directory “C:\Archivos de programa\Apache Group\Tomcat 4.1\webapps”. The Tomcat server will decompress the application and will install the application.

Once the deploy is done, a new directory in the path “C:\Archivos de programa\Apache Group\Tomcat 4.1\webapps”, called **usersDataMovil3.1**, is created. This directory contains all the files needed for the management tool. The following directories were created:

- img: directory for all the files containing the images used by the tool.
- WEB-INF: this directory contains the web.xml file that describes the application for the web server.
- WEB-INF/lib: this directory contains all the libraries used by the tool. It is required that the **JDBC driver** to access the database **is copied to this directory** (the JDBC driver is not provided in the CD). It is necessary **to restart** the web server after the driver was copied.
- WEB-INF/config: this directory contains the config.properties file that includes all the configuration parameters of the tool.

5.4.3. Configuration of the application

The initial configuration parameters must be adapted to the system where DataMovil 3.1 is installed.

This is an example of configuration file. The registers are described below.

```
#####
# CONFIGURATION FILE OF THE USERS AND GROUPS MANAGEMENT TOOL DATAMOVIL 3.1 #
#####
#
##
## Users and groups database
##

# Url and JDBC driver
main.database.url=jdbc:oracle:thin:j2me/j2me@claudia.malab.satec.es:1521:DBNRED
main.database.driver=oracle.jdbc.driver.OracleDriver
```

- main.database.url: string to connect to the database where the users and groups tables have been created.
- main.database.driver: class implementing the JDBC driver.

To complete the activation of the users and groups management tools, the administration tool and the version control servlet must be warned that they have to work with user access control. This is done setting the **main.users.active** parameter to **true** in the configuration file of both applications.

The last step is to restart the server so that all changes are activated.

6. Installation and configuration of the DataMovil smart client.

6.1. Installation in PocketPC

The installation in a PDA with the PocketPC operating system is realized following these steps:

- Connect the PDA to the PC through the cradle. The ActiveSync must be active.
- Execute DataMovil3.1.exe file, located in the /DataMovilBrowser/PocketPC directory of the DataMovil CD. This will automatically install the DataMovil client in the PDA.
- Edit the data.ini file installed in the PDA. In order to do this, there is a link to an application called “**EditorIni**” under the Windows “Start” button. This program runs in the PC and it is used to modify the data.ini file located in the PDA, by default in the directory “**Program Files\DataMovil**”. In order to perform the edition, the PDA and the PC must be connected through **ActiveSync**. In a default installation the link to the “**EditorIni**” application is under the folder “**Start/Programs/Satec/EditorIni**.. With this application we can modify any parameter of the DataMovil data.ini file. Any changes to the data.ini file must be carefully designed prior to the edition of the file. An error in the edition may provoke that the client does not work correctly. Nevertheless, modifying only those registers shown in the basic mode of operation should be enough in most cases to make the DataMovil client work.

6.2. Installation in MIDP devices

The installation process has two steps: configuration of the application and installation in the device.

To configure the application:

1. Open the JAR file (located at /DataMovilBrowser/MIDP) with one ZIP utility, or extract the files using the java “jar” utility.
2. Edit the configuration files (data.ini, and messages_en.ini or messages_es.ini).
3. Save the modified configuration file.
4. If all files were extracted with the java “jar” utility, a new DataMovil.jar file must be created.
5. Edit the JAD file and edit the line “MIDlet-Jar-Size:”, setting the new JAR file size in bytes (the jar size must be typed without the thousand delimiter char).

To deploy the application on the devices, the best option is OTA (Over The Air).

1. Put both files, JAR and JAD, in a web server.
2. Check that the mime-type “text/vnd.sun.j2me.app-descriptor” is supported by the server. If not, configure it in a form similar to “type=text/vnd.sun.j2me.app-descriptor exts=jad”.
3. Download the JAD URL in the mobile device, using the direct link cable (if available) or any kind or wireless connection (GPRS, UMTS, WiFi...).

6.3. Installation in BlackBerry devices

The **/DataMovilBrowser/BlackBerry/** directory contains client binaries for Blackberry. Several files are provided, built with different versions of the BlackBerry development tools. Each binary is in a subdirectory named like the version number of the BlackBerry JDE used.

Deployment can be done in two ways: by a direct link cable or by OTA like for the MIDP devices. For the first option, an utility script is provided: **/DataMovilBrowser/BlackBerry/4.x/deploy.bat** (where 4.x is the appropriate JDE version number).

6.4. Configuration files for POCKETPC devices.

There are three configuration files: data.ini, messages_en.ini and messages_es.ini. The first one is the properties file and the others are for localized text.

6.4.1. Data.ini

The Data.ini file keeps the configuration parameters of the DataMovil smart client. It is divided in logical sections depending on the purpose of the parameters groups. The value of the parameters must be configured according to the server and client installation.

The data.ini file is divided into the following sections:

- [MAIN]: General client related parameters (urls, filesystem, locale, date formats...).
- [LAYOUT]: Contains a set of default graphical layout related parameters (margin, alignment...).
- [CONNECTION]: TimeOut and SSL configuration
- [LOOK]: Contains the default graphical configuration parametes (color, fonts...).
- [FLAGS]: These parameters can modify the default behaviour of DataMovil.

Special attention to the following parameters of the MAIN section must be paid:

- SERVER_URL
- VERSIONS_URL
- FS_USER_PATH
- FS_CUEST_PATH
- FS_ENCUCU_PATH

Incorrect values of these parameters will make the system not to work.

A description of all the parameters follows, showing sample values and default values included in the delivery.

6.4.1.1 MAIN

SERVER_URL	Server URL where the DataMovil smart client initially connects. SERVER_URL=http://goliat.malab.satec.es:83
VERSIONS_URL	Directory relative to SERVER_URL where the DataMovil version control servlet is located. This servlet will be automatically created when the deployment of the “war” file is made. For this reason, the first component of the directory will depend on the name of the “war” file. VERSIONS_URL = /applications/versionControl
SEND_URL	Server URL to send the application data that were saved locally in the PDA. SEND_URL=http://goliat.malab.satec.es:83/aplicaciones/appReceive
SENDING_METHOD	It specifies the method to send the data to the previous URL. SENDING_METHOD = POST
FS_USER_PATH	Directory of the PDA where all the application information is locally saved. FS_USER_PATH = \\windows\\lib
FS_CUEST_PATH	Directory relative to FS_USER_PATH where the applications are locally saved. FS_CUEST_PATH = cuest2
FS_ENCUCU_PATH	Directory relative a FS_USER_PATH where data applications are locally saved FS_ENCUCU_PATH = encuc2
FS_SD_CARD	Name of the directory associated with an sd-card in a PDA. FS_SD_CARD=sd_card

ROM_STORAGE	Name of the directory associated with the ROM storage of the PDA. ROM_STORAGE =sd_card
FILE_DEFAULT_NAME*	Default name of the file where the data will be saved after a failed connection attempt. * FILE_DEFAULT_NAME = file\$jnow(yyyyMMMdd-HH-mm-ss)\$
FS_SESSION_DEFAULT_NAME*	Default name of the file where the session of the application will be saved. The format is the same as for FILE_DEFAULT_NAME. SESSION_DEFAULT_NAME = session\$jnow(yyyyMMMdd-HH-mm-ss)\$
INIT_FORM	Application loaded by default when DataMovil is executed. If left blank, DataMovil shows the welcome screen. INIT_FORM =
LANGUAGE	Language used to process date and time information. It is coded according to ISO-639. LANGUAGE = en
COUNTRY	Country used as a basis for the processing of date and time. It is coded according to ISO-3166. COUNTRY = US

6.4.1.2 LAYOUT

ALIGNMENT	Default value for the horizontal alignment of the controls. Allowed values are 0 (left), 1 (right) and 2 (center). ALIGNMENT = 0
VALIGNMENT	Default value for the vertical alignment of the controls. Allowed values are 0 (top), 1 (bottom) and 2 (middle). VALIGNMENT = 0
MARGIN	Default value for the component's margins. MARGIN_TOP = 5
SPACING	Size of the horizontal and vertical distances between elements within a component like lists of options, XML representations on the screen or the spacing among the elements of a table. SPACING = 5
CELLPADDING	Distance between cells in a repeat component. CELLPADDING=5

* The grammar for this field is: g := (\$APPNAME\$ | \$jnow(...)\$ | char)*

Any combination of characters (except . _ \ | / : * ? " < >), the jnow (defined in the Programming Manual) and the \$APPNAME\$ variable, which will be replaced with the application name..

CELLSPACING	Distance between cell border and cell component in a repeat component. CELLPADDING=5
TAB_SIZE	Tab size measured in pixels TAB_SIZE = 12

6.4.1.3 LOOK

COLOR_BG_MAIN	Background color of the panel for buttons. Format: R G B, where R, G and B are in the range 0-255 COLOR_BG_MAIN=210 210 210
COLOR_BG_MAINPNL	Background color of the panel where Xform controls are shown. Format: R G B, where R, G and B are in the range 0-255 COLOR_BG_MAINPNL=255 255 255
COLOR_BG_BUTTON	Background color of the buttons. Format: R G B, where R, G and B are in the range 0-255 COLOR_BG_BUTTON = 151 172 196
COLOR_FG_TEXT	Font color for all items drawn in the screen. Format: R G B, where R, G and B are in the range 0-255 COLOR_FG_TEXT = 91 123 159
COLOR_FG_BUTTON	Font color for all buttons drawn in the screen. Format: R G B, where R, G and B are in the range 0-255 COLOR_FG_TEXT = 255 255 255
FONT_FACE	“font” type to be used by all texts. The “fonts” permitted are those supported by the JVM. FONT_FACE=arial
FONT_BUTTON_SIZE	“font” size of the buttons of the text. FONT_BUTTON_SIZE = 10
FONT_TITLE_SIZE	“font” size of the text of the initial screen that is shown as DataMovil is launched. FONT_TITLE_SIZE = 22
FONT_LABEL_SIZE	“font” size of the text included in the <LABEL> component. FONT_LABEL_SIZE = 16
FONT_CHOICES_SIZE	« font » size of the text NOT included in the <LABEL> component FONT_CHOICES_SIZE = 12
FONT_PROGRESS_SIZE	« font » size of the text shown in the connection progressbar. FONT_PROGRESS_SIZE = 12

6.4.1.4 CONNECTION

SSL_SEND_EMPTY_FRAGMENT	<p>If the value is 1, then the SSL layer sends an empty fragment while the handshake phase.</p> <p>SSL_SEND_EMPTY_FRAGMENT = 1</p>
TIMEOUT	<p>Timeout (in seconds) for the connections realized by the DataMovil client.</p> <p>CONNECTION_TIMEOUT = 30</p>

6.4.1.5 FLAGS

VERSION_CONTROL	<p>Flag to control whether DataMovil saves locally all versions of an application or only the latest one. Setting the flag to 1 will save only the latest. Setting the flag to 0 will save all versions.</p> <p>VERSION_CONTROL=1</p>
AUTO_SEND	<p>Flag to control whether DataMovil sends the local store after a successful connection. Setting this flag to 0, the local store only will be sent manually.</p> <p>AUTO_SEND = 0</p>
AUTOMATIC_DOWNLOAD	<p>Flag to specify whether DataMovil downloads the new applications or new versions of applications at the time a successful connection to the server is done. Setting the flag to 0 does not download the applications. Setting the flag to 1 downloads the applications.</p> <p>AUTOMATIC_DOWNLOAD = 0</p>
USE_ZIP	<p>Flag to specify whether DataMovil will compress in zip format the data sent to the server or will transmit uncompress.</p> <p>Setting the flag to 1 will compress the information.</p> <p>Setting the flag to 0 will transmit uncompress.</p> <p>USE_ZIP = 0</p>
BUTTONS	<p>Flag to hide the buttons at the bottom of the client screen. The value 1 will show the buttons. The value 0 will hide these buttons.</p> <p>BUTTONS=1</p>
FLAG_DATAMOVIL_MENU	<p>Flag to hide the menu at the menubar of the client screen. The value 1 will show the menu. The value 0 will hide this menu.</p> <p>DATAMOVIL_MENU=1</p>
AUTOSAVE	<p>Flag to save the data that couldn't be sent to the server without user interaction. If the value is 1 and the parameter FILE_DEFAULT_NAME has been set, then the data will be automatically saved.</p> <p>If the value is 0 or the parameter FILE_DEFAULT_NAME has not been set, then the data will not be automatically saved</p> <p>FLAG_AUTOSAVE=0</p>

6.4.2. Variable Names

The following variables may be employed in DataMovil-Xforms applications to reference PDA file paths in the ‘src’ attribute of the ‘instance’ element.

%LOCAL_USER_PATH%, whose value is the LOCAL_USER_PATH parameter of the data.ini file

%RESOURCE_HOME%, whose value is

%LOCAL_USER_PATH%/cache/⟨⟨appname_appID_appversion⟩⟩/resource

%ROM_STORAGE%, whose value is the ROM-STORAGE parameter of the data.ini file

%SD_CARD%, whose value is the SD_CARD parameter of the data.ini file

6.4.3. Messages_es.ini / Messages_en.ini

TXT_TITLE_MAIN	This message is the title of the application window. TXT_TITLE_MAIN = DataMovil 3.0
TXT_LAPP_WELCOME	This text is the welcome message in the “Local Applications” screen. TXT_LAPP_WELCOME = Welcome to DataMovil 3.0
TXT_LAPP_TITLE	This text is the title of the “Local Applications” screen. TXT_LAPP_TITLE = Local Applications:
TXT_LAPP_NOAPP	This message is shown when there are not local applications stored in the device. TXT_LAPP_NOAPP = There are no applications.
TXT_STORE_TITLE	This text is the title of the “Local Store” screen. TXT_STORE_TITLE = Stored data:
TXT_STORE_CHECKALL	Text for the check button to select/deselect all stored items. TXT_STORE_CHECKALL = Select All/None
TXT_STORE_NODATA	This message is shown when there is no data stored in the local store. TXT_STORE_NODATA = There are not data stored.
TXT_STORE_SENDOK	Text shown when the data is sent correctly. TXT_STORE_SENDOK = Data sent correctly
TXT_STORE_SENDKO	Text shown when the data was not sent correctly. TXT_STORE_SENDKO = Error while sending data.

TXT_RAPP_TITLE	This text is the title of the “Remote Applications” screen. TXT_RAPP_TITLE = Server applications:
TXT_RAPP_NOTHING	This message is shown when there are not new versions or applications in the server. TXT_RAPP_NOTHING = There are not new versions or applications in the server
TXT_RAPP_ERR	This text is shown when the new version download fails. TXT_RAPP_ERR = The application could not be downloaded
TXT_RAPP_EXPIRED_LICENCE	This text is shown when the server license is invalid. TXT_RAPP_EXPIRED_LICENCE = The license has expired or it is not valid
TXT_SESSIONS_TITLE	This text is the title of the “Application Sessions” screen. TXT_SESSIONS_TITLE = Stored sessions:
TXT_SESSIONS_ERR_LOAD	This message appears when a session can’t be loaded. TXT_SESSIONS_ERR_LOAD = Session can't be loaded.
TXT_SESSIONS_ERR_SAVE	This message appears when a session can’t be saved. TXT_SESSIONS_ERR_SAVE = Session can't be saved.
TXT_XML_TITLE	This text is the title of the “Stored Item Viewer” screen. TXT_XML_TITLE = XML Viewer:
TXT_XML_ERR	This text is shown when the XML viewer can’t display a file. TXT_XML_ERR = XML Error.
TXT_BTN_OK	This is the label for “Ok” buttons. TXT_BTN_OK = OK
TXT_BTN_YES	This is the label for “Yes” buttons. TXT_BTN_YES = Yes
TXT_BTN_NO	This is the label for “No” buttons. TXT_BTN_NO = No
TXT_BTN_CANCEL	This is the label for “Cancel” buttons. TXT_BTN_CANCEL = Cancel

TXT_BTN_RUN	This is the label for “Execute” buttons. TXT_BTN_RUN = Execute
TXT_BTN_DEL	This is the label for “Delete” buttons. TXT_BTN_DEL = Delete
TXT_BTN_SEND	This is the label for “Send” buttons. TXT_BTN_SEND = Send
TXT_BTN_SAVE	This is the label for “Save” buttons. TXT_BTN_SAVE = Save
TXT_CNX_ERROR	This message appears when a connection fails. TXT_CNX_ERROR = Error
TXT_CNX_GETSRC	This is the title of the “Connection ProgressBar” when DataMovil is trying to download a src attribute. TXT_CNX_GETSRC = Downloading SRC
TXT_CNX_GETMEDIA	This is the title of the “Connection ProgressBar” when DataMovil is trying to download a media element. TXT_CNX_GETMEDIA = Downloading Media
TXT_CNX_GETRESOURCE	This is the title of the “Connection ProgressBar” when DataMovil is trying to download a resource file. TXT_CNX_GETRESOURCE = Downloading Resource
TXT_CNX_SUBMIT	This is the title of the “Connection ProgressBar” when DataMovil is trying to send a submission element. TXT_CNX_SUBMIT = Submission in progress
TXT_CNX_CANCEL_TITLE	This is the title of the dialog that notifies the user cancellation of the connection operation. TXT_CNX_CANCEL_TITLE = Cancelled
TXT_CNX_CANCEL_TXT	This is the body of the dialog that notifies the user cancellation of the connection operation. TXT_CNX_CANCEL_TXT = Cancelled by user.
TXT_CNX_UPLOAD	This text is drawn over the ProgressBar while an upload is being made. TXT_CNX_UPLOAD = Uploading...
TXT_CNX_DOWNLOAD	This text is drawn over the ProgressBar while an download is being made. TXT_CNX_DOWNLOAD = Downloading...
TXT_CNX_LOGIN_TITLE	This is the title of the dialog to get the pair user-password. TXT_CNX_LOGIN_TITLE = Authentication

TXT_CNX_LOGIN_USUARIO	This is the label of the textfield where the user can type its username. TXT_CNX_LOGIN_USUARIO = User:
TXT_CNX_LOGIN_PASSWORD	This is the label of the textfield where the user can type its password. TXT_CNX_LOGIN_PASSWORD = Password:
TXT_MIPS_ERROR_TITTLE	This is the title of the dialog that appears when there is a problem with the “Model Item Properties” in the current screen. TXT_MIPS_ERROR_TITTLE = ERROR - MIPS
TXT_MIPS_ERROR_TXT	This is the body of the dialog that appears when there is a problem with the “Model Item Properties” in the current application screen. TXT_MIPS_ERROR_TXT = Invalid Component
TXT_PROC_EOF	This message appears when the user tries to go after the last page of the application. TXT_PROC_EOF = End of application
TXT_PROC_BOF	This message appears when the user tries to go before the first page of the application. TXT_PROC_BOF = Start of application
TXT_PROC_EXC	This token is the first item shown when a generic error occurs. TXT_PROC_EXC = Error
TXT_MNU	This is the label of the root menuitem of the menubar. TXT_MNU = DataMovil
TXT_MNU_EXIT	This is the label of the menuitem to exit of the application. TXT_MNU_EXIT = Exit
TXT_MNU_APPS	This is the label of the menuitem which is the parent of “Local” and “Remotes” submenús. TXT_MNU_APPS = Applications
TXT_MNU_APPS_LOCALS	This is the label of the menuitem to go to the “Local Applications” screen. TXT_MNU_APPS_LOCALS = Locals
TXT_MNU_APPS_REMOTES	This is the label of the menuitem to go to the “Remote Applications” screen. TXT_MNU_APPS_REMOTES = Remotes
TXT_MNU_STORE	This is the label of the menuitem to go to the “Local Store” screen. TXT_MNU_STORE = Store
TXT_MNU_SESSIONS	This is the label of the menuitem to go to the “Stored Sessions” screen. TXT_MNU_SESSIONS = View Sessions

TXT_MNU_SAVE	This is the label of the menuitem to go to the “Save the Session” of the running application. TXT_MNU_SAVE = Save Session
TXT_QST_EXIT	This question is shown before the exit from DataMovil. TXT_QST_EXIT = Do you want to exit?
TXT_QST_DELETE	This question is shown before delete an application. TXT_QST_DELETE = This application will be deleted:
TXT_QST_DELETE_DATA	This question is shown before delete of a stored item. TXT_QST_DELETE_DATA = Do you want to delete the selected data?
TXT_QST_LEAVE_APP	This question is shown before exit from an application. TXT_QST_LEAVE_APP = You are going to leave the application. Do you want to save the session?
TXT_QST_FILENAME	This question is shown to require a filename to the user. TXT_QST_FILENAME= Type a filename (without . _ \\ / : * ? " < >):
TXT_QST_OVERWRITE	This question is shown before overwrite a file. TXT_QST_OVERWRITE = The file already exists. Overwrite?
TXT_QST_TOSTORE	This question is shown to ask the user about save an item to the local store.. TXT_QST_TOSTORE = Do you want to save the data into the local store?

6.4.4. DMLauncher.ini

This file is used by the executable file that launches the DataMovil application in PocketPC environment. It is located (in a default installation) at “Program Files/DataMovil”.

It has two sections: JVM (for system related parameters) and PLUGINS (for plugins classpaths).

6.4.4.1 JVM

exec	Absolute path to the JVM executable file. For JBed: exec=\Windows\evm.exe For IBM J9: exec=\Archivos de programa\J9\PPRO11\bin\j9w.exe
params	This parameters will be appended to the previous executable file. For JBed: params=-Xnowinceconsole -Xsplash:"\Windows\logoParaJBed.bmp" For IBM J9: params=-jcl:ppro11 -Dcom.ibm.oti.awt.PocketPCWindowBehavior=false
memory	This is the maximum size (in megabytes) for the java heap. memory=40

6.4.4.2 PLUGINS

This section is composed by many “pluginX=absolutePath” lines, where X is a number from 1 to N, and absolute path is the path to de class file, or jar package or a directory. These paths will be appended to the classpath of DataMovil.

For example:

```
[PLUGINS]
plugin1=/myPlugins/PassportChecker.class
plugin2=/myPlugins/BirthDayRemember.jar
plugin3=/myPlugins
```

6.5. Configuration files for MIDP/BlackBerry devices.

For basic configuration at runtime, the SysInfo MIDlet provides a setup screen which allows the user to customize the basic data.ini variables of the deployed client. This configuration interface is accessible through the “Config” command in the SysInfo main screen.

Depending on the device, configuration files may not be visible at runtime. If that is the case, configuration files can still be extracted from the JAR and replaced with customized versions before deployment. **Note: this is not possible in the BlackBerry binary because of its proprietary format.**

6.5.1. Data.ini

This file is equal to the PocketPC version, in the 6.4.1 section.

6.5.2. Variable Names

The variable names are the same as for PocketPC version, in the 6.4.2 section.

6.5.3. Messages_es.ini / Messages_en.ini

title.main	This is the title that will be displayed in every screen. title.main = DataMovilME 3.1
cmd.*	Commands are usually activated using the soft buttons in the handset.. cmd.exit = Exit cmd.close = Close cmd.cancel = Cancel cmd.load = Run cmd.send = Send cmd.download = Download cmd.delete = Delete cmd.start = Start cmd.next = Next cmd.prev = Prev cmd.edit = Edit cmd.ok = OK cmd.back = Back cmd.snapshot = Save session cmd.filename = Save

	<p>cmd.remote = Remote cmd.send_one = Send cmd.send_all = Send all cmd.jump = Jump to cmd.change_app = Change app cmd.session_in = Saved states cmd.pending_in = Pending store</p>
confirmation.*	<p>Messages of the confirmation dialogs prompting the user to accept or cancel an action.</p> <p>confirmation.close_app= Close application? confirmation.save_pending= Save pending submission? confirmation.override= Overwrite file? confirmation.session_delete= Delete session? confirmation.app_delete= Remove application? confirmation.pending_delete= Delete pending submission?</p>
alert.*	<p>Non-interactive screens that report about the result of user actions.</p> <p>alert.submit.error= Could not send (pending) alert.submit.sent= Sent. alert.end_of_form= Screen not available. alert.mips.error= Invalid data. alert.download.error= Error in connection. alert.license.error = License expired or not valid.</p>
gauge.*	<p>Titles for the progress indicators for application loading and data submission.</p> <p>gauge.loading= Loading... gauge.ok= OK gauge.submit= Sending...</p>
ticker.*	<p>Scrolling text that reports control refresh activity in background.</p> <p>ticker.refresh= Refreshing...</p>
input.*	<p>Label of text field in file dialog:</p> <p>input.filename.label= Filename:</p>
choice.*	<p>Labels of choicegroup controls in application management screens:</p> <p>choice.local_apps.label=Local apps: choice.remote_apps.label=Remote apps: choice.states.label=Saved sessions: choice.pending.label=Pending submissions:</p>
stringitem.*	<p>Miscellaneous strings used in non-interactive text items.</p> <p>stringitem.retrieving=Downloading stringitem.done=OK. stringitem.filedialog.pending=Filename for pending submission. stringitem.filedialog.session=Filename for session. stringitem.filedialog.warning=Invalid filename. stringitem.application.label=Application stringitem.cancelled=Cancelled by user stringitem.initializing=Initializing stringitem.noapps=No applications stringitem.nosessions=No saved sessions stringitem.nopending=No pending submissions</p>

7. Installation of the applications editor.

The installation of the editor is very simple. We only have to follow the assistant instruction.

To execute the installation assistant, we go to the CD directory *DataMovilEditor*. Within this directory, there is a file called *EditorDM.exe*. Then, we double click on the executable file and follow the instructions.

Once the installation is finished, we can execute the editor from the link created in Init Menu, in the section *Satec\EditorDM3.1*.

8. Installation of the evaluation version.

The DataMovil evaluation version includes, in a single package, all what is needed to test the functionalities of the product. This version includes a web server with some preinstalled applications: administration and version control. It also includes a database where the applications will be stored, an applications editor and a simulator of the clients to be installed in the PDA and in the MIDP device. All of them ready to use from the desktop. Both samples, PetStore and SmokinHabits, are included too.

The installation of all the software is done in a very simple way, using an installation assistant. We only have to follow the directions displayed on the screen. The assistant only request three data from the user.

The first one is the selection of the JDK that will use the servers (web server and database server) and the Editor to run. The default JDK that is installed in the system, will be selected by default but the use has the possibility to select a different JDK (NOTE: JDK must be 1.5 or higher).

After that, the user will be prompted to enter the port where the web server and the database will be listening. Default values are also available. It is important that we choose port numbers that are not being used, because if they are in use by other application, DataMovil will not work.

Once the assistant has finished the installation, if we go to the Init → Programs menu, under Satec\DataMovil we have direct links available to start and stop the servers, start the clients (the PDA or the MIDP version) and a link to the administration that opens the web navigator and displays the administration.

The last step to perform the installation is to copy the license.jar file to the server. In a default installation, this file must be copied to this directory:

```
c:\Program Files\Satec\DataMovil\jetty\webapps\controlDataMovil3.1\WEB-INF\lib
```

In this case we do not need to do all the steps performed before, because the database tables are already created and the web applications installed by default.

We do not need either to install the client in the PDA. A windows simulator is included that runs at the desktop and may connect to the server and download applications, execute them and send back data to the server.

The examples described in the next chapter are also included, what makes unnecessary to carry out the installation step for the examples.

Hints about the MIDP simulator:

- The simulator file system is cleared by the WTK between executions in some WTK versions. In the WTK 2.5 final release, the filesystem is persistent between executions.

- Like a real MIDP device, the simulator requires an user confirmation before each file system access. To grant filesystem access for ever, we must change a parameter in the WTK preferences. To do this, go to the “**Preferences**” item in the “Sun Java(TM) Wireless Toolkit 2.5 for CLDC” program group. Then, go to the “**Security**” category and change the “**Security domain**” value to “**manufacturer**”..

9. Installation and execution of the sample applications.

9.1. PetStore

We explain hereafter the installation procedure of the well-known PetStore application ported to DataMovil.

The first step is to install the server part of the application in the web server. In order to do so we have to make the deployment of the file **PetStore.war**, included in the distribution CD in the directory **/Samples/PetStore/Server**.

The deployment is made following the same procedure as described in paragraphs 5.2 y 5.3 for the control versions and administration **war** files.

Once the deploy is made, it is necessary to modify the **web.xml** file. The **url** parameter must address the web server where we have installed this sample, and the **dateFormat** parameter must be an *xsd:date*:

```
<init-param>
  <param-name>url</param-name>
  <param-value>http://goliat.malab.satec.es:1234/PetStore</param-value>
  <description>Port to be used</description>
</init-param>
<init-param>
  <param-name>dateFormat</param-name>
  <param-value>yyyy-MM-dd</param-value>
  <description>Dateformat to be used</description>
</init-param>
```

Optionally, the **log4j.cfg** file may be modified if we want to set a path for the log file different from the default file.

The second step consists in changing the file where the application is defined. This file can be found in the directory **/Samples/PetStore/DataMovil-XForms** of the distribution CD. The name of the file is **PetStore_1_1**. We edit this DataMovil-XForms ASCII file and look for the attribute “action” of the “submission” tag. The URL included there must be changed to the URL route where we have installed the server part of the application.

We also need to modify the “src” attributes of the “media” tags (located in a block delimited by the “mediablock” tag). The identifiers to modify are “title”, “bulldog”, “finch”, “golden”, “iguana”, “koi”, “parrot”, “persian”, “rattlesnake”, “shark” and “tailless”.

The modification consists in setting the right address where the application was installed so that it points to the images (for example, <http://goliat.malab.satec.es:1234/PetStore/TituloPetStore.gif>). This block would be:

```

<mediablock>
  <media id="title"
    src="http://goliat.malab.satec.es:2105/PetStore/TituloPetStore.gif"
    preload="true" mediatype="image/gif"/>
  <media id="bulldog"
    src="http://goliat.malab.satec.es:2105/PetStore22/images/bulldog.jpg"
    "      preload="true" mediatype="image/jpg"/>
  <media id="finch"
    src="http://goliat.malab.satec.es:2105/PetStore22/images/finch.jpg"
    preload="true" mediatype="image/jpg"/>
  <media id="golden"
    src="http://goliat.malab.satec.es:2105/PetStore22/images/golden.jpg"
    preload="true" mediatype="image/jpg"/>
  <media id="iguana"
    src="http://goliat.malab.satec.es:2105/PetStore22/images/iguana.jpg"
    preload="true" mediatype="image/jpg"/>
  <media id="koi"
    src="http://goliat.malab.satec.es:2105/PetStore22/images/koi.jpg"
    preload="true" mediatype="image/jpg"/>
  <media id="parrot"
    src="http://goliat.malab.satec.es:2105/PetStore22/images/parrot.jpg"
    preload="true" mediatype="image/jpg"/>
  <media id="persian"
    src="http://goliat.malab.satec.es:2105/PetStore22/images/persa.jpg"
    preload="true" mediatype="image/jpg"/>
  <media id="rattlesnake"
    src="http://goliat.malab.satec.es:2105/PetStore22/images/rattlesnake
    .jpg"      preload="true" mediatype="image/jpg"/>
  <media id="shark"
    src="http://goliat.malab.satec.es:2105/PetStore22/images/shark.jpg"
    preload="true" mediatype="image/jpg"/>
  <media id="tailless"
    src="http://goliat.malab.satec.es:2105/PetStore22/images/tailless.jp
    g"      preload="true" mediatype="image/jpg"/>
</mediablock>

```

Once we have made this change, we copy the file to the PDA. This file must be copied to the directory where the applications are stored. This directory is defined in the **data.ini** file of the of the DataMovil smart client in the field LOCAL_APPL_DIRECTORY relative to the LOCAL_USER_PATH field. If the “src” attribute of the “media” tag references a file (file://), the images must be copied to the specified directory.

Once the file is copied to the PDA, we can start the application launching DataMovil and choose PetStore within the list of local applications.

9.2. SmokingHabit

The first step consists in changing the file where the application is defined. This file can be found in the directory /Samples/SmokingHabit of the distribution CD. The name of the file is **SmokingHabit_1_1**. We edit this DataMovil-XForms ASCII file and look for the attribute “action” of the “submission” tag. The URL included there must be changed to the URL route where we want the data to be sent.

We also have to modify the “src” attribute of the “media” tag whose id is “titulo”. The value to insert will be *file://* followed by the route where the image will be “SmokingHabitsTitle.jpg” (for example, */Windows/lib/SmokingHabitsTitle.jpg*). Doing this operation, the media block will be as shown below (with three slashes between file and Windows):

```
<mediablock>
  <media id="titulo" src="file:///Windows/lib/SmokingHabitsTitle.jpg"
    preload="true" mediatype="image/jpeg"/>
</mediablock>
```

The next step is to copy the SmokingHabit_1_1 (to the application directory defined in the data.ini file) and SmokingHabitsTitle.jpg to the PDA, following the same procedure as for the previous example.

Once we have made this change, we copy the file to the PDA. This file must be copied to the directory where the applications are stored. This directory is defined in the **data.ini** file of the of the DataMovil smart client in the field LOCAL_APPL_DIRECTORY relative to the LOCAL_USER_PATH field.

Once the file is copied to the PDA, we can start the application launching DataMovil and choose SmokingHabit within the list of local applications.