



Nokia Data Gathering

Server Developer
Guide (3.04)

Contents

Contents.....	2
Introduction.....	3
1. Projects	4
1.1 ndg-commons-core (utilities and commons)	4
1.2 ndg-server-core (server engine).....	4
1.3 ndg-server-servlets (server <-> mobile communication).....	4
1.4 ndg-web-server (server<->administrator panel communication).....	4
1.5 ndg-web-ui (administrator panel)	4
2. Setting Environment	5
2.1 Tools.....	5
2.2 Prepare MySQL database	5
2.3 Import projects into Eclipse	6
2.4 Adding ant script to Eclipse	7
2.5 Setting up JBoss Application Server	7
3. Building and Deploying	8
4. OpenRosa implementation	9
4.1 Survey builder.....	9
4.2 Functionalities	9

Introduction

Nokia Data Gathering is a solution that helps organizations *to collect field data using mobile phones* instead of paper forms, PDAs or laptops. Since mobile phones can send data from many remote locations, collected data can be transmitted in near real-time for analysis. This makes collection of data much *faster, more accurate and more cost effective* to gather especially in remote locations when dealing with critical issues, such as public health, agricultural stock levels, emergency services and alike.

The solution consists of *two modules, server and mobile phone*, to enable smooth information transfer from the survey administrators to the field workforce and vice versa. The process includes the creation of questionnaires, their delivery to mobile phones and the subsequent integration and analysis of results.

This mobile client developer guide gives information on the projects, server development environment and the implementation of open rosa.. More information about Nokia Data Gathering and further resources can be found from <https://projects.forum.nokia.com/ndg/wiki>.

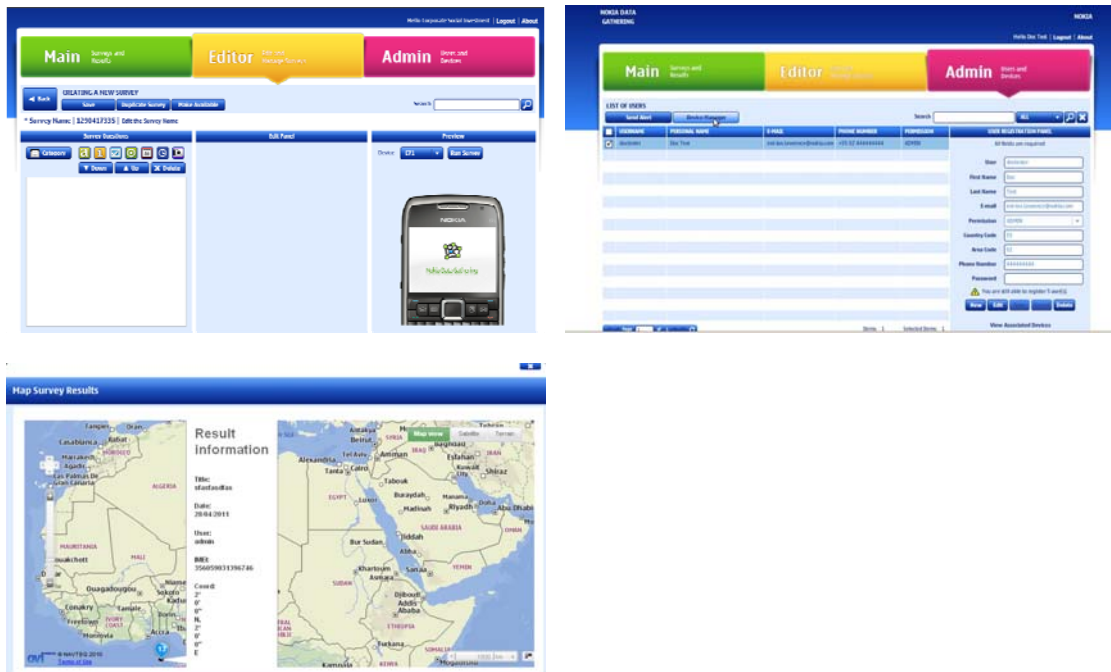


Figure 1 Nokia Data Gathering server

1. Projects

1.1 ndg-commons-core (utilities and commons)

This project holds classes which can be used across other projects. It is a good place to keep all kind of *utility classes*. Also *surveys and results parsing classes* are kept here.

1.2 ndg-server-core (server engine)

This is the server engine handling the *business logic and persistence* for surveys, results, devices, users, transaction history etc.

Data is stored in MySQL 5.x database. Persistence is achieved through the use of Enterprise JavaBeans 3 (EJB3) with Hibernate as a persistence provider. SMS, email, authorization and authentication logic is handled as well.

1.3 ndg-server-servlets (server <-> mobile communication)

This is the main point of *communication between the server engine and the mobile clients*.

Communication is achieved by means of servlets. Requests to update client application, register client, download surveys, upload results etc. are handled here. New surveys can be sent to server through the administrator panel in the case of Nokia Data Gathering surveys and directly by accessing the correct servlet in the case of OpenRosa surveys (see chapter 4).

1.4 ndg-web-server (server<->administrator panel communication)

This is the main point for *communication between the server engine and the administrator panel* (ndg-web-ui). It acts mostly as an interface between commands and actions from the administrator panel and the server engine logic. Actions like exporting survey results (.xls and .csv) are handled here as well.

1.5 ndg-web-ui (administrator panel)

This is the user interface for the *administrator panel* implemented using Flex. All the logic is provided by ndg-web-server.

2. Setting Environment

2.1 Tools

These are recommended tools with recommended versions. In most cases newer versions are available but they might need some changes in either the environment or the projects themselves in order to work.

- Eclipse (Helios Service Release 2)
- Ant
- MySQL (v5.x)
- Flex SDK (v3.5)
- Optional: Adobe Flash Builder 4 (commercial license required to extend trial version)
- JBoss Application Server (v4.2.2)
- Git

2.2 Prepare MySQL database

Warning: You might want to check the script if you use your database with other projects.

There is a script with SQL commands that needs to be executed to create the appropriate users, database and tables with example data.

The scripts are available at <http://brave.indt.org/NDG/>

To add basic NDG tables to database execute:

```
mysql -h localhost -u root -p < database_script.sql
```

Also adding a privileged user might be useful:

```
mysql -u root -p
mysql> grant usage on *.* to ndg@localhost identified by 'ndg';
mysql> grant all privileges on ndg.* to ndg@localhost;
```

To add OpenRosa support execute additionally:

```
mysql -h localhost -u root -p < database_script_openrosa.sql -f
```

NOTE: The two scripts have been merged into one to create the release packages for Windows, Red Hat and Debian/Ubuntu.

2.3 Import projects into Eclipse

1. Open Eclipse and switch to clean workspace

File -> Switch Workspace -> Other...

2. Import projects into workspace

File -> Import... -> General -> Existing Projects into Workspace

3. In 'Select root directory' browse to directory containing NDG server projects and OK.

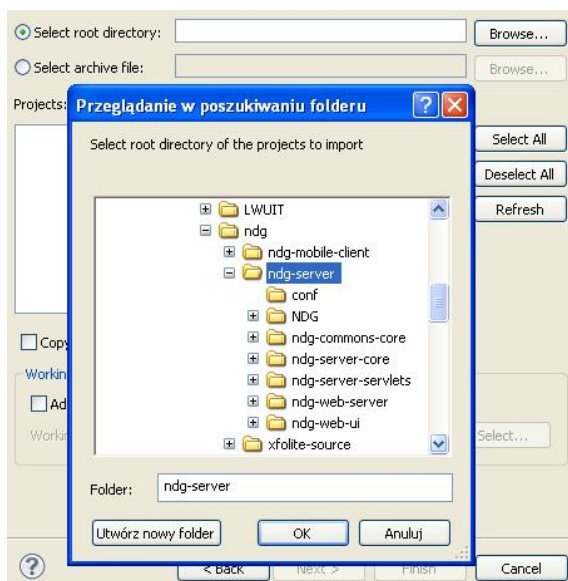


Figure 2 Selecting root directory

4. All five projects should appear checked on the list. Finish.

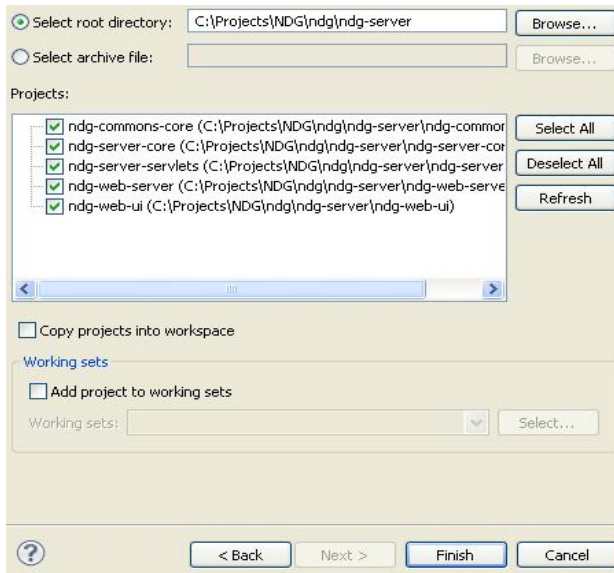


Figure 3 List of active projects

NOTE: Do not worry if Eclipse indicates any errors in the projects. It's (sort of) expected.

2.4 Adding ant script to Eclipse

The recommended way to build and deploy all projects is to use the prepared Ant script. We will now add it to Eclipse.

First get Ant View:

Window -> Show View -> Other...->type:Ant

In newly opened view:

Right-click and select 'Add buildfiles...'

Navigate to: *build_all.xml* in the root of the checkout from github <https://github.com/rodavelino/ndg/> (development takes place in the unstable branch).

2.5 Setting up JBoss Application Server

There are two ways to use the server. The server will require restarting quite often so choose your preference:

- Adding JBoss to Eclipse is simple. First You need to add Servers View:

Window -> Show View -> Other...->type:Servers

In opened view:

Right-click and New -> Server -> JBoss -> 4.2

'Next' and fill server address (or leave if you want to use it locally 127.0.0.1)

- Running from command line:

You can run JBoss by executing batch file located in:

```
$JBOSS_ROOT/bin/run.bat -b 0.0.0.0
```

-b x.x.x.x – argument specifies to what interface will JBoss work on, 0.0.0.0 stands for all interfaces

Read more about Nokia Data Gathering deployment is in the chapter 3.

3. Building and Deploying

The Ant script performs three steps. It

- Cleans up projects
- Build projects
- Deploys them to a server

The only step left is to set path to Flex SDK and JBoss deployment directories. You can do this either by running the script (double-click on earlier added 'Build all' script in Ant View) and following the error instructions or you can manually find the `jboss.dir` and `flex_SDK_home.dir` properties in `general.properties` file.

After setting the properties script it will build and deploy the files to JBoss.

Now simply run the JBoss server (as described in 'Setting up JBoss Application Server' section) and go to Nokia Data Gathering admin panel homepage:

```
http://server-address:server-port/ndgFlex/swf/main.html
```

(<http://127.0.0.1:8080/ndgFlex/swf/main.html> by default).

4. OpenRosa implementation

A new feature on the server side for the Nokia Data Gathering release 3.04 was adding support for the OpenRosa standard. OpenRosa is a standard based on XForms for data collection on mobile devices. The goal of this feature implementation was to have a server capable of supporting both the Nokia Data Gathering proprietary data collection protocol and OpenRosa.

After analyzing the similarities and differences of both standards it was decided to keep them separate (we only have some relations in the database and some common servlets).

4.1 Survey builder

<http://build.opendatakit.org/>

It was decided to use an external survey builder. Both the server and client side implementations were adjusted to process surveys created using the builder on OpenDataKit.org. This means that any survey created on the site is considered valid to be used and can be uploaded to Nokia Data Gathering server and then exported.

NOTE: Some OpenRosa question types (such as Barcode) are not supported on the client side. The client application must be ready to handle these unsupported questions.

4.2 Functionalities

Administrator panel

- Upload new surveys

/ndg-servlets/PostSurveys?do=uploadOpenRosa

Surveys are uploaded using the same servlet as the Nokia Data Gathering protocol. Surveys are NOT currently verified in terms of OpenRosa validity so they have to be valid before they are sent.

- Assign surveys to be download for a registered IMEI

/ndg-servlets/OpenRosaManagement?action=setSurveysForUser

IMEIs and available surveys are listed. Upon submission checked surveys are APPENDED to the currently selected IMEIs list of surveys available for download.

- Export results per IMEI

/ndg-servlets/OpenRosaManagement?action=exportResultsForUser

IMEI with results available for export are listed. On submission the selected IMEIs results are downloaded in a zipped archive.

Mobile client interface

- Upload results

/ndg-servlets/PostResultsOpenRosa

Result file should be uploaded using HTTP POST method using multipart encoding.

- Download surveys list

/ndg-servlets/ReceiveSurveys?do=list&deviceID=<IMEI>

<IMEI> has to be registered in Nokia Data Gathering database (the same as in the Nokia Data Gathering proprietary protocol). Returned xml document contains items with download URLs as specified by OpenRosa standard.

- Download survey

/ndg-servlets/ReceiveSurveys?do=download&deviceID=<IMEI>&formID=<formID>

URL in this format will be returned for every survey (identified by formID) by a survey-listing request.