Private Public Partnership Project (PPP)
Large-scale Integrated Project (IP)

D.6.3.1b: FI-WARE Installation and Administration Guide

Project acronym: FI-WARE
Project full title: Future Internet Core Platform
Contract No.: 285248
Strategic Objective: FI.ICT-2011.1.7 Technology foundation: Future Internet Core Platform
Project Document Number: ICT-2011-FI-285248-WP6-D.6.3.1b
Project Document Date: 2012-11-13
Deliverable Type and Security: Public
Author: FI-WARE Consortium
Contributors: FI-WARE Consortium
1.1 Executive Summary

This deliverable contains the Installation and Administration Guide for the Generic Enablers of Data/Context Management chapter, being included in the First FIWARE release. Concretely, installation and administration guides are provided for GEs implementing the following functionalities: BigData, Complex Event Processing, Context Broker for notifications management, video analysis in the compressed domain, location server and semantic and query tools.

This update/re-submission consolidates new contents and also contents in previous issues of Release 1. The reason for re-delivering parts that were already issued is twofold:

- FIWARE has made an effort to create a unified and improved format. The parts generated in the past are also provided in the new enhanced format for the sake of uniformity and readability.
- A single reference document per chapter is clearer and easier to handle that two incremental issues.
1.2 About This Document

The "FI-WARE Installation and Administration Guide" comes along with the software implementation of components, each release of the document referring to the corresponding software release (as per D.x.2), to facilitate the users/adopters in the installation (if any) and administration of components (including configuration, if any).

1.3 Intended Audience

The document targets system administrators as well as system operation teams of FI-WARE Generic Enablers from the FI-WARE project.

1.4 Chapter Context

FI-WARE will enable smarter, more customized/personalized and context-aware applications and services by the means of a set of assets able to gather, exchange, process and analyze massive data in a fast and efficient way. Nowadays, several well-known free Internet services are based on business models that exploit massive data provided by end users. This data is exploited in advertising or offered to 3rd parties so that they can build innovative applications. Twitter, Facebook, Amazon, Google and many others are examples of this. The "Data/Context Management" FI-WARE chapter aims at providing outperforming and platform-like GEs that ease development and provision of innovative Applications that require management, processing and exploitation of context information as well as data streams in real-time and at massive scale. Combined with enablers coming from the Applications/Services Ecosystem and Delivery chapters, application providers will be able to build innovative business models such as the ones described above and beyond.

FI-WARE Data/Context Management GEs enables to:

- Record, subscribe for being notified about and query for context information coming from different sources.
- Model changes in context as events that can be processed to detect complex situations that will lead to generation of actions or the generation of new context information (therefore, also treatable as events).
- Processing large amounts of context information in an aggregated way, using map&reduce techniques, in order to generate knowledge that may also lead to execution of actions and/or creation of new context information.
- Process data streams (particularly, multimedia video streams) coming from different sources in order to generate new data streams as well as context information that can be further exploited.
- Process metadata that may be linked to context information, using standard semantic support technologies.
- Manage some context information, such Location information, in a standardized way.

A cornerstone concept within this chapter is the structural definition of Data Elements enclosing its "Data Type", a number of "Data Element attributes" (which enclose the following: Name, Type, Value) and, optionally, a set of "Metadata Elements" (which have also in turn Data-like attributes: Name, Type, Value). However, this precise definition remains unbound to any specific type of representation and is able to represent "Context Elements"
and "Events" as "Data Element" structures. More comprehensive information is available at FI-WARE Data/Context Chapter vision.

"Data" in FI-WARE refers to information that is produced, generated, collected or observed that may be relevant for processing, carrying out further analysis and knowledge extraction. A cornerstone concept in FI-WARE is that data elements are not bound to a specific format representation.

The following diagram shows the main components (Generic Enablers) that comprise the first release of FI-WARE Data/Context chapter architecture.

More information about the Data Chapter and FI-WARE in general can be found within the following pages:

- [Data/Context Management Architecture](http://wiki.fi-ware.eu/)

### 1.5 Structure of this Document

The document is generated out of a set of documents provided in the public FI-WARE wiki. For the current version of the documents, please visit the public wiki at [http://wiki.fi-ware.eu/](http://wiki.fi-ware.eu/)

The following resources were used to generate this document:

- **D.6.3.1b FI-WARE Installation and Administration Guide front page**
- [BigData Analysis - Installation and Administration Guide](http://wiki.fi-ware.eu/)
- [CEP GE - IBM Proactive Technology Online Installation and Administration Guide](http://wiki.fi-ware.eu/)
1.6 Typographical Conventions

Starting with October 2012 the FI-WARE project improved the quality and streamlined the submission process for deliverables, generated out of the public and private FI-WARE wiki. The project is currently working on the migration of as many deliverables as possible towards the new system.

This document is rendered with semi-automatic scripts out of a MediaWiki system operated by the FI-WARE consortium.

1.6.1 Links within this document

The links within this document point towards the wiki where the content was rendered from. You can browse these links in order to find the "current" status of the particular content. Due to technical reasons not all pages that are part of this document can be linked document-local within the final document. For example, if an open specification references and "links" an API specification within the page text, you will find this link firstly pointing to the wiki, although the same content is usually integrated within the same submission as well.

1.6.2 Figures

Figures are mainly inserted within the wiki as the following one:

```
[[Image:.....|size|alignment|Caption]]
```

Only if the wiki-page uses this format, the related caption is applied on the printed document. As currently this format is not used consistently within the wiki, please understand that the rendered pages have different caption layouts and different caption formats in general. Due to technical reasons the caption can't be numbered automatically.

1.6.3 Sample software code

Sample API-calls may be inserted like the following one.

```
http://[SERVER_URL]?filter=name:Simth*&index=20&limit=10
```
1.7 Acknowledgements
The current document has been elaborated using a number of collaborative tools, with the participation of Working Package Leaders and Architects as well as those partners in their teams they have decided to involve.

1.8 Keyword list

1.9 Changes History

<table>
<thead>
<tr>
<th>Release</th>
<th>Major changes description</th>
<th>Date</th>
<th>Editor</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1</td>
<td>First Version</td>
<td>2012-11-01</td>
<td>TID</td>
</tr>
<tr>
<td>v2</td>
<td>Final Version</td>
<td>2012-11-13</td>
<td>TID</td>
</tr>
</tbody>
</table>

1.10 Table of Contents

1.1 Executive Summary ................................................................. 2
1.2 About This Document ........................................................................ 3
1.3 Intended Audience ........................................................................... 3
1.4 Chapter Context .................................................................................. 3
1.5 Structure of this Document .............................................................. 4
1.6 Typographical Conventions ............................................................... 5
1.6.1 Links within this document .......................................................... 5
1.6.2 Figures ........................................................................................... 5
1.6.3 Sample software code ..................................................................... 5
1.7 Acknowledgements .............................................................................. 6
1.8 Keyword list ....................................................................................... 6
1.9 Changes History .................................................................................. 6
1.10 Table of Contents .............................................................................. 6
2 BigData Analysis - Installation and Administration Guide .................... 12
2.1 Introduction ....................................................................................... 12
2.2 System Requirements ......................................................................... 12
2.2.1 Operating System Support ............................................................. 12
2.2.2 Resource requirements........................................................................................................12
2.2.3 Configuration options ........................................................................................................13
2.2.4 Platform home directory ....................................................................................................13
2.2.5 Platform work directory ....................................................................................................13
2.2.6 Platform logging directory ................................................................................................14
2.3 Installation software requirements ........................................................................................14
2.4 Package installation ................................................................................................................14
2.5 Configuration ..........................................................................................................................15
  2.5.1 Node Configuration ...........................................................................................................15
2.6 Sanity check procedures .........................................................................................................16
  2.6.1 End to End testing ...............................................................................................................16
  2.6.2 List of Running Processes .................................................................................................17
  2.6.3 Network interfaces Up & Open ........................................................................................18
  2.6.4 Databases ........................................................................................................................18
2.7 Diagnosis Procedures ..............................................................................................................18
  2.7.1 Resource availability .........................................................................................................18
  2.7.2 Remote Service Access .....................................................................................................18
  2.7.3 Resource consumption .....................................................................................................18
  2.7.4 I/O flows ..........................................................................................................................18
2.8 References ................................................................................................................................18
3 CEP GE - IBM Proactive Technology Online Installation and Administration Guide...........20
  3.1 CEP GE ..................................................................................................................................20
    3.1.1 Prerequisites ....................................................................................................................20
    3.1.2 Installation .......................................................................................................................20
    3.1.3 Configuration ...................................................................................................................20
    3.1.4 Running ............................................................................................................................21
  3.2 Sanity check procedures .........................................................................................................22
    3.2.1 End to End testing ............................................................................................................22
    3.2.2 List of Running Processes ...............................................................................................22
    3.2.3 Network interfaces Up & Open .......................................................................................22
    3.2.4 Databases ........................................................................................................................22
  3.3 Diagnosis Procedures .............................................................................................................22
    3.3.1 Resource availability .......................................................................................................22
    3.3.2 Remote Service Access ....................................................................................................23
    3.3.3 Resource consumption ....................................................................................................23
3.3.4 I/O flows ........................................................................................................ 23

4 Publish/Subscribe Context Broker - Context Awareness Platform - Installation and Administration Guide ........................................................................................................ 24
  4.1 Publish/Subscribe GE (Broker) Description .................................................. 24
  4.2 Software architecture ...................................................................................... 24
  4.3 Deployment architecture .................................................................................. 25
    4.3.1 Pub/Sub GE Database .............................................................................. 25
    4.3.2 Publish/Subscribe GE Software .............................................................. 26
    4.3.3 Pub/Sub GE Configuration ...................................................................... 30
  4.4 Diagnosis Procedures ...................................................................................... 32
    4.4.1 Resource availability ............................................................................... 32
    4.4.2 Remote Service Access .......................................................................... 33
    4.4.3 Resource consumption ........................................................................... 33
    4.4.4 I/O flows ................................................................................................. 33
  4.5 Sanity check procedures .................................................................................... 33
    4.5.1 End to End testing .................................................................................. 33
    4.5.2 List of Running Processes ....................................................................... 34
    4.5.3 Network interfaces Up & Open ............................................................... 34
    4.5.4 Databases ............................................................................................... 34

5 Publish/Subscribe Context Broker - SAMSON Broker - Installation and Administration Guide ........................................................................................................ 35
  5.1 Installation and Preparation ............................................................................ 35
    5.1.1 SAMSON Broker Installation .................................................................. 35
    5.1.2 MySQL installation and startup ............................................................... 35
    5.1.3 MySQL preparation ................................................................................ 35
  5.2 Starting pubSub Broker ................................................................................... 36
  5.3 Log file ............................................................................................................ 36
  5.4 Administration REST Interface ...................................................................... 37
    5.4.1 Modifying/Getting verbose level ............................................................. 37
    5.4.2 Modifying/Getting debug level .............................................................. 37
    5.4.3 Modifying/Getting reads level ............................................................... 37
    5.4.4 Modifying/Getting writes level .............................................................. 37
    5.4.5 Modifying/Getting trace levels ............................................................... 37
    5.4.6 Getting version of running executable ................................................... 38
  5.5 Sanity check procedures ................................................................................... 38
5.5.1 End to End testing.................................................................38
5.5.2 List of Running Processes....................................................38
5.5.3 Network interfaces Up & Open...........................................38
5.5.4 Databases.............................................................................38
5.6 Diagnosis Procedures...............................................................39
  5.6.1 Resource availability..........................................................39
  5.6.2 Remote Service Access.......................................................39
  5.6.3 Resource consumption.......................................................40
  5.6.4 I/O flows............................................................................40
6 Compressed Domain Video Analysis - Installation and Administration Guide..........41
  6.1 Prerequisites........................................................................41
    6.1.1 System........................................................................41
    6.1.2 Network........................................................................41
  6.2 Installation.............................................................................41
  6.3 Configuration.........................................................................41
  6.4 Sanity Check Procedures........................................................41
    6.4.1 End to End Testing............................................................41
    6.4.2 List of Running Processes................................................42
    6.4.3 Network Interfaces Up & Open.........................................42
    6.4.4 Databases........................................................................42
  6.5 Diagnosis Procedures...............................................................42
    6.5.1 Resource Availability.......................................................42
    6.5.2 Remote Service Access....................................................42
    6.5.3 Resource Consumption...................................................42
    6.5.4 I/O Flows........................................................................42
  6.6 References..............................................................................43
7 Location Server - LOCS - Installation and Administration Guide..........................44
  7.1 Introduction............................................................................44
    7.1.1 Change History ...............................................................44
  7.2 Installation and Preparation......................................................44
    7.2.1 Resource requirements.....................................................44
    7.2.2 Operating System and COTS Installation...........................45
    7.2.3 Location GE Software Installation....................................45
    7.2.4 End User Handset Simulator Installation............................47
  7.3 Starting Location GE.................................................................47
7.4 Stopping Location GE .................................................................48
7.5 Logging .................................................................................48
7.6 REST Interface configuration ..................................................48
7.7 MLP Interface configuration ......................................................49
7.8 Sanity check procedures ...........................................................49
  7.8.1 End to End testing ...............................................................49
  7.8.2 List of Running Processes ......................................................49
  7.8.3 Network interfaces Up & Open ...............................................49
  7.8.4 Databases ........................................................................49
7.9 Diagnosis Procedures ...............................................................50
  7.9.1 Resource availability ...........................................................50
  7.9.2 Remote Service Access .......................................................50
  7.9.3 Resource consumption ........................................................50
  7.9.4 I/O flows ........................................................................50
8 Query Broker - Installation and Administration Guide .......................51
  8.1 Introduction ...........................................................................51
  8.2 System Requirements .............................................................51
    8.2.1 Operating System Support ..................................................51
    8.2.2 Resource requirements ......................................................51
  8.3 Configuration ........................................................................51
  8.4 Sanity check procedures ........................................................54
    8.4.1 End to End testing ...............................................................54
    8.4.2 List of Running Processes ......................................................55
    8.4.3 Network interfaces Up & Open ...............................................55
    8.4.4 Databases ........................................................................55
  8.5 Diagnosis Procedures .............................................................55
    8.5.1 Resource availability ...........................................................55
    8.5.2 Remote Service Access .......................................................55
    8.5.3 Resource consumption ........................................................55
    8.5.4 I/O flows ........................................................................55
9 Semantic Application Support - Installation and Administration Guide ....56
  9.1 Introduction ...........................................................................56
  9.2 System Requirements .............................................................56
  9.3 Installation guidelines ..............................................................56
    9.3.1 Configuring the infrastructure .............................................56
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.3.2</td>
<td>Deploying the GE</td>
<td>60</td>
</tr>
<tr>
<td>9.4</td>
<td>Sanity check procedures</td>
<td>63</td>
</tr>
<tr>
<td>9.4.1</td>
<td>End to End testing</td>
<td>63</td>
</tr>
<tr>
<td>9.4.2</td>
<td>List of Running Processes</td>
<td>63</td>
</tr>
<tr>
<td>9.4.3</td>
<td>Network interfaces Up &amp; Open</td>
<td>64</td>
</tr>
<tr>
<td>9.4.4</td>
<td>Databases</td>
<td>64</td>
</tr>
<tr>
<td>9.5</td>
<td>Diagnosis Procedures</td>
<td>64</td>
</tr>
<tr>
<td>9.5.1</td>
<td>Resource availability</td>
<td>64</td>
</tr>
<tr>
<td>9.5.2</td>
<td>Remote Service Access</td>
<td>64</td>
</tr>
<tr>
<td>9.5.3</td>
<td>Resource consumption</td>
<td>64</td>
</tr>
<tr>
<td>9.5.4</td>
<td>I/O flows</td>
<td>64</td>
</tr>
<tr>
<td>10</td>
<td>Semantic Annotation - Administration and Installation Guide</td>
<td>65</td>
</tr>
<tr>
<td>10.1</td>
<td>Introduction</td>
<td>65</td>
</tr>
<tr>
<td>10.2</td>
<td>System Requirements</td>
<td>65</td>
</tr>
<tr>
<td>10.3</td>
<td>Installation guidelines</td>
<td>65</td>
</tr>
<tr>
<td>10.3.1</td>
<td>Configuring the infrastructure</td>
<td>65</td>
</tr>
<tr>
<td>10.4</td>
<td>Deploying the GE</td>
<td>66</td>
</tr>
<tr>
<td>10.4.1</td>
<td>Getting the software</td>
<td>66</td>
</tr>
<tr>
<td>10.4.2</td>
<td>Configuring the software</td>
<td>67</td>
</tr>
<tr>
<td>10.5</td>
<td>Sanity check procedures</td>
<td>68</td>
</tr>
<tr>
<td>10.5.1</td>
<td>End to End testing</td>
<td>68</td>
</tr>
<tr>
<td>10.5.2</td>
<td>List of Running Processes</td>
<td>68</td>
</tr>
<tr>
<td>10.5.3</td>
<td>Network interfaces Up &amp; Open</td>
<td>68</td>
</tr>
<tr>
<td>10.5.4</td>
<td>Databases</td>
<td>68</td>
</tr>
<tr>
<td>10.6</td>
<td>Diagnosis Procedures</td>
<td>69</td>
</tr>
<tr>
<td>10.6.1</td>
<td>Resource availability</td>
<td>69</td>
</tr>
<tr>
<td>10.6.2</td>
<td>Remote Service Access</td>
<td>69</td>
</tr>
<tr>
<td>10.6.3</td>
<td>Resource consumption</td>
<td>69</td>
</tr>
<tr>
<td>10.6.4</td>
<td>I/O flows</td>
<td>69</td>
</tr>
</tbody>
</table>
2     BigData Analysis - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

2.1     Introduction
This guide covers the steps needed to install and configure the SAMSON platform.

2.2     System Requirements
This section covers the base level requirements needed to install and use the SAMSON platform.

2.2.1     Operating System Support
The SAMSON platform has been built and tested against the following Linux distributions:
- Redhat Enterprise Server 6.0, 6.1 and 6.2
- CentOS 6.0, 6.1 and 6.2

2.2.2     Resource requirements
In order to operate effectively the SAMSON platform requires an adequately configured disk storage system in addition to physical RAM. Minimum and recommended resource specifications can be found below.

2.2.2.1     Minimum resource requirements
The following table contains the minimum resource requirements for running the platform on desktop based systems:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Requirement</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>2-4 cores</td>
<td></td>
</tr>
<tr>
<td>Physical RAM</td>
<td>2G-4GB</td>
<td></td>
</tr>
<tr>
<td>Disk Space</td>
<td>500GB(^2)</td>
<td>See (^3)</td>
</tr>
</tbody>
</table>

2.2.2.2     Recommended resource requirements
The following table contains the recommended resource requirements for running the platform in production. Note that the values below are for each node in the cluster:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Requirement</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>8-16 cores</td>
<td></td>
</tr>
<tr>
<td>Physical RAM</td>
<td>16-32GB</td>
<td></td>
</tr>
</tbody>
</table>
2.2.3 Configuration options
The SAMSON platform is made up from 4 components
- Base platform
- Development libraries for analytical module development
- Centralized log server

2.2.3.1 Base platform
The core of the SAMSON platform is contained within the base platform package. For a minimal install this is the only package that is required in order to operate a working SAMSON cluster of nodes.

2.2.3.2 Development libraries
In order to develop new analytical modules the development package needs to be installed. This package contains the necessary header files needed to bootstrap new module development as well as to make use of the data-types and operations provided by existing modules.

2.2.3.3 Centralized log server
The log server provides a centralized collection point for accessing and querying the log files generated by each node in the cluster. Whilst the platform can work without a log server, installing is recommended for reducing the administrative burden.

2.2.3.4 User account
By default the SAMSON platform is installed using the `samson` operating system account. If the account does not exist, the package installer creates it without any set password (to login to this account a password must be set using the `passwd` utility).

2.2.4 Platform home directory
The `/opt/samson` directory tree is used to store the platform applications as well as run time libraries for the analytical models. The location of the home directory is available via the environment variable `SAMSON_HOME`, configured in `/etc/profile.d/samson.sh`.

2.2.5 Platform work directory
The `/var/samson` directory is used to store the configuration file as well as the intermediate and output files resulting from processing. The location of the work directory is available via the environment variable `SAMSON_WORKING`, configured in `/etc/profile.d/samson.sh`. The amount of space allocated to the work area will depend on the amount of data that needs to be processed. The contents of the work folder are as follows:
- blocks – intermediate data blocks used for processing

Disk Space 8TB² See ³
• etc – node startup and cluster configuration
• log – current transaction data for operations

2.2.6 Platform logging directory
The SAMSON platform has two modes for logging activities within a given node; to the local file system or to a centralized log server. When the samsonWorker process is unable to connect to the log server the log files are saved in /var/log/samson. To read the log files generated by samsonWorker you must use the logCat command; see XXXXXXXX for information on using this utility.

2.2.6.1 Log server
The log server stores its log files in a different location to the SAMSON platform node. By default the files it generates are kept in /var/log/logserver. Unlike the files generated by samsonWorker, the log server files are not readable using standard Linux utilities such as grep, awk or sed. To read the log files you must use the logClient command; see the User Guide for information on using this utility.

2.3 Installation software requirements
To simplify the installation process we have provided pre-built packages for both platforms. The following section explains the pre-requisites for installing SAMSON:
In addition to the standard Redhat package repository, there is an external repository that needs to be installed before SAMSON can be installed. The Extra Packages for Enterprise Linux (EPEL) repository is needed for the correct version of Google’s protobuf software, not found in the standard RedHat repositories. To install this repository run the following command:

```
wget http://download.fedoraproject.org/pub/epel/6/i386/epel-release-6-5.noarch.rpm
rpm --Uhv epel-release-6-5.noarch.rpm
```

2.4 Package installation
With the downloaded RPMs you can install using the following command:

```
sudo yum install samson-0.6.1-FIWARE_1.1.0_SNAPSHOT.x86_64.rpm \
    samson-devel-0.6.1-FIWARE_1.1.0_SNAPSHOT.x86_64.rpm \
    samson-logserver-0.6.1-FIWARE_1.1.0_SNAPSHOT.x86_64.rpm \
    mongo-2.0.6-FIWARE_1.1.0_SNAPSHOT.x86_64.rpm \
    mongo-server-2.0.6-FIWARE_1.1.0_SNAPSHOT.x86_64.rpm
```
In addition to laying down the files needed to run the SAMSON platform the packing install performs the following additional tasks:

- Creates a `samson` operating system user without setting a password
- Adds an environment script file to `/etc/profile.d` that adds `SAMSON_HOME` and `SAMSON_WORKING` to each shell environment
- Installs an init.d script that will start up the SAMSON platform, in particular the `samsonWorker` process when the server restarts

To start the SAMSON platform either reboot the server or execute

```
sudo /etc/init.d/samson start
```

## 2.5 Configuration

### 2.5.1 Node Configuration

When installing the platform software for the first time the post-install phase configures the SAMSON node based on the current system configuration (number of CPU cores and physical system memory).

#### 2.5.1.1 Buffer size

The `samsonWorker` process uses shared memory segments to share data between operations. By default the `samsonWorker` process creates a 256MB shared memory segment for each CPU core. The size of this segment can be controlled via the parameter `general.shared_memory_size_per_buffer` defined in `$SAMSON_WORKING/etc/setup.txt`. As a general rule of thumb the size of the segment can calculated based on the following formula:

\[ \text{Available Memory} \div \text{number of cores} \times 20\% \]

#### 2.5.1.2 Shared memory sizing

In order for the SAMSON node to function correctly a large enough shared memory segment must be allocated. The size of this shared memory segment is based on the following formula:

\[ \text{No of cores} \times \text{general.shared_memory_size_per_buffer} \]

Where `general.shared_memory_size_per_buffer` is defined in `$SAMSON_WORKING/etc/setup.txt`. The following command will calculate the total amount of shared memory needed, in bytes, based on the current configuration:

```
expr $(grep ^processor /proc/cpuinfo| wc -l) "*" \n$(grep shared_memory_size_per_buffer $SAMSON_WORKING/etc/setup.txt | \ awk '{print $2}')
```
In a 4 CPU/core system with `general.shared_memory_size_per_buffer` configured to 256MB we get a value of 1073741824 (1GB)

There are two kernel parameters that affect the amount of shared memory that can be allocated:

- `kernel.shmall` - number of pages that can be allocated
  - When multiplied by the page size gives us the total amount of shared memory that can be allocated
- `kernel.shmmmax` – maximum size of a shared memory buffer

When `samsonWorker` starts it performs a check to see that `kernel.shmall` and `kernel.shmmmax` are correctly configured. If not, a message is presented informing the user of the values needed and how they can be changed.

### 2.5.1.3 Ports

The SAMSON platform uses the following ports for communication:

<table>
<thead>
<tr>
<th>Port</th>
<th>Protocol</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>1324</td>
<td>TCP</td>
<td>Cluster communications</td>
</tr>
<tr>
<td>1202</td>
<td>TCP</td>
<td>RESTful API</td>
</tr>
<tr>
<td>6000</td>
<td>TCP</td>
<td>Log Server queries from logClient</td>
</tr>
<tr>
<td>6001</td>
<td>TCP</td>
<td>Log Server communications</td>
</tr>
</tbody>
</table>

It should be noted that whilst these are the expected ports that SAMSON will use, it is possible to configure the platform to make use of different ports. In order to provide a more secure installation the firewall should be used on each node to allow traffic from known nodes. Refer to the operating system guide for information on configuring `iptables` (Redhat/CentOS) or `ufw` (Ubuntu).

### 2.5.1.4 SELinux / AppArmor

At the current time neither SELinux nor AppArmor supported and should be disabled.

### 2.6 Sanity check procedures

This section provides the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

#### 2.6.1 End to End testing

The following are to be executed from the command line as the `samson` user:

Check to see the cluster is up and running

```
  delilah -command "cluster info"
```
Add another node to the cluster, changing `nodename` to the name of the additional node. **Note** this command will only work with in a multi-node SAMSON cluster and should not be run for single node instances.

```bash
delilah -command "cluster add nodename"
```

Get a list of the active modules

```bash
delilah -command "ls_modules -group name"
```

### 2.6.2 List of Running Processes

Depending on the configuration of the cluster there can be one or two processes running:

- `samsonWorker` - SAMSON platform node process
- `logServer` - Centralized log server

**N.B.** only one `logServer` needs to be running within the cluster

Sample output from `ps`:

```
samson   13974     1  0 12:46 ?        00:00:00 samsonWorker
samson   13994     1  0 12:49 ?        00:00:00 logServer
```

In addition to processes, the `samsonWorker` process creates a number (depending on the `setup.txt` configuration) of shared memory segments. To view these use the following command.

```bash
ipcs
```

**e.g.**

```
$ ipcs -m

-------- Shared Memory Segments --------

key status shmid owner perms bytes nattch

0x00000000 61865992 samson 600 268435456 1
0x00000000 61898761 samson 600 268435456 1
0x00000000 61931530 samson 600 268435456 1
0x00000000 61964299 samson 600 268435456 1
0x00000000 61997068 samson 600 268435456 1
0x00000000 62029837 samson 600 268435456 1
0x00000000 62062606 samson 600 268435456 1
0x00000000 62095375 samson 600 268435456 1
0x00000000 62128144 samson 600 268435456 1
0x00000000 62160913 samson 600 268435456 1
0x00000000 62193682 samson 600 268435456 1
0x00000000 62226451 samson 600 268435456 1
```
2.6.3  Network interfaces Up & Open

See the Ports sub-section on the ports the platform uses.

2.6.4  Databases

By default SAMSON does not require a database to function. If you are processing data that is stored in an external database refer to the documentation for that product. To verify that MongoDB is working, see the MongoDB Manual.

2.7  Diagnosis Procedures

2.7.1  Resource availability

See resource requirements for information regarding the required and recommended configurations for this GE.

2.7.2  Remote Service Access

N/A

2.7.3  Resource consumption

The resources used by the platform depend on the amount of data being processed and the complexity of the algorithms being used to process the data. In order to monitor acceptable usage of the platform tools like Nagios and Ganglia can be used.

2.7.4  I/O flows

N/A

2.8  References

- 1 - For example processing 1 million 10KB entities, would require approximately 10GB of disk storage. More if the data is to be persisted for later queries.
- 2 - The actual disk space will depend on the type of problem being solved by the platform, the amount of data being processed and whether that data will need to be stored within the platform or in an external system such as MongoDB or Apache HDFS.
- Platform installation uses up approximately 40MB. The additional space requirement is for data processing.
3 CEP GE - IBM Proactive Technology Online
Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

3.1 CEP GE

The CEP GE is implemented by the IBM Proactive Technology Online (Proton) component. The Proton has a run time engine component that detects patterns above input events at run time (Proton engine) and an authoring tool user interface that allow the user to write CEP applications (Proton authoring tool).

3.1.1 Prerequisites

The Proton engine is a J2SE (Java 2 Platform, Standard Edition) application. It requires Java SE 6 or later installed. The Proton authoring tool is a web based user interface. It runs on a Apache Tomcat 7 and required its installation. It also requires Java SE 6 or later installed.

3.1.2 Installation

3.1.2.1 Installing the IBM Proactive technology online (Proton) engine

To install the Proton engine simple unzip the proton zip file (proton-<version-number>.zip) to a local folder.

3.1.2.2 Installing the IBM Proactive technology online (Proton) authoring tool

To install the Proton authoring tool, remove the version number from the two authoring tool war files (AuthoringTool-<version-number>.war, AuthoringToolWebServer-<version-number>.war) and deploy the two war files (AuthoringTool.war, AuthoringToolWebServer.war) to the Apache Tomcat server. Instruction on how to deploy application to an Apache Tomcat server can be found at Tomcat Web Application Deployment.

3.1.3 Configuration

Proton engine configuration consists of the following files:

- Proton.properties – the runtime looks for this file under ./config directory. Alternatively, an absolute path to the file can be specified when invoking Proton runtime as an argument. The file contains the following properties:
  - metadataFileName – the name of the JSON metadata file containing Proton EPN definitions. This file is created either by using the authoring tool web user interface or programmatically. In case the authoring tool is used to create the application definition, you should activate the export option in the authoring tool and set the metadataFileName property to the exported file name.
Future Internet Core Platform

- inputPortNumber – the port number for SocketServer for input adapters (this is the port number through which input adapters will communicate with Proton runtime). Can be any free port in the system.
- outputPortNumber – the port number for SocketServer for output adapters (this is the port number through which output adapters communicate with Proton runtime). Can be any free port in the system.
- logging.properties – can be found under ./config directory. This is the properties file for Java logging API. The most relevant entries within the file is the properties regarding ConsoleHandler and FileHandler. ConsoleHandler properties define how console logging is handled, specifically java.util.logging.ConsoleHandler.level property defines the logging level of messages which are displayed on the console, the default value is INFO. The file handler properties define where the log file is stored, what is the logging level to the file etc. Those properties can be used to control the logging of Proton runtime. In order to let Proton use this file instead of default logging properties, run the Proton jar with the following VM argument:
  
  java -Djava.util.logging.config.file=./config/logging.properties -jar ProtonApp.jar
  
  You can specify either a relative or an absolute path to the logging.properties file.

Instructions on how to define the input and output adapters to get input events and send output events are given in the programmer guide. The sample application that is provided with the installation is using file input and output adapters that are defined in the sample application definition file (no need for further adapter configuration).

No configuration is required for the Proton authoring tool.

3.1.4 Running

3.1.4.1 Running the Proactive Technology Online Engine
To run the Proton engine jar, open a command prompt in the folder you extracted the Proton zip file, and use the following cmd command: java -Djava.util.logging.config.file=./config/logging.properties -jar ProtonApp.jar

or alternatively if you want to specify a different Proton.properties file: java -Djava.util.logging.config.file=./config/logging.properties -jar ProtonApp.jar
c:\temp\proton\config\myProton.properties

3.1.4.2 Running the authoring tool
To run the authoring tool you need to have the Apache Tomcat server running and the two authoring tool war files (AuthoringTool-<version-number>.war, AuthoringToolWebServer-<version-number>.war) deployed and running. To access the authoring tool to edit application definition open a browser and point to the following link by setting the server IP address http://server-ip:port-number/AuthoringTool/Main.html The application was tested on Google Chrome.
3.2  Sanity check procedures

3.2.1  End to End testing
To verify that the Proactive Technology Online engine is running, open a command prompt in the folder you extracted the Proton zip file and run the launchProton.bat batch file (on Windows) or launchProton.sh (on Linux). You should see that events are sent to the engine and it processes them without any exception.

To verify that the Proactive Technology Online authoring tool is running, open a browser and open the following line (after setting the server IP and the Apache Tomcat port number)
http://server-ip:port-number/AuthoringTool/Main.html The authoring tool was tested on the Google Chrome and the Mozilla Firefox browsers.
The user interface should be opened with the buttons "open project" and "new". You should be able to click on the "open project" button and see the example project.

3.2.2  List of Running Processes
When running the Proactive Technology Online authoring tool, the Apache Tomcat server is running and you can see its Tomcat7.exe process. When running the Proactive Technology Online engine, the J2SE is running and you can see its java.exe process.

3.2.3  Network interfaces Up & Open
The Proactive Technology Online engine uses two ports. The ports can be configurable. However the default ports are 3000, 3300 The Proactive Technology Online authoring tool uses the Apache Tomcat port (the default HTTP port is 8080).

3.2.4  Databases
No Database is used.

3.3  Diagnosis Procedures
When an error occurs, the trace log of the Proactive Technology Online engine can be checked. The log file is stored in the location specified in the logging.properties file that can be found under the ./config directory. The logging level can be changed using this file. See more details in the configuration section above.

3.3.1  Resource availability
The required RAM is dependant on the application defined rules and the state size the application needs to hold at run time. Usually the hard disk required during run time is neglected, unless the application is uses adapters of type "File" and the input file or the generated output file are very big.
3.3.2 Remote Service Access
In the current first release, the CEP GE has no built-in integration with other GEIs.

3.3.3 Resource consumption
The resource consumption is highly depend on the actual CEP application defined and the actual event streams being processed. There are no typical numbers.

3.3.4 I/O flows
N/A
4 Publish/Subscribe Context Broker - Context Awareness Platform - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

4.1 Publish/Subscribe GE (Broker) Description

The Pub/Sub GE includes the software components installed in dedicated virtual server machines consisting of two servers:

- Pub/Sub GE components as a front-end;
- Pub/Sub databases as a back-end;

The Pub/Sub GE exposes REST http-based interfaces to publish and to retrieve context information, by means of ContextML or NGSI representation model, in both synchronous or asynchronous mode (i.e. subscription/notification).

Once the service is run it is always on and potentially available for any application or service requesting the context, so for both 3rd parties developers and end-users of the applications or services.

Context Provider could be dynamically registered or unregistered from the Pub/Sub GE using the registering announcement interfaces available. This allows decoupling Context Providers' interfaces from the Pub/Sub GE interfaces exposed towards Context Consumers, which are always available in order to serve any context-aware application or customer using the context.

4.2 Software architecture

The following Figure 2-3 shows the software architecture of the Publish/Subscribe GE.

![Figure 2-1: PSGE's SW Architecture](image)

D.6.3.1b FI-WARE Installation and Administration Guide
4.3 Deployment architecture

The following Figure 3-1 shows the high-level deployment architecture of the entire Publish/Subscribe GE including its context handling components and the databases.

![Deployment Architecture Diagram]

Figure 3-1: Pub/Sub GE Deployment Architecture

Detailed description of the SW components and their operations are described in the sections below.

4.3.1 Pub/Sub GE Database

4.3.1.1 Location of the component

Please see the installation procedure in the below section for the location.

4.3.1.2 System Requirements

- MySQL Server v5.0 DBMS with UTF-8 and InnoDB support

4.3.1.3 Installation

1. Create a new user "causer" on MySQL DBMS
2. Set permissions for creating databases to user “causer”
3. "causer" should have read/write access to data on databases created here below, both locally and remotely
4. Execute scripts from mysql command line. Scripts include commands to create databases
Future Internet Core Platform

4.3.1.4 Configuration
Publish/Subscribe GE requires some databases to be installed on the database machine. After databases have been installed (see section 3.1.3.) the following database configurations must be completed:

1. Copy "mysql-CAP-ds.xml" file into "%JBOSS_HOME%\server\ca_cap\deploy" folder.
2. In the file "mysql-CAP-ds.xml" check DB credentials and IP/ports, modifying the following parameters in each defined datasource, for example:

```xml
<connection-url>jdbc:mysql://DATABASE_HOST:3306/db_name</connection-url>
<user-name>username</user-name>
<password>password</password>
```

where DATABASE_HOST is the IP address of the database server, username and password are the access credentials and db_name is the database name for each created database.

4.3.2 Publish/Subscribe GE Software

4.3.2.1 Location of the component
The Publish/Subscribe GE components are placed in the following location:
C:\jboss-5.1.0.GA\server\ca_cap\deploy and available at the following URIs:
- [http://pubsub.lab.fi-ware.eu](http://pubsub.lab.fi-ware.eu) for the Publish/Subscribe GE
- [http://pubsub.lab.fi-ware.eu/AUP](http://pubsub.lab.fi-ware.eu/AUP) for the Advanced User Profile (AUP)

And in order to check functionality it’s possible to use the following method:

4.3.2.2 Requirements
Hardware requirements
- Tested configuration:
  - 1 server with 2 CPU;

```bash
> mysql -u causer -p --default-character-set utf8 -e "source filename.sql"
```
a. context_broker.sql (CB database)
b. aup.sql (AUP database)
c. create_xcap_table.sql
d. insert_caps.sql
e. populate_schema.sql
f. populate_auids.sql
g. lp.sql (LP database)
h. other databases: oauth, geoapi, ecc....
- 2 GB RAM;
- minimum 50 GB free hard disk space on used partition

Software requirements
- Operating System
- CentOS 32-bit
- Software components (recommended versions):
  - Java SE Development Kit (JDK) v1.6.0 available from http://www.oracle.com/technetwork/java/javase/downloads/
  - JBoss Application Server v5.1.0GA available from http://www.jboss.org/jbossas/downloads/

Connectivity requirements
- The server where the Pub/Sub GE will be installed requires connectivity with:
  - MySQL Server v5.0 DBMS for database connections
  - back-end systems (external providers)
  - All TCP ports required by JBoss default configuration (80, etc.) must be available

The Pub/Sub GE by default will provide access to the offered APIs on TCP port 80.

4.3.2.3 JDK and JBoss installation

NOTE All the installation procedure should be performed as "root" user

CENTOS JDK installation
1. Download file jdk-6u29-linux-i586-rpm.bin from Oracle site (or newer jdk-6u…-rpm.bin files) in "/root" folder and operate in that folder.
2. Set permissions:

```bash
> chmod a+x jdk-6u29-linux-i586-rpm.bin
```
3. Execute the file with command:

```bash
> ./jdk-6u29-linux-i586-rpm.bin
```
4. JDK should be installed in /usr/java/ folder.
5. Test installation with:

```bash
> java -version
```

UBUNTU JDK installation
1. Check file /etc/apt/sources.list, if necessary uncomment lines:
   o deb http://archive.canonical.com/ubuntu maverick partner
   o deb-src http://archive.canonical.com/ubuntu maverick partner
and launch the following command to update the repository configuration:

```bash
> apt-get update
```
2. Execute the command:

```bash
> apt-get install sun-java6-jdk
```
3. JDK should be installed in /usr/lib/jvm/java-6-sun folder.
4. Test installation with:

```bash
> java -version
```
JBoss installation and first run

1. Download JBoss (jboss-5.1.0.GA-jdk6.zip, from http://sourceforge.net/projects/jboss/files/JBoss/JBoss-5.1.0.GA/jboss-5.1.0.GA-jdk6.zip/download) and decompress it in a chosen installation folder (e.g. "/usr/local/jboss-5.1.0"):

```
> cd /tmp
> cd /usr/local
> unzip /tmp/jboss-5.1.0.GA-jdk6.zip
> mv ./jboss-5.1.0.GA-jdk6 ./jboss-5.1.0
```

In the following `%JBOSS_HOME%` will refer to the chosen JBoss installation directory.

2. Copy the following file from the installation package (folder JBoss_bin) in `%JBOSS_HOME%`/bin:
   a. log4j.properties

3. Make a copy of "%JBOSS_HOME%\server\default" and name it as "%JBOSS_HOME%\server\ca_cap":

```
> cp –R %JBOSS_HOME%/server/default %JBOSS_HOME%/server/ca_cap
```

4. Copy the file "jboss-log4j.xml" from the installation package (folder JBoss_instance_conf ) into "%JBOSS_HOME%\server\ca_cap\conf"

5. Create a JBoss user, without password (Ubuntu requires you type some ENTER after every subsequent question):

```
> adduser jboss
```

6. Copy init redhat script renamed to /etc/init.d/jboss:

```
> cp %JBOSS_HOME%/bin/jboss_init_redhat.sh /etc/init.d/jboss
```

7. Set script permissions:

```
> chmod 755 /etc/init.d/jboss
> chown –R jboss /usr/local/jboss-5.1.0
> chown jboss /etc/init.d/jboss
```

8. Modify the script /etc/init.d/jboss:
   a. Add the following lines just after line "#!/bin/sh" (verify that java path JAVAPTH and JBOSS_HOME folder match the correct ones):

```
# chkconfig: 2345 95 20
# description: Starts and stops JBossAS
# processname: jboss
JBOSS_HOME="/usr/local/jboss-5.1.0"
JAVAPTH="/usr/java/default"
```
9. Verify /etc/hosts file, it should contain the server name in the “127.0.0.1” line, otherwise there could be some UnknownHostException during JBoss startup.

It is recommended to change memory settings, substituting the following line of run.conf file in `%JBOSS_HOME%/bin` folder:

```java
JAVA_OPTS="-Xms128m -Xmx512m -XX:MaxPermSize=256m -Dorg.jboss.resolver.warning=true -Dsun.rmi.dgc.client.gcInterval=3600000 -Dsun.rmi.dgc.server.gcInterval=3600000"
```

with this one:

```java
JAVA_OPTS="-Xrs -Xmx512m -Xms512m -XX:NewSize=64m -XX:MaxNewSize=64m -XX:SurvivorRatio=6 -XX:PermSize=256m -XX:MaxPermSize=256m -Xdebug -Dorg.jboss.resolver.warning=true -Dsun.rmi.dgc.client.gcInterval=3600000 -Dsun.rmi.dgc.server.gcInterval=3600000"
```

Follow these steps to launch JBoss for the first time from the command line. Check for errors or problems (e.g. port conflicts, etc.), enter command:

```
> /etc/init.d/jboss start
```

From a web browser, verify the JBoss application server opening the following URL:

http://pubsub.lab.fiiware.eu:8080/web-console/

If the page is not reachable from another machine, verify firewall settings (use commands “setup” to disable/configure it, and “iptables -L” to verify).

To stop JBoss, use command:

```
> /etc/init.d/jboss stop
```

To run JBoss at the OS startup/reboot, under CentOS, use commands:

```
> cd /etc/init.d
> chkconfig --add jboss
```

(or use “ntsysv” utility)

Under Ubuntu, use instead the following commands:

```
> cd /etc/init.d
> update-rc.d jboss defaults
```

At the end of this process, JBoss should be started as a system service.

1. Copy the following files, provided with the Pub/Sub distribution, into the folder "%JBOSS_HOME%/server/ca_cap/lib"
   - activation-1.1.jar
   - avalon-framework.jar
   - ca-commons-1.0.6.jar
   - ca-commons-platform-1.0.1.jar
   - commons-beanutils-1.8.0.jar
   - commons-codec.jar
   - commons-dbcp-1.3.jar
- commons-digester-1.8.jar
- commons-discovery.jar
- commons-httpclient.jar
- commons-jxpath-1.3.jar
- commons-pool-1.5.4.jar
- httpclient-4.0.jar
- httpcore-4.0.1.jar
- icu4j_3_4.jar
- iri.jar
- jackson-core-asl-1.9.10.jar
- jackson-jaxrs-1.9.10.jar
- jackson-mapper-asl-1.9.10.jar
- jackson-xml-1.9.10.jar
- javassist-3.6.0.GA.jar
- jaxrs-api-2.3.1.GA.jar
- jcip-annotations-1.0.jar
- jcip-annotations.jar
- jcl-over-slf4j-1.5.8.jar
- jdom-1.0.jar
- mysql-connector-java-5.1.13.jar
- resteasy-jackson-provider-2.3.1.GA.jar
- resteasy-jAXB-provider-2.3.1.GA.jar
- resteasy-jaxrs-2.3.1.GA.jar
- resteasy-oauth-2.3.1.GA.jar
- scannotation-1.0.3.jar
- sjsxp-1.0.1.jar

- Components installation
  1. Copy the following files into "%JBOSS_HOME%\server\ca_cap\deploy " folder, where x.x.x is the version number of the component:
     - cb-x.x.x.ear
     - lp-x.x.x.ear
     - aup-x.x.x.ear
     - schemas.war
     - ngsi-cbwrapper-x.x.x.war
  2. Copy the file ca_commons_config.properties into "%JBOSS_HOME%\server\ca_cap\conf" folder
  3. Copy other “_config.properties file to the same folder, if present in Pub/Sub installation package.
  4. Copy "CB-jboss-destinations-service.xml" file into "%JBOSS_HOME%\server\ca_cap\deploy\messaging" folder

4.3.3 Pub/Sub GE Configuration
The configuration properties are in the file:
"%JBOSS_HOME%\server\ca_cap\conf\ca_commons_config.properties", provided in the installation package
4.3.3.1 **Logging Configuration**

Log files can be found in the folder: `%JBOSS_HOME%/server/ca_cap/log`. Log files are automatically rotated by the application server. Configuration files for logging are:

- "%JBOSS_HOME/bin/log4j.properties" for JBoss boot logging (when launched in manual mode)
- "%JBOSS_HOME%/server/ca_cap/conf/jboss-log4j.xml" for all other logging

To edit logging levels and appenders change the logging configuration in "%JBOSS_HOME%/server/ca_cap/conf/jboss-log4j.xml". For example, it is possible to modify the following parameters:

```xml
<appender name="SIZED_FILE_CB"
  class="org.jboss.logging.appender.RollingFileAppender">
  <errorHandler
class="org.jboss.logging.util.OnlyOnceErrorHandler"/>
    <param name="File" value="${jboss.server.log.dir}/cb.log"/>
    <param name="Append" value="true"/>
    <param name="Encoding" value="UTF-8"/>
    <param name="Threshold" value="INFO"/>
    <param name="MaxFileSize" value="10000KB"/> <!-- rotation file size -->
    <param name="MaxBackupIndex" value="10"/> <!-- rotation files -->
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern" value="%d{ISO8601} %-5p [%c] (%t:%x) %m%n"/>
    </layout>
  </appender>

[...]

<category name="CB" additivity="false">
  <level value="INFO" />
  <appender-ref ref="SIZED_FILE_CB" />
[...]
</category>
```

It is possible to select manually the threshold for the output of log messages within a single appender by editing the attribute value of the tag param Threshold. Accepted log levels are:

- DEBUG
4.3.3.2  JBoss scripts
The service.bat and run.conf.bat files provided in CA Platform installation package can be
derived from the original versions in JBoss bin folder, applying the following modifications:

- File service.bat
  Comment this row:
  ```
  set JAVA_OPTS=
  ```
  In "cmdStart" and "cmdRestart" functions, find the string
  ```
  call run.bat < .r.lock >> run.log 2>&1
  ```
  and modify it adding "-c all" and "-b 0.0.0.0":
  ```
  call run.bat -c all -b 0.0.0.0 < .r.lock >> run.log 2>&1
  ```
  In "cmdStop" and "cmdRestart" functions, find the string
  ```
  call shutdown < .s.lock >> shutdown.log 2>&1
  ```
  and modify it adding "--server=jnp://localhost:1099" and "--shutdown":
  ```
  call shutdown --server=jnp://localhost:1099 --shutdown < .s.lock >> shutdown.log 2>&1
  ```

- File run.conf.bat
  Modify the following row by adding "-Xrs" at the beginning of the string:
  ```
  set "JAVA_OPTS=-Xms128M -Xmx512M -XX:MaxPermSize=256M"
  ```

4.4  Diagnosis Procedures

4.4.1  Resource availability
A simple command "telnet localhost 8080" should test if JBoss is alive.
A host running the GE shall have at least:
  - 1CPU;
  - 2GB or RAM;
  - 100GB of hard-drive space.
If the free RAM is getting less then 1GB and there is less then 5GB of free disk space, the
GE instance begins to experience problems.
4.4.2 Remote Service Access
There are no remote connections established if no external entities like providers or consumers connected. If a provider or a consumer is connected TCP connections to the GE’s server will be established expecting the HTTP POST or GET communications. The connectivity pattern is of a JSON AS web service by HTTP protocol.

4.4.3 Resource consumption
It depends on the platform load. It can be assumed that a memory consumption of 500Mb to 1Gb is related to a normal working state. If the memory associated to JBoss java process is higher, it may be due to a malfunctioning.

4.4.4 I/O flows
The GE’s listening port shall be enabled for incoming connections in the local firewall as well as the outgoing connectivity to the GE’s server IP and listening port shall be enabled in the remote (consumer or provider, if remote) firewall.
The communication is a TCP sessions between the GE’s server and a remote entity (provider or consumer).
A GE’s server firewall shall enable the outgoing from the GE communication to the subscribed consumer’s IPs and it listening port in case of a subscription.

4.5 Sanity check procedures

4.5.1 End to End testing
Try to invoke this URL from a browser: http://pubsub.lab.fi-ware.eu/CB/ContextBroker/ping If everything is OK, the server should return a response similar to the following one:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<contextML xmlns="http://ContextML/1.7"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ContextML/1.7 http://contextml.tilab.com/ContextML-1.7.xsd">
<contextResp>
<contextProvider id="CB" v="1.4.6"/>
<timestamp>2012-07-26T15:33:38+02:00</timestamp>
<method>ping</method>
<resp status="OK" code="200"/>
<dataPart>
<par n="start">2012-07-18T10:41:48+02:00</par>
<parA n="methods">
</contextML>
```
If something is wrong, the response shows a description of the problem.

### 4.5.2 List of Running Processes
JBoss runs in a single JVM process (named "java ..."), which should be visible to a "ps -aux" command. It is important to verify that only ONE "java" process is active.

### 4.5.3 Network interfaces Up & Open
Pub/Sub GE services are exposed to standard port 80 (JBoss is active on port 8080, reverse proxied to the standard HTTP 80, which is the only port to be reachable from the internet).

### 4.5.4 Databases
The ping command checks also the availability of database connection. For a direct check, use command "telnet localhost 3306" on the database server, or "mysql" command line. Active MySQL databases (user/pwd: causer/cadb) could be checked with the following SQL queries:
- context_broker:
  - check with query: "select count(*) from atomicscope;"
- aup:
  - check with query: "select count(*) from user"
5 Publish/Subscribe Context Broker - SAMSON Broker - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

5.1 Installation and Preparation

5.1.1 SAMSON Broker Installation
The SAMSON Broker is a implementation of the Publish/Subscribe GE. It is distributed as a binary called pubSubBroker (available in the following .tgz for PPP members), that can be installed in the preferred place for the use. The suggested location is /usr/local/bin/pubSubBroker.

Note that pubSubBroker binary needs several libraries that has to be installed in the system. The libraries are provided by the following packages in CentOS 6 (information for other GNU/Linux distributions is not available, but there should be similar packages). Note that some packages are not available in "plain" CentOS 6.3 distribution and EPEL repositories are also needed (see http://vjetnamnet.com/how-to-configure-epel-repository-on-centos-6-3/).

- json-c
- cryptopp
- libxml2
- mysql-libs
- nss
- nss-utils
- libstdc++
- openssl

Internally, SAMSON Broker maintains its data on a MySQL database. The main advantages of this is that a huge amount of data can be supported (not limited on the amount of RAM in the system) and that the SAMSON Broker will not loose any of its data in case it needs to be restarted. Thus, in order for SAMSON Broker to work, MySQL must be installed and prepared.

5.1.2 MySQL installation and startup

```
% sudo yum install mysql-server.x86_64
% sudo service mysqld start
```

5.1.3 MySQL preparation

```
% mysql -u root
mysql> create database psb;
mysql> create user 'psb' identified by 'psb';
mysql> grant all on psb.* to 'psb'@'localhost' identified by 'psb';
```
As the MySQL commands suggests, the name of the database is 'psb', the username is 'psb' and its password is also 'psb'. These are also the default values for the corresponding command line options of pubSubBroker binary. It is highly recommended to alter these values when preparing the database and of course use the three command line options (-db, -dbuser and -dbpwd) to indicate the correct values when starting pubSubBroker binary.

5.2 Starting pubSubBroker

pubSubBroker supports a number of command line options, the most important being:

- -u (print the complete usage on screen)
- -port (port to receive new connections)
- -reset (reset database at startup)
- -dbhost (host where the database server runs)
- -dbuser (username to login to database)
- -dbpwd (password to login to database)
- -db (name of the database)

The first time pubSubBroker is started it will find an empty database and will then create all the necessary tables for it to function.

At any time, pubSubBroker can be killed (Unix standard: Ctrl-C) and restarted (without using the '-reset' option) and in the second run, the same data content will be available as in the first. If the option '-reset' is used when starting pubSubBroker, all database table will be emptied at startup.

5.3 Log file

pubSubBroker maintains a log file at /tmp/pubSubBrokerLog (the log directory can be overridden with: -logDir) and the level of debugging is determined by the following command line options:

- -t <trace level>
- -v (verbose mode)
- -vv (verbose2 mode)
- -vvv (verbose3 mode)
- -vvvv (verbose4 mode)
- -vvvvv (verbose5 mode)
- -d (debug mode)
- -r (reads mode)
- -w (writes mode)

The '-t' option is followed by a comma-separated list of ranges of trace levels, from 0 to 255. The significance of each of the trace levels is explained in [ User Guide ]. The trace levels are used for specific actions, while the verbose levels are used for more general debug output. If you just want to know what the program is doing, on a higher level, the verbose levels should be used, whereas if something fails, a specific trace level should be set. The five levels of verbose mode is pretty self explanatory - the more v:s, the more verbose output.

The '-d' option is deprecated and is not used. The '-r' and '-w' options are to be set if monitorization of incoming and outgoing packages is desired.
All these debug levels can be modified and viewed using an administration REST interface (see below in this same page).

5.4 Administration REST Interface

5.4.1 Modifying/Getting verbose level
- curl --request DELETE <host>:<port>/log/verbose
- curl --request GET <host>:<port>/log/verbose
- curl --request PUT <host>:<port>/log/verbose/off
- curl --request PUT <host>:<port>/log/verbose/[0-5]

5.4.2 Modifying/Getting debug level
Depracated but still implemented.
- curl --request DELETE <host>:<port>/log/debug
- curl --request GET <host>:<port>/log/debug
- curl --request PUT <host>:<port>/log/debug/on
- curl --request PUT <host>:<port>/log/debug/on

5.4.3 Modifying/Getting reads level
- curl --request DELETE <host>:<port>/log/reads
- curl --request GET <host>:<port>/log/reads
- curl --request PUT <host>:<port>/log/reads/on
- curl --request PUT <host>:<port>/log/reads/on

5.4.4 Modifying/Getting writes level
- curl --request DELETE <host>:<port>/log/writes
- curl --request GET <host>:<port>/log/writes
- curl --request PUT <host>:<port>/log/writes/on
- curl --request PUT <host>:<port>/log/writes/on

5.4.5 Modifying/Getting trace levels
- curl --request DELETE <host>:<port>/log/trace
- curl --request DELETE <host>:<port>/log/trace/t1
- curl --request DELETE <host>:<port>/log/trace/t1-t2
- curl --request DELETE <host>:<port>/log/trace/t1-t2,t3-t4
- curl --request GET <host>:<port>/log/trace
- curl --request PUT <host>:<port>/log/trace/t1
- curl --request PUT <host>:<port>/log/trace/t1-t2
- curl --request PUT <host>:<port>/log/trace/t1-t2,t3-t4
5.4.6  Getting version of running executable
- curl --request GET <host>:<port>/version

5.5  Sanity check procedures
The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.
As preliminary sanity test, check that the log file exists in the place is supposed to, typically /tmp/pubSubBrokerLog.

5.5.1  End to End testing
Login to any host from where the SAMSON Broker host is visible and issue the command:

```
% curl <pubSubBroker host>:1026/version
```

Expected Output:

```
% <ngsi><key>version</key><value>0.0.1</value></ngsi>
```

If you are not using default port (1026) curl command has to be adapted consequently.

5.5.2  List of Running Processes
- pubSubBroker. The PID file of this process is stored in /var/log/pubSubBroker.pid (this can be overridden with the -pidpath option)
- mysqld

5.5.3  Network interfaces Up & Open
- TCP:1026 (pubSubBroker default listenting port, configurable through -port)
- TCP:3306 (MySQL server)

5.5.4  Databases
SAMSON Broker is pretty configurable, but if the default values are used for pubSubBroker: MySQL(database='psb', user='psb', password='psb')
Simple test:

```
% mysql -u psb -p psb
password> psb
mysql> SHOW TABLES;
+-----------------------------+
| Tables_in_psb              |
+-----------------------------+
| attribute                  |
```

5.6 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

5.6.1 Resource availability

The SAMSON Broker needs the following resources to work: 2 CPU cores, 2GB RAM, 50 GB disk (to back the MySQL storage persistently).

In fact, this is a very conservative estimation (based on the FI-WARE testbed installation of this component) but until we can do a more precise profiling, it should be taken into account to install SAMSON Broker.

5.6.2 Remote Service Access

Context consumer and producer devices will connect directly to the SAMSON Broker. The input needed to connect is the host where the pubSubBroker binary is running and the port
where pubSubBroker listens for connections. The default port value is 1026, but that can be changed using the ‘-port’ command line option at starting pubSubBroker.

5.6.3 Resource consumption
The consumption of resources (CPU, RAM and MySQL disk usage) is proportional to the number of external connections (produces and consumer) the SAMSON Broker is supporting in a given moment, although particular figures are not available.

5.6.4 I/O flows
I/O flows will use TCP port 1026 by default (this can be overridden with -port parameter). However, the throughput cannot be estimated, this depends totally on the amount of external connections.
6  Compressed Domain Video Analysis - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

6.1  Prerequisites

6.1.1  System
Codoan requires Windows XP (or above) or Linux to be run. The system architecture has to be x86-compatible and must support multi-threading techniques. For running Codoan on desktop based systems at least a CPU with 4 cores and physical RAM of at least 2GB should be available (4GB preferred).

6.1.2  Network
Codoan provides a RESTful API and therefore requires port 80 to be open for input and output operations. The media is received as stream for which RTSP and RTP/RTCP is used. RTSP requires port 554 to be open and RTP/RTCP uses random ports >= 1024.

6.2  Installation
The RESTful web server and the core library of Codoan is provided as single executable file (Windows: codoan.exe, Linux: codoan). Therefore, no explicit installation is required.

6.3  Configuration
Each instance within Codoan can be configured separately. A single instance receives and processes a single RTP stream. The available configuration methods are configureEventDetection and configureObjectDetection (see [1]). They are accessible via the RESTful API. The standard procedure to set up an instance is as follows: createInstance -> configureEvent/ObjectDetection -> addSink(s)

6.4  Sanity Check Procedures
The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

6.4.1  End to End Testing
To check that Codoan is up and running, request a list of all running instances by sending the following HTTP request.

GET //server/mma/listInstances HTTP/1.1
6.4.2 List of Running Processes
All modules of Codoan (e.g., analyzer, RTP stack, web server, ...) are included in the executable file. Therefore, Codoan will run in a single process (codoan.exe/codoan).

6.4.3 Network Interfaces Up & Open
- 80/HTTP: RESTful API provided by Mongoose web server
- 554/RTSP: Establishing sessions for RTP reception
- 1024-65535/RTP-RTCP: Receiving RTP streams (random port pair for each session)

6.4.4 Databases
Codoan does not use any databases.

6.5 Diagnosis Procedures
The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

6.5.1 Resource Availability
Although 4GB RAM are recommended, the system should not have less than 2GB to work properly. That is because of some internal data structures that are stored temporarily during processing. The hard disk size is less critical since Codoan just writes some small log files to disk.

6.5.2 Remote Service Access
In the first release, Codoan has no built-in integration with other GEs.

6.5.3 Resource Consumption
The resource consumption depends on the number and the characteristics of processed streams. No typical numbers can be stated here.

6.5.4 I/O Flows
n/a
6.6 References

[1] Compressed Domain Video Analysis GE RESTful API Specification
7 Location Server - LOCS - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

7.1 Introduction

This guide covers the steps needed to install and configure the Location GE server platform. LOCS implements the FI-WARE GE Open Specifications associated to the Location GE available at: http://forge.fi-ware.eu/plugins/mediawiki/wiki/fiware/index.php/FIWARE.ArchitectureDescription.Data.Location

Whenever the term "Location GE" is used, you may assume that we are indeed referring to LOCS that implements the Location GE Open Specifications or an instance of LOCS. The Location GE is provided as tar gzip archives. First of all, Operating System and external COTS installation prerequisites is described. Then, installation/initial configuration of Location GE software is described. Protocol and localization functional configuration is defined in editable text properties files. Location system configuration data (network cell definition, privacy policy and end user access control, etc..) is maintained internally in a MySQL database).

7.1.1 Change History

History of changes is described in the following table.

<table>
<thead>
<tr>
<th>Date</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apr 18, 2012</td>
<td>• Initial Version</td>
</tr>
<tr>
<td>Sept 26, 2012</td>
<td>• Complete document for location GE version 2.1</td>
</tr>
<tr>
<td></td>
<td>(Add locationGE service installation, MLP interface and LRX agent deployment)</td>
</tr>
</tbody>
</table>

7.2 Installation and Preparation

This chapter describes the host server COTS prerequisites before installing the Location GE Software. In general, standard installation of these COTS is sufficient. Specific settings is indicated here for each COTS installation if needed.

7.2.1 Resource requirements

In order to operate effectively the Location GE platform requires the following minimum recommended resource specifications:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>2 cores</td>
</tr>
<tr>
<td>Physical RAM</td>
<td>2GB</td>
</tr>
</tbody>
</table>
7.2.2 Operating System and COTS Installation

7.2.2.1 Operating System
Linux operating system is required and must be installed before. Location GE can run using Ubuntu / RedHat. Ubuntu 10.04 or higher is recommended. RedHat 5.3 or higher is recommended. The `chkconfig` package is needed for installing locationGE as a service at server reboot. If not present on Ubuntu this package can be obtained with the command `apt-get install chkconfig`.

7.2.2.2 Java JRE
Standard Java JRE 6.0 is required. At least version 1.6.0_31 is recommended. On Ubuntu, Java JRE can be obtained running `apt-get openjdk-6-jre`.

7.2.2.3 MySQL Installation
MySql Community edition is required. Both client and server rpms must be installed. At least version 5.1.49 is recommended. On Ubuntu, MySQL can be obtained using `apt-get install mysql-server`.

7.2.2.4 MySQL Preparation and Specific Settings
The Location GE software assumes that the mysql root account is equal to `mysql`. If not done at MySql installation, change the password of the root account using the following command:

```
mysql -u root
mysql>set password for root@localhost = PASSWORD("mysql");
```

7.2.3 Location GE Software Installation
`/etc/hosts` file shall be completed if necessary with IP address of target server.
It is recommended to create a specific user account for administering the location GE (for example locationGE), under which installation/start/stop procedures are run.
The top installation directory is not enforced. The proposed one is `/opt/fiware/locationGE`.
User (admin) account running the Location Server GE must have read/write access to this directory contents (including sub-directories). Get the Location GE distribution archive (`Standalone-Fiware-LOCs-<version>.tar.gz`) and unzip the archive in the chosen installation directory.
7.2.3.1  **Environment settings**  
It is mandatory to define the `$LOCS_HOME` system variable with value equal to the top directory of the installation distribution layout. For example define in the `.bashrc` of the Location GE admin account.

```
export LOCS_HOME=/opt/fiware/locationGE
```

7.2.3.2  **Location GE service installation**  
Optionally, it is possible to install the location GE as as standard init.d service. This option allows automatic start-up of service on location GE server reboot.  
To perform such configuration, run the following command as `root` user :

```
cd $LOCS_HOME/install  
chmod +x *.sh  
./InstallLocsService.sh
```

7.2.3.3  **Deployment description**  
One unzipped, the Location GE software directory organization is the following :

```
$LOCS_HOME
   ├── cots
   │    ├── lib
   │    │    └── monconf
   │    └── scripts
   ├── install
   └── dtd
```

Overview of directories content is given here below :  
- **cots** : contains the java third party libraries used by the Location GE software  
- **lib** : contains the Location GE java software libraries  
- **monconf** : contains the Location GE standalone software configuration  
- **scripts** : contains the Location GE standalone management scripts (start, shutdown)  
- **install** : contains the Location GE standalone post-installation materials (database set-up)  
- **dtd** : contains the XML dtd schemas used in Location GE agent APIs.
Future Internet Core Platform

- **xsd**: contains the XML xsd schemas used in Location GE agent APIs.
- **log**: contains log file outputted at execution time
- **data**: contains Location GE test data
- **reports**: contains output reports generated by tests execution

### 7.2.3.4 Post-installation initialization procedure

**Database Initialization**:
Database schema contents and minimum insertion of test data shall be performed using the following command:

```
cd $LOCS_HOME/install
chmod +x *.sh
./Install.sh
```

At any time, this command can be used to reset the test database contents.

### 7.2.4 End User Handset Simulator Installation

An handset SUPL V2 simulator needs to be also installed in order to exercise SUPL scenarios workflows between Location GE Server and an end-user handset.

The top installation directory is not enforced. The proposed one is `/opt/fiware/handsetSimulator`. User (admin) account running the Location Server GE must have read/write access to this directory contents (including sub-directories). Get the simulator distribution archive (`TestTool-Fiware-<version>.tar.gz`) and unzip the archive in the chosen installation directory.

To start the handset simulator, use the following command:

```
cd /opt/fiware/handsetSimulator
./TestTool.sh
```

To stop the handset simulator, just interrupt the script with CTRL-C.

### 7.3 Starting Location GE

**Manual start procedure**

To be used if locationGE is not installed as a service.

Open a terminal on Location GE host with Location GE admin account, and run the `StartUp.sh` script in the `$LOCS_HOME/scripts` directory (for background execution run `StartUp.sh > $LOCS_HOME/log/StartUp.log &`).

**Start of locationGE service**

If locationGE is optionally installed as a service, the following procedure is valid also:

Open a terminal on Location GE host with Location GE admin account, and run the `service locsd start` command.

Location GE run-time application is composed of 6 java agent processes. Sub-version of each software components is outputted at start-up, then each started agent is listed.

For NET-Initiated scenario involving SMS sending from SUPL Agent to simulated handset, a SMS gateway simulator is also needed.
1. To manually start provided SMS gateway simulator, run the `SmscSimulator.sh` script in the `$LOCS_HOME/scripts` directory.

2. To include automatic start of SMS gateway simulator in locationGE start-up, set the following deployment settings (default settings):

   In `LightAgent_app.properties` configuration file in `$LOCS_HOME/monconf` directory, set the property `LightAgent.startSmscSimulator` to true.

### 7.4 Stopping Location GE

**Manual stop procedure**

There are two ways of stopping manually the LOCS:

- Type `CTRL-C` (interrupt signal) in the terminal in which the deployment manager is started (in interactive session)
- Run the script `Shutdown.sh` in the `$LOCS_HOME/scripts` directory (to be used in case of reconnection to server)

In interactive session, the shutdown procedure lists the agent/monitor currently stopping.

**Stops locationGE service**

If locationGE is optionally installed as a service, the following procedure is valid also:

Open a terminal on Location GE host with Location GE admin account, and run the `service locsd stop` command.

### 7.5 Logging

Each Location GE agent/monitor produces an individual log file in the `$LOCS_HOME/log` directory.

- `DBMonitor.log`: manage DB access and global agent/monitor monitoring.
- `MLPAgent.log`: contains front-end processing of Terminal Location REST request.
- `SS7Agent.log`: contains log of simulated execution of Network CELL-ID location requests.
- `LrxMonitor.log`: contains log of agent provisioning locationGE with live GPS data.
- `SUPLAgent.log`: contains log of SUPL exchanges with simulated end-user terminal.
- `SmscMonitor.log`: contains log of SMS gateway interface monitor.
- `TestSmsc.log`: contains log of simulated network SMS gateway.

### 7.6 REST Interface configuration

Terminal Location API REST Interface can be configured using the `$LOCS_HOME/monconf/MLPAgent_app.properties`.

HTTP connection port can be configured using the property `http_rest.port`. Default value at installation is 3128.
7.7 MLP Interface configuration

MLP Interface can be configured using the
$LOCS_HOME/monconf/MLPAgent_app.properties.
HTTP connection port can be configured using the property http.port. Default value at
installation is 3129.

7.8 Sanity check procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify
that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure
that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests
and user validation.

7.8.1 End to End testing

To verify access to the Terminal Location API, just issue the following GET request to the
Location GE Host using a web browser:

```
http://<location GE server ip>:3128/location/v1/queries/location
```

The location GE shall answer in the browser with a serviceException indicating a SYNTAX
ERROR.

7.8.2 List of Running Processes

- 7 java processes (one per agent + deployment launcher)
- mysqld
- optionally if launched, one java process for SMS Gateway simulator and handset
  simulator.

7.8.3 Network interfaces Up & Open

- HTTP:3128 (depending on Terminal Location API port defined)
- HTTP:3129 (depending on MLP API port defined)
- TCP:3306 (MySQL)

7.8.4 Databases

MySQL(database='CELLS', user='root', password='mysql') and MySQL(database='OPE',
user='root', password='mysql')

Simple test:

```
mysql -user root --password=mysql -h 127.0.0.1
mysql>use CELLS;
mysql>select * from Cell;
```
7.9 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

Starting point for diagnosis procedure shall check:

- contents of log files for "ERROR" messages (See Logging)
- running processes (See List of Running Processes)

7.9.1 Resource availability

See resource requirements for information regarding the required minimum configuration for this GE.

7.9.2 Remote Service Access

N/A

7.9.3 Resource consumption

The resources used by the platform depend on the amount of location request received per second par the Location GE server.

7.9.4 I/O flows

N/A
8 Query Broker - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

8.1 Introduction
This guide covers the steps needed to install and configure the QueryBroker platform.

8.2 System Requirements
This section covers the base level requirements needed to install and use the QueryBroker platform.

8.2.1 Operating System Support
The QueryBroker is a J2SE (Java 2 Platform, Standard Edition) application. It requires Java JRE 6 or later installed. The REST interface is realized as a wrapper and provided as WAR file (or Web application ARchive) together with the QueryBroker core. It is recommended to deploy it on a Apache Tomcat 7 (which requires its installation).

The QueryBroker platform has been built and tested against a Windows XP SP3 distribution. Other platforms running at least Java JRE 6.0 should work too.

8.2.2 Resource requirements
In order to operate effectively, the QueryBroker platform requires an adequately physical RAM, as the queries are processed in-memory. For running the platform on desktop based systems at least a CPU with 2-4 cores and physical RAM of about 2G-4GB should be available.

8.3 Configuration
In order to be able to register and access data repositories according "data base connectors", namely MPQF interpreters need to be implemented.

The following steps have to be conducted in order to integrate a new connector:

1. **Construct a MPQF service description:**

   In order to register a new retrieval service at the QueryBroker, a valid MPQF service description must be produced. In the following two example service description XML files are depicted, describing a LIRE (Lucene Image REtrieval) and the Picasa retrieval service.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<mpqf:MpegQuery mpqfID="001" xmlns:mpqf="urn:mpeg:mpqf:schema:2008"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:mpeg:mpqf:schema:2008
mpqf_semantic_enhancement.xsd">
  <mpqf:Management>
    <mpqf:Input>
```

```xml
```
A complete description of the used elements can be found in the MPQF standardization document (ISO_IEC_FDIS_15938--12_(E).pdf) along with the needed classification scheme. The ServiceID points to the Java class of the interpreter implementation (see next point). The generated file must be stored in the classpath of the QueryBroker-Server WAR-file for automatic registration.
2. Implement the Interface 'Service':
Besides the capability description of a service also a connector to it must be realized. This is
done by implementing the interface "Service" of the package "de.uop.dimis.air.backend"
accordingly. The system always sends a valid MPQF query to the retrieval service. If the
original query consists of more than one query type, this initial query will be split into several
sub-queries (one for each query type). This ensures, that every retrieval service only gets the
query type for processing, which is defined in the corresponding service description file.
Therefore, a retrieval service must translate the information encapsulated in the incoming
query into the underlying language/API calls. At the end of processing, the retrieved data
must be encapsulated into a valid MPQF response.
In the following an example source code of a QueryByDescription-Service for the service
"ExampleService" is given:

```java
// ExampleService.java
public class ExampleService implements Service {
    @Override
    public MpegType execute(MpegQueryType distributedQuery) {
        // get the query conditions
        BooleanExpressionType conditions = distributedQuery.getQuery().getInput().getQueryCondition();

        // ... do your program logic with the query conditions ...

        // create a result container for the computed results
        ObjectFactory mpqfObjFac = new ObjectFactory();
        MpegQueryType result = mpqfObjFac.createMpegQueryType();
        Query qry = mpqfObjFac.createMpegQueryTypeQuery();
        result.setQuery(qry);
        OutputQueryType oqt = mpqfObjFac.createOutputQueryType();
        qry.setOutput(oqt);
        List<ResultItemType> resultItems = oqt.getResultItem();

        // for each result of the service endpoint create a result
        // item and add it
        // to the results list
        de.uop.dimis.air.internalObjects.jpsearch.ObjectFactory
        jpsearchObjFac =
        new
de.uop.dimis.air.internalObjects.jpsearch.ObjectFactory();
        for(...) {
```
```java
ResultItemType resultItem = mpqfObjFac.createResultItemType();
resultItems.add(resultItem);
Description description = mpqfObjFac.createResultItemTypeDescription();
resultItem.getDescription().add(description);
JPSearchCoreType coreType = jpsearchObjFac.createJPSearchCoreType();
description.getContent().add(coreType);

// set the result properties in the jpsearch coreType object. (e.g.
// identifier)
coreType.setIdentifier("...");

// set the origin to identify from which service endpoint the
// result item was generated
resultItem.setOriginID("MedicoExecuteDICOM");

return result;
```

**Important:**

A serviceID is the full qualified class name (e.g., de.uop.dimis.test.Service) of the implemented service. The generated class must be stored in the classpath of the QueryBroker-Server WAR-file for automatic registration.

### 8.4 Sanity check procedures

Some basic tests to quickly evaluate whether the QueryBroker is up and running.

#### 8.4.1 End to End testing

To verify the access to the QueryBroker, just issue the following GET request to the QueryBroker Host using a web browser (checking which data repositories are available/already registered):

```
http://{serverRoot}/QueryBrokerServer/services/{capability}
```

where `{capability}` ∈ {QueryByDescription, QueryByFeatureRange, QueryByFreeText, QueryByMedia, QueryByRelevanceFeedback, QueryByXQuery, SpatialQuery, TemporalQuery, QueryBySPARQL}
The service invoked shall answer in the browser either "No valid capability set. Valid capabilities are: QueryByDescription, QueryByFeatureRange, QueryByFreeText, QueryByMedia, QueryByRelevanceFeedback, QueryByXQuery, SpatialQuery, TemporalQuery, QueryBySPARQL" or the corresponding service(s).

8.4.2 List of Running Processes
- 1 Application Server / Servlet-Container (e.g. Tomcat) instance, plus the QueryBroker processes started in Tomcat

8.4.3 Network interfaces Up & Open
- TCP:8080 (if Tomcat is employed)
- further depend on the connection types to the data repositories to be registered at the QueryBroker

8.4.4 Databases
- The GE itself has no database. However, it connects to external data repositories to be registered by the user in order to perform the queries.

8.5 Diagnosis Procedures

8.5.1 Resource availability
See resource requirements for information regarding the required and recommended configurations for this GE.

8.5.2 Remote Service Access
n/a

8.5.3 Resource consumption
The resources used by the platform depend on the amount of data being processed, especially on the size of the returned results as these are processed in memory.

8.5.4 I/O flows
n/a
9 Semantic Application Support - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

9.1 Introduction

This guide covers the steps needed to install and configure the Semantic Application Support GE.

9.2 System Requirements

In order to deploy the Semantic Application Support GE the following software must be previously installed:

- Java™ Platform, Standard Edition Development Kit (JDK™) 6 [1].
- Maven 2.1.1 [2]
- JBoss AS 7.1.0 Final [3]
- Apache Tomcat 6.0.32 [4]
- Sesame 2.6.6 [5]
- OWLIM SE 5.1 [6]
- MySQL Community Server GA 5.1.63 [7]

Online installation documentation is available for all required software in case it is needed.

9.3 Installation guidelines

This guide defines the procedure to install the Semantic Web Application Support. For the sake of simplicity, all the commands and procedures included in this guide are oriented to a Linux server, but, as the Semantic Web Application Support is a JEE application, it can be easily installed on a windows server.

9.3.1 Configuring the infrastructure

In order to deploy a working instance of the GE a suitable JEE datasource and Sesame plus OWLIM knowledge base should be needed. In this section we will learn how to deploy a new datasource using MySQL and JBoss AS and how to create a new knowledge base using Sesame console plus OWLIM.

9.3.1.1 Creating a Datasource

The first step towards creating a new MySQL datasource is to install the MySQL JDBC driver. If you have already done it in your JEE container, you can just skip this step. In JBoss AS, we should need to:

- Create a folder in $JBOSS_HOME/modules/com/mysql/main
- Copy the MySQL 5.1 java connector jar file into the created folder.
- Create a module.xml file into the created folder containing:
Once the connector is installed, we would need to add the datasource definition itself into our JEE container. The reference documentation for JBoss datasource deployment descriptor can be found in [8]. To do so, we would need to modify the file

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!--
~ JBoss, Home of Professional Open Source.
~ Copyright 2010, Red Hat, Inc., and individual contributors
~ as indicated by the @author tags. See the copyright.txt file in the
~ distribution for a full listing of individual contributors.
~
~ This is free software; you can redistribute it and/or modify it
~ under the terms of the GNU Lesser General Public License as
~ published by the Free Software Foundation; either version 2.1 of
~ the License, or (at your option) any later version.
~
~ This software is distributed in the hope that it will be useful,
~ but WITHOUT ANY WARRANTY; without even the implied warranty of
~ MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
~ Lesser General Public License for more details.
~
~ You should have received a copy of the GNU Lesser General Public
~ License along with this software; if not, write to the Free
~ Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston,
~ MA
~ 02110-1301 USA, or see the FSF site: http://www.fsf.org.
-->

<module xmlns="urn:jboss:module:1.0" name="com.mysql">
  <resources>
    <resource-root path="mysql-connector-java-5.1.15.jar"/>
  </resources>
  <dependencies>
    <module name="javax.api"/>
  </dependencies>
</module>
```
$JBOS_HOME/standalone/configuration/standalone.xml, including the following description under the <datasources> tag:

```xml
<datasource jta="false" jndi-name="java:jboss/datasources/ExampleDS" pool-name="ExampleDS"
            enabled="true" use-ccm="false">
  <connection-url>CONNECTION_URL</connection-url>
  <driver-class>com.mysql.jdbc.Driver</driver-class>
  <driver>mysql</driver>
  <security>
    <user-name>USER_NAME</user-name>
    <password>PASS</password>
  </security>
  <validation>
    <validate-on-match>false</validate-on-match>
    <background-validation>false</background-validation>
    <background-validation-millis>0</background-validation-millis>
  </validation>
  <statement>
    <prepared-statement-cache-size>0</prepared-statement-cache-size>
    <share-prepared-statements>false</share-prepared-statements>
  </statement>
</datasource>
```

As the GE would try to create the needed tables and sequences, the user provided for the data source connection should have the proper permissions to create, drop, insert or alter tables into the database.

Once defined, you should be able to see the created datasource into your JEE management tool e.g.:
9.3.1.2 Creating a Knowledge Base

In order to create the knowledge repository, the sesame console would be used. To do so, we would run:

```
$ cd SESAME_DIR/bin
$ ./console.sh
```

The following output should be shown in the console:

```
14:02:08.976 [main] DEBUG info.aduna.platform.PlatformFactory - os.name = linux
14:02:08.981 [main] DEBUG info.aduna.platform.PlatformFactory - Detected Posix platform
Connected to default data directory

Commands end with '.' at the end of a line
Type 'help.' for help
>
```

Then, we would need to connect to the registry. To do so:
Disconnected from default data directory
Connected to http://klab.development.atosresearch.eu:8280/openrdf-sesame

Now we can create a new OWLIM-SE repository. To do so:

> create owlim-se.
Please specify values for the following variables:
Repository ID [owlim-se-test]: owlim-se-test
Repository title [OWLIM-SE test repository]:
Storage folder [storage]:
License file (leave blank for evaluation):
Rule-set [owl-horst-optimized]: owl2-rl
Base URL [http://example.org/owlim#]:
Imported RDF files('; delimited):
Default namespaces for imports('; delimited):
Entity index size [200000]:
Total cache memory [80m]:
Main index memory [80m]:
Use predicate indices [false]:
Predicate index memory [0]:
Full-text search memory [0]:
Full-text search indexing policy [never]:
Full-text search literals only [true]:
Use PCSOT index [false]:
Use PTSOC index [false]:
Cache literal language tags [false]:
Repository created
>

In this case, we have created a knowledge base called owlim-se-test that uses owl2-rl rule set. Now we should need to configure our GE binaries to work with the database and knowledge base created.

9.3.2 Deploying the GE
This section describes the steps needed to deploy a working version of the Semantic Web Application Support GE. To do so, we would need to download the binaries from the FI-
WARE private SVN (or contact GE owner to get a configured binarie), configure the GE and deploy the resulting war file in our JEE container.

9.3.2.1 Getting the software
Semantic Web Application Support source code can be downloaded from the FI-WARE SVN (with the proper user and password) by using the following command:

```
```

By doing so, three Maven projects should be downloaded into your current directory:

- ontology-registry-rest-service, that contains the wrapper that allows Rest access style to the GE functionality.
- ontology-registry-service-ejb, that contains the JEE application implementing the ontology registry business logic.
- ontology-registry-test, that contains the software needed to test a clean installation of the GE.

9.3.2.2 Configuring the software
Once the Maven projects are downloaded, we would need to modify the configuration files to connect the GE with a proper data source and knowledge base. To do so we would need to modify two files:

- `/ontology-registry-service-ejb/src/main/resources/META-INF/persistence.xml`: Here we would need to configure the GE to use the data source created in section "Configuring the infrastructure" (or any other suitable database).

```xml
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.0"
xmlns="http://java.sun.com/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/persistence http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd">
    <persistence-unit name="ontology-registry-service-ejb">
        <jta-data-source>DATA_SOURCE_NAME (e.g. java:jboss/datasources/ExampleDS)</jta-data-source>
        <class>eu.atosresearch.ontologyregistry.jpa.OntologyJPA</class>
        <properties>
            <property name="hibernate.hbm2ddl.auto" value="update"/>
        </properties>
    </persistence-unit>
</persistence>
```
Here we would need to configure the GE to connect to the knowledge base created in section "Configuring the infrastructure" (or any other sesame plus owlim knowledge base). We would also need to configure the registry base that will be used to generate the urls for ontologies loaded into the registry.

OntologyRegistry.registryBase=REGISTRY_BASE (e.g. http://localhost:8080/ontology-registry-service/webresources/ontology-registry/ontologies)
OntologyRegistry.repositoryId=KNOWLEDGE_BASE_NAME (e.g. ontology-registry)
OntologyRegistry.sesamePass=KNOWLEDGE_BASE_PASSWORD (e.g. registryPass)
OntologyRegistry.sesameServer=KNOWLEDGE_BASE_SESAME_SERVER (e.g. http://klab.development.atosresearch.eu:8280/openrdf-sesame)
OntologyRegistry.sesameUser=KNOWLEDGE_BASE_USER (e.g. sesameadmin)

After modifying the files, we would need to build the project in order to generate the proper binaries to install. To do so, we should run the following commands:

```
$ cd <directory containing ontology-registry-service-ejb pom.xml file>
$ mvn clean install
$ cd <directory containing ontology-registry-rest-service pom.xml file>
$ mvn clean install
```

A war file (ontology-registry-service.war) to be deployed will be generated under /ontology-registry-rest-service/target

### 9.3.2.3 Deploying the software

Once built, you would need to deploy the ontology-registry-service.war to a JEE container. To do so, you should copy the war file to the deployments directory e.g.:

```
$ cp ontology-registry-service.war $JBOSS_HOME/standalone/deployments
```

You can also use any other deployment tool provided by your JEE container e.g.:
9.4 Sanity check procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

9.4.1 End to End testing

To verify access to the access to the Semantic Application Support Ontology Manager API, just issue the following GET request to the Location GE Host using a web browser:

http://<location GE server ip>:8080/ontology-registry-service/webresources/ontology-registry/sioc_pruned.owl/1.0

The service invoked shall answer in the browser with the owl/rdf serialization of the sioc_pruned ontology stored in the repository.

9.4.2 List of Running Processes

- 1 Tomcat instance, plus the Sesame and OWLIM processes started in Tomcat
- 1 JBoss instance
9.4.3 Network interfaces Up & Open
- TCP:8080 (Tomcat)
- TCP:8280 (Sesame/OWLIM, depending on the configuration of the Sesame, currently hosted in http://klab.development.atosresearch.eu:8280)

9.4.4 Databases
By default, the Semantic Application Support GE does not require a database to function. OWLIM and Sesame may persist the RDF triples using a database as backend (optional). Check the Sesame Manual for more information on this matter.

9.5 Diagnosis Procedures
The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.
The following sections have to be filled in with the information or an “N/A” (“Not Applicable”) where needed. Do not delete section titles in any case.

9.5.1 Resource availability
- For the NeOn Toolkit client, client-side of the enabler: A simple common PC or laptop with at least 2GB RAM is enough to run the NeOn toolkit. The NeOn Toolkit is multiplatform and can be used on Linux or Windows.
- For the server-side API: The main critical requirements comes from the OWLIM deployment. OWLIM provides a documentation that studies different configurations and the hardware requirements, from a single machine with RAM of 4GB and no more than 100GB of hard-disk space, to a cluster of multicore servers with 32GB RAM each. A good on-server configuration requirements are to have at least 12GB of RAM and 500GB of hard disk.

9.5.2 Remote Service Access
N/A

9.5.3 Resource consumption
The resource consumption depends heavily on the amount of triples being uploaded, as OWLIM provides materialization of inferred knowledge in real-time. Nevertheless the usage of OWLIM in the GE does not pose in principle major threads in terms of inference. Therefore no major resource consumption is envisaged for a regular usage of the GE.

9.5.4 I/O flows
N/A
10 Semantic Annotation - Administration and Installation Guide

You can find the content of this chapter as well in the wiki of fi-ware.

10.1 Introduction
This guide covers the steps needed to install and configure the Semantic Application Support GE.

10.2 System Requirements
In order to deploy the Semantic Annotation GE
- Apache 2.2+
- Php 5.3
- Pear Extension
- Curl Extension
- Freeling 2.2 (not tested on Freeling 3.0)
- Mysql 5.1+
Online installation documentation is available for all required software in case it is needed. The System must have outbound access to the web.

10.3 Installation guidelines
This guide defines the procedure to install the Semantic Annotation GE. For the sake of simplicity, all the commands and procedures included in this guide are oriented to a Linux server.

10.3.1 Configuring the infrastructure
Freeling must be properly configured before deploying the software. Please refer to the web documentation to install freeling. In order to perform Query Logging, Semantic Annotation GE needs a mysql user and database to work. With the Mysql root user, enter Mysql console by using:

```
mysql -u root -p
```

once opened the mysql console, create a user and grant all privileges to the newly created user on the newly created database:

```
mysql> CREATE DATABASE sanr;
mysql> CREATE USER 'sanr'@'localhost' IDENTIFIED BY 'mypass';
```
mysql> GRANT ALL ON database TO sanr@'localhost' IDENTIFIED BY 'mypass';

Execute then the following script to create the correct database structure:

```sql
DROP TABLE IF EXISTS `queries`;
CREATE TABLE `queries` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `text` text NOT NULL,
  `snippets` tinyint(1) NOT NULL,
  `timestamp` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=latin1;

DROP TABLE IF EXISTS `results`;
CREATE TABLE `results` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `query_id` int(10) unsigned NOT NULL,
  `lang` varchar(2) NOT NULL,
  `keywords` text,
  `extags` text,
  `freeling` text,
  `topics` text,
  `context` text,
  `terms` text,
  `proc_time` int(10) unsigned NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=latin1;
```

After that the database is ready.

10.4 Deploying the GE

This section describes the steps needed to deploy a working version of the Semantic Annotation GE. To do so, we would need to download the source code from the FI-WARE private SVN

10.4.1 Getting the software

Semantic Web Application Support source code can be downloaded from the FI-WARE SVN (with the proper user and password) by using the following command:
By doing so, a TGZ file should be downloaded in the current working directory, simply UNTAR the files into the web server folder.

```
tar -xcvf sanr.tgz /var/www
```

## 10.4.2 Configuring the software

Main configuration options are stored into /sanr/config/config.php.

```php
<?php

define('DB_HOST', 'localhost');
define('DB_NAME', 'sanr');
define('DB_USER', 'sanr');
define('DB_PASS', '');

//define('PROXY_HOST', '');
//define('PROXY_PORT', '');
//define('PROXY_USER', '');
//define('PROXY_PASS', '');
//define('PROXY_AUTH', CURLAUTH_NTLM);

define('HTTP_TIMEOUT', 15);

define('SPARQL_DBPEDIA', 'http://dbpedia.org/sparql');
define('SKIP_FT_WRAPPER', '');
define('SKIP_TAG_WRAPPER', '');
define('PROVIDE_SNIPPETS', 'false');

define('FREELING_HOME', '');
define('FREELING_PATH', '');
```

The parameters in the config.php file are quite self-explaining:
- DB_HOST, DB_NAME, DB_USER and DB_PASS are the database parameter and must be filled according to the ones set during the previous section of the installation guide. Database is used to log queries and their result, in case "caching" of queries needs to be implemented to increase performance.
- PROXY_HOST, PROXY_PORT, PROXY_USER, PROXY_PASS and PROXY_AUTH are parameters to be uncommented if Semantic Annotator GE stands behind a proxy server. The annotator should reach the wrapper to query them, if a proxy is between the annotator and the wrapper simply put proxy parameters in the config.php and uncomment.
- SPARQL_DBPEDIA: sparql endpoint of dbpedia. Could be a local triple store url also.
- SKIP_*_WRAPPERS: Wrappers to skip.
- PROVIDE_SNIPPET: default value for the provide_snippet option
- FREELING_HOME: Freeling DATA folder
- FREELING_PATH: Freeling path containing Binaries

10.5 Sanity check procedures
The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

10.5.1 End to End testing
To verify access to the access to the Semantic Annotation GE, just issue the following GET request to the Location GE Host using a web browser:

```
http://<base_url>/sanr/ajax/extract_words
```

10.5.2 List of Running Processes
- Apache2 WebServer
- Mysql

10.5.3 Network interfaces Up & Open
- TCP:80 (Apache)

10.5.4 Databases
The Semantic Annotation GE itself does not really require a database to work properly, but a MySql database is used to log queries and their result for caching purposes. Concerning external data sources, it could be linked to any triple store using the config file.
10.6 Diagnosis Procedures
The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section. The following sections have to be filled in with the information or an “N/A” (“Not Applicable”) where needed. Do not delete section titles in any case.

10.6.1 Resource availability
Much depends on the presence or absence of the triple store. If the annotator comes together with a triple store for the annotation using domain-specific datasets, the resource usage will raise consistently as at least 16Gb Ram are needed for a triple store with an average amount of triples stored. In case external service providers are used memory usage should not use more than 2GB RAM

10.6.2 Remote Service Access
The enabler itself is an API which can be recalled by using HTTP GET Requests

10.6.3 Resource consumption
The resource consumption strongly depends on the text to analyze and once again, from the presence/absence of the triple store.

10.6.4 I/O flows
N/A