# D5.4.4 Integrated SemaGrow Stack API components

<table>
<thead>
<tr>
<th>Deliverable Form</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Project Reference No.</strong></td>
</tr>
<tr>
<td><strong>Deliverable No.</strong></td>
</tr>
<tr>
<td><strong>Relevant Workpackage:</strong></td>
</tr>
<tr>
<td><strong>Nature:</strong></td>
</tr>
<tr>
<td><strong>Dissemination Level:</strong></td>
</tr>
<tr>
<td><strong>Document version:</strong></td>
</tr>
<tr>
<td><strong>Date:</strong></td>
</tr>
<tr>
<td><strong>Authors:</strong></td>
</tr>
<tr>
<td><strong>Document description:</strong></td>
</tr>
</tbody>
</table>
# Document History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Author (Partner)</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Draft 0.8</td>
<td>13/12/2013</td>
<td>SWC</td>
<td>Pre-final internally circulated draft</td>
</tr>
<tr>
<td>Draft 0.9</td>
<td>20/12/2013</td>
<td>NCSR-D</td>
<td>Internal review</td>
</tr>
<tr>
<td>Final 1.0</td>
<td>23/12/2013</td>
<td>SWC</td>
<td>Delivered as D5.4.1</td>
</tr>
<tr>
<td>Draft 1.1</td>
<td>16/04/2014</td>
<td>SWC</td>
<td>Various updates, formatting issues, and new figures. Added Section 1.4 on Big Data Aspects.</td>
</tr>
<tr>
<td>Draft 1.2</td>
<td>19/06/2014</td>
<td>SWC</td>
<td>Started installation information and added table in Section 2.4 (now Section 3).</td>
</tr>
<tr>
<td>Draft 1.9</td>
<td>23/6/2014</td>
<td>NCSR-D</td>
<td>Internal review</td>
</tr>
<tr>
<td>Final 2.0</td>
<td>30/6/2014</td>
<td>SWC</td>
<td>Delivered as D5.4.2</td>
</tr>
<tr>
<td>Draft 2.1</td>
<td>13/11/2014</td>
<td>SWC</td>
<td>Added new Section 3 with installation and usage information for the Semagrow Stack WebApp. Extended overall prototype description into new Section 4.</td>
</tr>
<tr>
<td>Draft 2.9</td>
<td>18/11/2014</td>
<td>NCSR-D</td>
<td>Internal review</td>
</tr>
<tr>
<td>Draft 3.0</td>
<td>20/11/2014</td>
<td>SWC</td>
<td>Typos fixed and comments incorporated</td>
</tr>
<tr>
<td>Final 3.1</td>
<td>21/11/2014</td>
<td>SWC</td>
<td>Delivered as D5.4.3</td>
</tr>
<tr>
<td>Draft 3.8</td>
<td>24/07/2015</td>
<td>SWC</td>
<td>Documents new codebase structure (Section 2.3.2, Section 2.3.3, Chapter 4) and dependency on Java8 introduced by work in Task 3.4 (section 2.3.3).</td>
</tr>
<tr>
<td>Draft 3.9</td>
<td>29/07/2015</td>
<td>NCSR-D</td>
<td>Internal review</td>
</tr>
<tr>
<td>Final 4.0</td>
<td>31/07/2015</td>
<td>SWC</td>
<td>Delivered as D5.4.4</td>
</tr>
</tbody>
</table>
EXECUTIVE SUMMARY

This document describes the integrated SemaGrow stack that is developed as a configurable web application. Furthermore it documents the build environment and workflows to create it. It gives an overview of the build environment's hosting server and installed components to be able to create and maintain distributions of the SemaGrow Stack along with the SemaGrow Stack Web Application. Besides that it describes documentation that is automatically generated by the build environment's continuous integration process.
TABLE OF CONTENTS

LIST OF TABLES...................................................................................................................................................... 5

LIST OF FIGURES....................................................................................................................................................... 5

LIST OF TERMS AND ABBREVIATIONS.................................................................................................................... 6

1. INTRODUCTION .................................................................................................................................................. 7
   1.1 Purpose and Scope................................................................................................................................................ 7
   1.2 Approach......................................................................................................................................................... 7
   1.3 Relation to other Work Packages and Deliverables....................................................................................... 7
   1.4 Big Data Aspects............................................................................................................................................ 7

2. BUILD ENVIRONMENT ........................................................................................................................................ 8
   2.1 Introduction.................................................................................................................................................... 8
   2.2 CI Hosting Server.......................................................................................................................................... 8
   2.3 CI Software components............................................................................................................................... 8
      2.3.1 Java....................................................................................................................................................... 8
      2.3.3 Apache Continuum............................................................................................................................... 9
      2.3.4 Apache HTTP Server.......................................................................................................................... 13

3. SEMAGROW STACK WEB APP.......................................................................................................................... 14
   3.1 Installation Process....................................................................................................................................... 14
   3.2 Controlling the SemaGrow Stack............................................................................................................... 16
   3.4 Using the SPARQL Endpoint..................................................................................................................... 16

4. INTEGRATED PROTOTYPE..................................................................................................................................... 21

REFERENCES........................................................................................................................................................... 23
LIST OF TABLES

Table 1: CI Hosting Server................................................................................................................. 9
Table 2: JDK on current Hosting Server............................................................................................. 9
Table 3: Apache Maven on current Hosting Server............................................................................... 9
Table 4: Maven Build Profiles.............................................................................................................. 10
Table 5: Apache Continuum on current Hosting Server...................................................................... 10
Table 6: SemaGrow Project Group...................................................................................................... 12
Table 7: Members of SemaGrow Project Group................................................................................... 12
Table 8: Current Installation URLs..................................................................................................... 21
Table 9: Sample Maven Sites and Reports............................................................................................ 22

LIST OF FIGURES

Figure 1: Continuous Integration Build Sequences............................................................................... 11
Figure 2: Query results as shown on the Web user interface................................................................. 17
Figure 3: Query execution plan as shown on the Web user interface.................................................. 18
## LIST OF TERMS AND ABBREVIATIONS

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>CI</td>
<td>Continuous Integration</td>
</tr>
<tr>
<td>CGI</td>
<td>Common Gateway Interface</td>
</tr>
<tr>
<td>JDK</td>
<td>Java Development Kit</td>
</tr>
<tr>
<td>JRE</td>
<td>Java Runtime Environment</td>
</tr>
<tr>
<td>JNDI</td>
<td>Java Naming and Directory Interface</td>
</tr>
<tr>
<td>OSGI</td>
<td>Open Services Gateway Initiative</td>
</tr>
<tr>
<td>POM</td>
<td>Project Object Model</td>
</tr>
<tr>
<td>RDF</td>
<td>Resource Description Framework</td>
</tr>
<tr>
<td>RPM</td>
<td>Red Hat Package Manager</td>
</tr>
<tr>
<td>SPARQL</td>
<td>Simple Protocol and RDF Query Language</td>
</tr>
<tr>
<td>SQL</td>
<td>Structured Query Language</td>
</tr>
</tbody>
</table>
1. INTRODUCTION

1.1 Purpose and Scope
This document describes the SemaGrow stack integrated prototype and the continuous integration tools and processes that have been set up to create an installable and deployable application. The SemaGrow Stack WebApp serves with its SPARQL Endpoint as the main gateway to SemaGrow technology. This document describes what is necessary to set up a server hosting the continuous integration components and how to set up and maintain the related tools.

This document's scope is limited to the description of the continuous integration process and tools necessary in order to prepare the SemaGrow Stack as an immediately deployable web application.

The description of the inner workings of the SemaGrow Stack's core components is out of scope of this document and is provided in the relevant WP3 and WP4 deliverables.

1.2 Approach
This document describes all necessary components and tools to be able to maintain and extend the compilation of technology produced by the SemaGrow project as well as the integration to a deployable application in a technical manner.

This deliverable describes the working installation of the continuous integration server.

1.3 Relation to other Work Packages and Deliverables
As the integration component of the SemaGrow project it is closely related to all work packages that produce output either in the form code or data, where code is to be understood as software services or applications and data could include default mappings or data summaries. This applies to all such output especially from work packages 3, 4 and 6.

1.4 Big Data Aspects
This deliverable does not face any big-data challenges which is the subject matter of the individual components that this deliverable integrates.
2. BUILD ENVIRONMENT

2.1 Introduction
Continuous integration [1] as an automated build process is crucial to any large software project. Updates, versioning, compiling and making publicly available or deploying of written software can hardly be managed manually on a daily basis. This document describes the efforts undertaken to set up a completely integrated server that manages software builds and deployments as well as making available machine generated documentation.

The core component of the implementation of SemaGrow's continuous integration effort was chosen to be Apache Continuum [2] managing build instructions in the three main components of the SemaGrow Stack, the SemaGrow Stack Modules, the SemaGrow Stack WebApp and the SemaGrow Assembly itself, which is responsible for creating a deployable application in the form of linux packages usable by default linux package managers like aptitude [3] or rpm [4]. The build instructions mentioned above are themselves part of the code in the Apache Maven [5] Project Object Models (pom).

This document describes the instructions in those POMs and shows how they are applied by the continuous integration server.

The integration server is installed on a virtual machine and can be transferred to other hosting environments.

2.2 CI Hosting Server
The CI Hosting Server is defined as being a server having any linux operating system installed. There are now special system requirements regarding the operating system or the hardware, an minimum of 2GB RAM should be available, which is the normally the case on modern hardware. The linux distribution and version of the current installation of the continuous integration server are as described in Table 1.

The server is an OpenVZ [6] container with 2048MB of RAM and 32GB of disk space allocated on a larger server hosting many virtual machines. RAM is limited due to the main purpose of the machine. 32GB of disk space are layed out for long term hosting of future versions of all output of the SemaGrow technology.

2.3 CI Software components
The default continuous integration process requires a couple of components that need to be installed on the hosting server, these components and their respective installation is described below.

2.3.1 Java
A JDK [7] with a minimal version of 1.5 must be installed on the hosting server. The JDK of the current installation is described in Table 2.

The JDK was installed without any customization using Ubuntu's package manager, aptitude.

2.3.2 Apache Maven
Apache Maven with a minimal version of 2.2.1 is required. It is however highly recommended to install Apache Maven 3 or higher. This is also done on the hosting server, where Apache Maven in version 3.1.0 is currently used for the build process. The installed version of maven is given in the Table 3.

Apache Maven 3 has been installed manually according to the installation instructions from Apache Maven's website [8]. At the time of installation only as stable maven version 2.2.1 was available from debian repositories. The manual installation of Apache Maven 3 is however trivial.
In addition to the main Apache Maven Project where the SemaGrow Stack is developed, two Apache Maven Archetypes [9] have been created. Apache Maven Archetypes are code assemblies that define the structure and content of a newly create Apache Maven Project. This is to ensure standard source code and documentation layout for future SemaGrow Stack Modules.

**POM Definitions**

A POM defines in which way the code is built, deployed and documented. Every piece of code of the SemaGrow Stack that is written in the Java Programming Language is managed by a POM, defined in a Apache Maven pom.xml. The current POMs make use of Apache Maven's Lifecycle [10] and Maven Profiles [11]. SemaGrow Stack and SemaGrow Stack Assembly support the Apache Maven Profiles as defined in Table 4.

The Apache Maven Goals and Profiles are managed by the CI Server, described in more detail below.

**2.3.3 Apache Continuum**

The third and most important component of the continuous integration process is Apache Continuum, a dedicated CI server solution managed via a Web interface. Apache Continuum was chosen over CI alternatives, such as Hudson [13], for its ability to build non-maven sources by using shell projects that for example make use of the gcc compiler [12], for it's clean Web interface, and for it's well defined file structure.
### Table 4: Maven Build Profiles

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>semagrow-stack</td>
<td>Create the modules documentation.</td>
</tr>
<tr>
<td>semagrow-stack-http</td>
<td>Create a Tomcat distribution of the SemaGrow Stack endpoint</td>
</tr>
<tr>
<td>semagrow-stack-http</td>
<td>Create a RPM Linux Package</td>
</tr>
<tr>
<td>semagrow-stack-http</td>
<td>Create a Debian Linux Package</td>
</tr>
</tbody>
</table>

### Table 5: Apache Continuum on current Hosting Server

<table>
<thead>
<tr>
<th>Version</th>
<th>Apache-Continuum-1.4.2-SNAPSHOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuum home</td>
<td>/usr/local/apache-continuum-1.4.2-SNAPSHOT</td>
</tr>
<tr>
<td>Continuum bin</td>
<td>/usr/local/apache-continuum-1.4.2-SNAPSHOT/bin</td>
</tr>
</tbody>
</table>

The Apache Continuum Web application runs on an integrated jetty [14] Web server that does not require an extra installation. The server can be started by running the “continuum” binary in Apache Continuum's bin directory. When accessing the web application the first time an admin user must be generated along with an admin password. It should be noted that Apache Continuum 1.4.2 does not run on Java8. To come around the problem that the SemaGrow Stack requires Java8 to be compiled a pointer to a Java7 binary must be set in Apache Continuum's wrapper.conf file. The dependency on Java8 was introduced by the new reactive execution engine (cf. D3.4, Section 4.1), which is implemented using data streaming capabilities introduced in Java8.

The installed version of Apache Continuum is given in Table 5.

**Project Groups**

Apache Continuum supports “Project Groups”, which are assemblies of different projects to be built that have some common requirements. A SemaGrow Project Group was setup (“SemaGrow”), containing build definitions for all major parts of the SemaGrow Stack. The SemaGrow Project Group uses common settings that are summarized in Table 6.

**Build Definitions**

Build Definitions are matched to the project's POM files and run the maven commands as defined by the Apache Maven Profiles described above. These Build Definitions have a fixed schedule defined by an administrator, so that a smooth build chain can be guaranteed. The projects that take part in the SemaGrow Project Group are shown in Table 7.
Figure 1: Continuous Integration Build Sequences
Local Repository | semagrow-maven | semagrow-maven refers to the setting’s name and resolves to /var/www/mvn in the file system.
--- | --- | ---
Build Definition | default-clean | This build definition runs a mvn clean command on every project that is part of the SemaGrow Project Group.
Build Environment | SemaGrow-BE | This refers to the name of a Build Environment, which is terms of Apache Continuum a set of environmental settings which can be expanded upon need.

<table>
<thead>
<tr>
<th>Table 6: SemaGrow Project Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>semagrow</td>
</tr>
<tr>
<td>semagrow-commons</td>
</tr>
<tr>
<td>semagrowcore-api</td>
</tr>
<tr>
<td>semagrow-core</td>
</tr>
<tr>
<td>semagrow-monitor</td>
</tr>
<tr>
<td>semagrow-sail</td>
</tr>
<tr>
<td>semagrow-http</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 7: Members of SemaGrow Project Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>semagrow</td>
</tr>
<tr>
<td>semagrow-commons</td>
</tr>
<tr>
<td>semagrowcore-api</td>
</tr>
<tr>
<td>semagrow-core</td>
</tr>
<tr>
<td>semagrow-monitor</td>
</tr>
<tr>
<td>semagrow-sail</td>
</tr>
<tr>
<td>semagrow-http</td>
</tr>
</tbody>
</table>

The Apache Maven Goals of the members of the SemaGrow Project Group are triggered by Apache Continuum in a well defined timely sequence, which can be seen in Figure 1.

The CI Server environment polls the SemaGrow source code repository [15] at the given time for changes and starts runs the Apache Maven Goal with the given arguments in case there are changes to the source code. A log file is created for every single build. In case of error Apache Continuum will trigger the defined notifiers and inform administrators about the cause of the error.

Build Sequence
The build sequence defines three major parts, first the normal compilation and deployment of Apache Maven Artifacts, second the distribution and assembly part and third the documentation part.

Clean and Install
Apache Continuum will first use the default Maven Goals of “clean” and “install”. These goals will clean the current working directory, compile source written in the Java Programming Language and deploy the finished Apache Maven Artifact [16] to the Local Repository as defined in Table 6.

This Maven Repository is the default deploy target of all parts of the SemaGrow Stack and additionally provides all Apache Maven Artifacts that are defined as dependency in our code. This is to make sure a build
is not hampered by offline Apache Maven Repositories. Other arbitrary Maven Projects that want to make use of SemaGrow Stack technology can define this Apache Maven Repository as a source of dependencies and will be able to include SemaGrow Stack Maven Artifacts as dependencies.

The build process defines the following sequence of the “install” target for the core parts of the SemaGrow Stack:

1. semagrow
2. semagrow-addons
3. semagrow-http

**Package**

As soon as all Apache Maven Artifacts are freshly available in the Local Repository, Apache Continuum will trigger the package and assembly Apache Maven Goals and Profiles of the parts of the SemaGrow Stack Core that provide such features.

First the “distribution” Apache Maven Profile of the SemaGrow Stack WebApp is triggered. This will take the compiled SemaGrow Stack WebApp, download a copy of Apache Tomcat Application Server [17], currently in version 7.0.52, and create a fully functional distribution of a Servlet Container with the SemaGrow Stack WebApp, all necessary dependencies and all necessary configuration files, like a user database, deployed. This Apache Tomcat distribution can be started and will be fully functional to a well defined degree. All parts that make use of external requirements will not work at this stage, for example an external data source might be required in the future that cannot be deployed to an Apache Tomcat Server. The produced Apache Tomcat installation's purpose is also not to be entirely functional, because it represents an intermediate stage that will be finished in the SemaGrow Stack Assembly's build process. The output of the build process is the Apache Tomcat Server as a zip file, which is deployed to the Local Repository as a normal Apache Maven Artifact. As soon as the Apache Tomcat Server is available as an Apache Maven Artifact in the Local Repository Apache Continuum will trigger the two assembly profiles that are defined in the SemaGrow Stack Assembly project.

These profiles make use of the deb [18] and rpm [19] Apache Maven Plugins [20]. These profiles, as defined in their respective POM will download the previously deployed Apache Tomcat Server and assemble a suitable Linux Package. The SemaGrow Stack Http POM defines the structure of the Linux Packages, which will conform to standard linux application layout. The Linux Packages will include any additional requirement and dependency that cannot be assembled into a Tomcat Application Server and provide themselves the necessary binaries, configuration files, install and update scripts, which will for example create SemaGrow user and group on the target linux operating system. The finished linux packages will be deployed on the CI Server's debian and rpm repositories and made available for installation by the target system's respective package manager.

**2.3.4 Apache HTTP Server**

The Hosting CI Server also has a default Apache Httpd [21] installed. This server simply enabled access to directories that are filled by the overall build process. It has been installed on the Hosting Server using Ubuntu's Package Manager aptitude. Additionally it serves as a proxy to Apache Continuum, so that no port information is visible in the Apache Continuum URL. To use Apache HTTP Server is not hard requirement, any other HTTP Server is in principal possible, it is often a matter of choice of a server's administrator.
3. SEMAGROW STACK WEB APP

The SemaGrow Stack WebApp serves as the main gateway to the SemaGrow Stack: it provides the SPARQL endpoint that exposes the federation of endpoints and also host all necessary information for specifying the endpoints that are included in the federation.

The SemaGrow Stack WebApp has been created using the Spring Framework. A simple AJAX [22] framework has been deployed to be able to add additional components and Web services besides the core Stack. The SemaGrow Stack WebApp can be installed on a target system by adding the linux package repository URL (Table 8, Section 4) to the respective Linux Package Manager following normal installation procedures on the target system.

The SemaGrow Stack WebApp exposes a standard SPARQL endpoint capable of doing default content negotiation, returning sparql-results+json as a default content-type for SPARQL Select Queries and content-type text/turtle as a default for SPARQL Construct Queries, text/plain is the only content-type for SPARQL ASK Queries.¹ Standard content-types are available either via HTTP Accept Headers² or via URL Params. The possibility to do content-negotiation either via URL Param or HTTP Accept Headers ensures that the SPARQL Endpoint is fully functional either via an integrated Web Application or as Web Service integrated into any other software component.

3.1 Installation Process

The installation process of the current version of the SemaGrow Stack is done using Aptitude³, the Debian⁴ package manager. We have used Debian based Ubuntu⁵ for our development and testing. To be able to install the SemaGrow Stack package it is first necessary to include the SemaGrow source repository in Aptitude's sources list. This can be done by issuing the following command in a terminal:

```
    sudo sed -i '$ a\deb http://semagrow.semantic-web.at/deb/ lucid free'
    /etc/apt/sources.list
```

Since the debian repository is signed using a self-signed certificate, users should import the repository's key to stop the package manager from asking for installation permissions and if the repository is trusted. This can be done by issuing the following command:

```
    wget -qO - http://semagrow.semantic-web.at/deb/packages.semagrow.key |
    sudo apt-key add -
```

With the following command issued in a terminal, Aptitude will get the repository metadata and the SemaGrow Stack will be ready for installation:

```
    aptitude update
```

---

¹ About SELECT, CONSTRUCT, and ASK queries., please cf. http://www.w3.org/TR/sparql11-query
³ Please cf. https://wiki.debian.org/Aptitude
⁴ Please cf. https://www.debian.org
⁵ Please cf. http://www.ubuntu.com
This command shows the available versions (currently only one) of the SemaGrow Stack that can be installed:

```bash
aptitude search semagrow
```

This command performs the installation:

```bash
aptitude install semagrow
```

The installation process (also formally described in the debian package) creates the following file structure:

```
/usr/
  local/
    semagrow/ (tomcat root)
    bin/
    conf/
    domains (symlink to /var/lib/semagrow/domains)
    lib/semagrow/ (semagrow specific libs)
    logs (symlink to /var/log/semagrow)
    temp/
    work/

/etc/
  init.d/
    semagrow (start/stop script)
    default/
      semagrow/
        metadata.ttl (federation configuration)
        sparql.samples.ttl (sparql samples)

/var/
  run/
    semagrow/ (containing pid file)
  log/
    semagrow/ (containing log files)
  lib
    semagrow/
      domains/
        localhost/
        webapps/
          SemaGrow.war
```

This files structure is the same on every installation and generally follows the default installation structure of Apache Tomcat [17] with the exception of dedicated domains directory, which makes it easier to keep and backup the Web Application.

The most important files for customization of the installation according to local requirements are in `/etc/default/semagrow` where all SemaGrow related configuration files are located and the configuration files under `/usr/local/semagrow/conf/` with which the hosting Tomcat be configured and customized.
3.2 Controlling the SemaGrow Stack

The SemaGrow Stack is controlled by standard Linux start scripts and offer a start, stop, restart and a status parameters. To start the SemaGrow Stack the following command must be issued in a terminal:

```
/etc/init.d/semagrow start
```

This will start the Apache Tomcat WebApplication Server hosting the SemaGrow Stack WebApp, which will expose the SemaGrow SPARQL Endpoint, which wraps the SemaGrow Stack's core technology inside a OpenRDF Sail API implementation. The SemaGrow Stack will expose it's Web Application at the current host's domain + "SemaGrow" (e.g. http://semagrow.semantic-web.at/SemaGrow) at default port 8080. The SPARQL Endpoint itself is available at host:8080 + "SemaGrow/sparql".

This SPARQL Endpoint can be queried either via the web interface or any other client, capable of issuing HTTP requests, like curl. It should be noted that the current installation of the SemaGrow Stack on Semantic Web Company's installation is hidden behind an Apache HTTP Server that proxies requests to all services installed on the demo server. By default the SemaGrow Stack runs on port 8080, which can easily be changed by editing the hosting Apache Tomcat's server configuration.

The SemaGrow Stack can be stopped by issuing the following command in a terminal:

```
/etc/init.d/semagrow stop
```

The federation that is to be hosted and made available via the SemaGrow Stack's installation is configured in the current version using the file metadata.ttl. This file can edited using any text or RDF editor, although it is recommended that the specialized ELEON Authoring Tool (cf. Deliverable 5.2) is used. The default metadata.ttl bundled with the distribution specifies a federation of 10 endpoints from different areas of the life science domain.

The currently loaded configuration can be obtained from the SPARQL endpoint by the following query:

```
PREFIX void:<http://rdfs.org/ns/void#>
SELECT DISTINCT ?endpoint FROM <http://www.semagrow.eu/metadata> WHERE {
    ?x void:sparqlEndpoint ?endpoint
} LIMIT 20
```

The federation configuration is stored locally in the graph http://www.semagrow.eu/metadata All queries on this graph will be answered from this local store and not from the federation.

3.4 Using the SPARQL Endpoint

The SemaGrow Stack WebApp exposes several services. These services can be used programmatically either using asynchronous Ajax calls or by issuing the appropriate requests using any HTTP client. The WebApp also presents a user interface which calls the underlying Web services for presentation and debugging purposes (Figure 2, Figure 3).

6 http://curl.haxx.se
Figure 2: Query results as shown on the Web user interface
Figure 3: Query execution plan as shown on the Web user interface
The SPARQL Endpoint itself is available under the "sparql" service, e.g.:

```
http://example.com/SemaGrow/sparql
```

It expects an URL encoded SPARQL 1.1. (non update) query given as the "query" URL parameter in a POST request using either "application/sparql-results+json" or "application/sparql-results+xml" as an accept header for SELECT queries and a common rdf mime type for CONSTRUCT queries.

```
```

Besides the main querying service, the SemaGrow Stack WebApp also exposes two additional Web services that show the inner workings of the SemaGrow Stack technology. First the "explain" method which can be either used via the Web interface or a curl command shows the query execution plan produced by the Sesame query execution engine. The second method "decompose" shows the query execution plan produced by the SemaGrow decomposition component (cf. Deliverable 3.4.2).

The SPARQL Engine will work on a query plan that can be seen by using the "decompose" Web service which shows what endpoint will actually be queried:

```
```

The resulting query plan can be seen below, it shows how SemaGrow incorporates SPARQL Endpoint statistics into it's query plan by calculating a the cost of querying an endpoint and how SPARQL Endpoints that do not offer suitable information according to the federation's setup are not touched.

```
QueryRoot
  Projection
    ProjectionElemList
      ProjectionElem "title"
      ProjectionElem "abstract"
    Plan (cost = 1.6021567E7, card = 2535579)
    BindJoin
      Plan (cost = 1926574.0, card = 1926569)
        SourceQuery (source = http://202.45.139.84:10035/catalogs/fao/repositories/agris)
          StatementPattern
            Var (name=p)
            Var (name=abstract)
          Plan (cost = 3.273639E7, card = 10912125)
```
D5.4.4 Integrated SemaGrow Stack API components


StatementPattern
  Var (name=p)
    Var (name=title)

The difference is easily spottable when comparing the query plan of the decomposed query, created by the SemaGrow Stack’s core technology against the query plan delivered by default SPARQL engine, which would query all and every single SPARQL Endpoint in the federation, where the SemaGrow SPARQL Engine restricts queries to those endpoints that are known to host useful information with respect to the query. The standard query plan can be obtained by issuing the following curl call:


The resulting query plan can be seen below:

Projection
  ProjectionElemList
    ProjectionElem "title"
    ProjectionElem "abstract"
  Join
    StatementPattern
      Var (name=p)
        Var (name=title)
    StatementPattern
      Var (name=p)
        Var (name=abstract)
4. INTEGRATED PROTOTYPE

Integrated part of this deliverable is a working installation of the described components responsible for continuous integration along with visible output from the described build process. This chapter gives an overview of the components and outputs that are visible to the outside world.

Integrated part of the overall build process is automatically generated documentation in the form of Maven Sites. These documentation pages are already been created by the Build Definitions defined above. These documentations are especially interesting for developers wanting to make use of SemaGrow Stack Core technology.

The documentation pages contain information on how to build the different parts of the SemaGrow Stack in form of technical documentation that layout for example every Apache Maven Goal and Profile and enables administrators to set up a own build process. This is to ensure that possible future developments have a basis for further developments. Table 9 gives samples of these documentations and automatically generated reports.

The SemaGrow Stack API is based on the Sesame SAIL API. Specifically, the integrated Semagrow System comprises:

- The SemaGrow Stack WebApp (Section 3), which exposes the SemaGrow Stack API as a SPARQL endpoint. The SemaGrow Stack API is implemented by the SemaGrow Stack of RDF4J SAIL implementations.
- The SemaGrow Stack, comprising the prototypes developed in WP3 and WP4 wrapped under Sesame SAIL APIs.
- The ELEON Authoring Tool for visualizing and editing the descriptions of the federated endpoints required by the SemaGrow Stack (Task 5.1)
- The various off-stack tools that make up the SemaGrow ecosystem (Task 5.1)

The list of visible URLs is shown in Table 8.

<table>
<thead>
<tr>
<th>Component</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache Continuum</td>
<td><a href="http://semagrow.semantic-web.at/continuum/">http://semagrow.semantic-web.at/continuum/</a></td>
</tr>
<tr>
<td>SemaGrow Maven Repository</td>
<td><a href="http://semagrow.semantic-web.at/mvn/">http://semagrow.semantic-web.at/mvn/</a></td>
</tr>
<tr>
<td>SemaGrow Maven Sites</td>
<td><a href="http://semagrow.semantic-web.at/docs/">http://semagrow.semantic-web.at/docs/</a></td>
</tr>
<tr>
<td>SemaGrow Debian Repository</td>
<td><a href="http://semagrow.semantic-web.at/deb/">http://semagrow.semantic-web.at/deb/</a></td>
</tr>
<tr>
<td>SemaGrow RPM Repository</td>
<td><a href="http://semagrow.semantic-web.at/rpm/">http://semagrow.semantic-web.at/rpm/</a></td>
</tr>
<tr>
<td>SemaGrow StackWebapp</td>
<td><a href="http://semagrow.semantic-web.at/SemaGrow/">http://semagrow.semantic-web.at/SemaGrow/</a></td>
</tr>
<tr>
<td>SemaGrow Modules Source</td>
<td><a href="https://github.com/semagrow/semagrow/releases/tag/1.4.0">https://github.com/semagrow/semagrow/releases/tag/1.4.0</a></td>
</tr>
</tbody>
</table>

| Table 8: Current Installation URLs |

---

### Table 9: Sample Maven Sites and Reports

<table>
<thead>
<tr>
<th>Site</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="https://example.com/logo.png" alt="Logo" /></td>
<td><img src="https://example.com/image.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="https://example.com/chart.png" alt="Chart" /></td>
<td><img src="https://example.com/diagram.png" alt="Diagram" /></td>
</tr>
</tbody>
</table>

| ![Diagram](https://example.com/diagram.png) | ![Image](https://example.com/image.png) |
| ![Chart](https://example.com/chart.png) | ![Logo](https://example.com/logo.png) |
REFERENCES

[3] https://wiki.debian.org/Aptitude
[18] https://github.com/tcurdt/jdeb