D4.1 Scalability & Robustness Experimental Methodology

<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 v0.10</td>
<td>04/11/2013</td>
<td>NCSR-D, UAH</td>
<td>Document Setup</td>
</tr>
<tr>
<td>Draft v0.5</td>
<td>18/11/2013</td>
<td>NCSR-D</td>
<td>Section 2, Section 3</td>
</tr>
<tr>
<td>Draft v0.8</td>
<td>25/11/2013</td>
<td>NCSR-D, UAH</td>
<td>Section 4, Section 5</td>
</tr>
<tr>
<td>Draft v1.0</td>
<td>09/12/2013</td>
<td>AK, NCSR-D, UAH</td>
<td>Internal Review, Revisions</td>
</tr>
<tr>
<td>Final v1.0</td>
<td>23/12/2013</td>
<td>NCSR-D, UAH</td>
<td>Delivered as D4.1</td>
</tr>
</tbody>
</table>
**EXECUTIVE SUMMARY**

Given the scope, complexity and innovative aspect of the proposed SemaGrow solution, it is imperative that a concrete, cohesive methodology for testing the efficiency and performance of the various technological components is established.

The present report aims to lay the foundation for the rigorous testing methodology of SemaGrow, by:

1. Identifying the critical points that impact the performance of the overall SemaGrow solution
2. Establishing the measures and metrics for estimating the efficiency of the core components
3. Specifying the requirements for designing and implementing automatic testing components that will monitor the performance of the SemaGrow Stack under realistic and projected conditions

As the experimentation on the various SemaGrow components proceeds, the Scalability & Robustness Experimental Methodology will accordingly be refined in order to reflect the evolving real-world settings and needs identified by the undergoing experiments. Furthermore, the scalability of the overall SemaGrow solution will be tested in sync with the increase on data volume, as well as, over realistic generated data. The findings of these experiments will guide the refinement of the testing methodology and the updates on the corresponding automated rigorous testing components.
# TABLE OF CONTENTS

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

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

1. **INTRODUCTION** .............................................................................................................. 6
   1.1 Purpose and Scope ........................................................................................................ 6
   1.2 Approach for Work Package and Relation to other Work Packages and Deliverables ... 6
   1.3 Methodology and Structure of the Deliverable ........................................................... 6

2. **SEMA GROW ARCHITECTURE** .................................................................................. 7
   2.1 The SemaGrow Stack .................................................................................................. 7
   2.2 SemaGrow Stack Workflow ....................................................................................... 9
   2.3 Critical Points for Measuring System Performance .................................................... 9

3. **MEASURING THE EFFICIENCY OF DISTINCT SEMAGROW COMPONENTS** .......... 10
   3.1 Indexing Algorithms ................................................................................................. 10
      3.1.1 Storage and Retrieval of Data Summaries ......................................................... 10
      3.1.2 Source Selection ............................................................................................... 11
   3.2 Query Decomposition and Rewriting ......................................................................... 11
   3.3 Ontology Alignment .................................................................................................. 12

4. **MEASURING THE EFFICIENCY OF THE SEMAGROW STACK** ............................. 13
   4.1 Overall Performance ................................................................................................. 13
   4.2 Scalability .................................................................................................................. 13

5. **CONCLUDING REMARKS** ......................................................................................... 15
LIST OF FIGURES

Figure 1: The SemaGrow Stack ............................................................. 7
Figure 2: Generator of Realistic Project Datasets ..................................... 13

LIST OF TABLES

Table 1: Performance Measures for the Core SemaGrow Research Outcomes ........................................... 10
Table 2: Requirements for the Automated Rigorous Testing Components ........................................... 15
1. INTRODUCTION

1.1 Purpose and Scope

Task T4.1 will develop and implement a rigorous automated testing methodology that will allow the project to reliably measure and compare the efficiency of the developed research components system under realistic conditions. The methodology will take into account individualities that occur due to distinct properties of the system such as the heterogeneity and the distributed nature of the repositories. It will also include a number of requirements for developing a rigorous testing component/environment within the semantic infrastructure that will be developed in the context of the project.

1.2 Approach for Work Package and Relation to other Work Packages and Deliverables

The core objective of Work Package 4 is to design and implement the necessary processes for the rigorous testing of the proposed SemaGrow solution. More specifically, WP4 aims to:

a) Develop an automated method that will realize the continuous measurement of the system’s efficiency;

b) Collect operative data from the system and identify the relevant metrics for spotting congestion phenomena;

c) Build on existing benchmarks in order to implement the necessary infrastructure for quantifying the efficiency of the SemaGrow components under realistic assumptions.

As the scope of the testing methodology and the corresponding components is the whole SemaGrow system, the work package has a high degree of interaction with all the technical tasks. More specifically, the methodology is informed by the characteristics and goals of the components developed in the context of all WP3 tasks. Additionally, it covers the whole SemaGrow system, which will be integrated in the context of WP5.

1.3 Methodology and Structure of the Deliverable

Taking into account the currently established SemaGrow Stack architecture, as well as, the specification of the SemaGrow maintenance components, the present document reports on the processes for measuring the performance and efficiency of the major components in isolation, as well as, the performance of the overall SemaGrow solution. Section 2 provides a brief overview of the SemaGrow Stack and identifies the critical points that have an impact to the performance and efficiency of the system. Section 3 provides details on the methodologies for measuring the performance of individual components, while Section 4 presents our approach for measuring the performance of the overall system. Finally, Section 5 specifies the requirements for the automatic rigorous testing components that will be integrated in the infrastructure and implement the proposed testing methodology.
2. SEMAGROW ARCHITECTURE

2.1 The SemaGrow Stack

The SemaGrow Stack integrates the components needed in order to offer a single SPARQL endpoint that federates a number of heterogeneous data sources, also exposed as SPARQL endpoints. The main difference between the SemaGrow Stack and most existing distributed querying solutions is that SemaGrow targets the federation of heterogeneous and independently provided data sources. In other words, SemaGrow aims to offer the most efficient distributed querying solution that can be achieved without controlling the way data is distributed between sources and, in general, without having the responsibility to centrally manage the data sources of the federation.

Figure 1 depicts the architecture of the SemaGrow Stack.

![Figure 1: The SemaGrow Stack](image)

The core components of the SemaGrow stack are the following:
The SemaGrow SPARQL endpoint: The SemaGrow SPARQL endpoint is the Web service through which client applications access SPARQL endpoints that have been federated using the SemaGrow Stack.

The Query Decomposition Component: The query decomposition component analyses SPARQL queries and decides about the optimal way to break them up into query fragments to be dispatched to sources’ endpoints. The query decomposition component comprises a decomposer module that syntactically analyses queries and suggests possible decompositions and a selector module that evaluates these suggestions using information and predictions from the resource discovery component about the data sources where each query fragment can be executed.

The result is a matching between query fragments and the source that each fragment is to be dispatched to. This is by necessity an approximation, since completeness can only be guaranteed by querying all sources. This is guided by the reactivity parameters that specify the client application’s wished position in the trade-off between efficiency and completeness in terms of how much time it is possible and worth it to wait to get more results, or the minimum number of results required, or some other similar policy balancing between completeness and effort.

The Resource Discovery Component: The resource discovery component provides an annotated list of candidate data sources that possibly hold triples matching a given query pattern; including sources that follow a different (but aligned) schema than that of the query pattern. The sources are annotated with schema and instance-level metadata and predicted response volume from the data summaries endpoint; as well as run-time information about current source load. When a source following an aligned schema is used, the annotation also includes relevant meta-information, such as the semantic proximity of the query schema and the source schema.

The Data Summaries Endpoint: The data summaries endpoint serves metadata about the schema used and the instances stored in the various federated data stores. In particular, it indexes in the opposite direction than conventional data catalogues, by receiving entity URIs and responding with the repositories where triples involving these entities are located. Entities can be both at the schema level (classes, properties) and the instance level. Furthermore, it serves ontology alignment knowledge regarding entity equivalences between different sources. Although in principle any repository infrastructure can be used to store these data summaries, it is one of the core research objectives of SemaGrow to experiment with the POWDER protocol in order to take advantage of naming convention regularities to compress such indexes; and to develop a triple store that is especially well suited to serving metadata expressed using POWDER.

The Federated Endpoint Wrapper: The Federated End-point Wrapper manages the communication with the external data sources that are federated by the SemaGrow Stack. Its Query Manager module is responsible for (a) where necessary, applying the Query Transformation Service to access repositories that follow a different schema than the one of the original query; (b) forwarding query fragments to the Query Results Merger; and (c) collecting and forwarding dynamic run-time statistics to the Resource Discovery components.

The Query Transformation Service applies alignment knowledge served from the schema mappings repository. It re-writes query fragments from the schema of the original query to that of the data source that will be used for each fragment and also query results back into the schema of the original query so that they can be joined with results from other sources.

As joining distributed query results can degenerate into a situation where massive data volumes need to be copied to and processed by the results collector, SemaGrow envisages a Query Results Merger that exhibits pay-as-you-go behaviour, providing a first approximation with minimal usage of computational resources and iteratively refining it if more computation time and space are warranted by the reactivity parameters set by the client application. Distributed incremental result fetching operators exhibit this property, so that results can be incrementally requested and forwarded on arrival of new tuples. The confidence of mappings can be taken into account, offering higher priority to more certain mappings when needed, as per user requirements.
2.2 SemaGrow Stack Workflow

Populating and updating the individual repositories of the federation is performed independently for each repository as the SemaGrow system does not exert any control or influence over the way data is distributed between the members of the federation. In the event that population functionality is deemed to be useful (to be decided at a later stage), a SERVICE clause must obligatorily specify the source that will be updated.

Queries that are received by the SemaGrow SPARQL endpoint are forwarded to Query Decomposition, which produces and evaluates alternative ways to decompose queries into pairs consisting of query fragments and the data source where each fragment will be executed. During this, Query Decomposition sends query patterns to Resource Discovery, which responds with candidate data sources for each pattern and annotations pertaining to the source schema, volume of matching triples, and current source load. Resource Discovery queries the data summaries and the schema mapping repositories for this information; Resource Discovery is informed of the data source's current load via the control channel from the Federated Endpoint Wrapper.

Upon deciding about the optimal way to break up the query into fragments, these are forwarded to the Federated Endpoint Wrapper. Where necessary, the Federated Endpoint Wrapper uses information from the schema mapping repository to translate the fragment into one using the same vocabulary as the source. Then, the Federated Endpoint Wrapper initializes query connections with all relevant sources and starts collecting responses; the Federated Endpoint Wrapper might notify Query Decomposition of problems such as unavailable sources and request an alternative query decomposition that does not rely on such sources.

Once queries have started executing on the endpoints, the Federated Endpoint Wrapper receives the result triples as they become available and (where necessary) translates them back into the vocabulary of the original query. In this incremental receipt, non-blocking join operators applied on triples merge individual parts of the queries into response tuples. The query decomposition strategy, together with the join operators themselves, allow the incremental handling of massive result sets, allowing the use of secondary memory. Temporarily blocked sources and delays are deferred by returning tuples as they become available without hurting the final result set. This approach is also applicable for streams and, aggregate queries, and in general all contexts foreseen by the SemaGrow use cases.

The results are finally sent to the SemaGrow SPARQL endpoint to be forwarded to the client application.

2.3 Critical Points for Measuring System Performance

Throughout the described information flow for a client application query, there are several points on which the overall performance of the SemaGrow solution depends.

*Producing, Storing and Retrieving Data Summaries:* The compression achieved by the data summary is a crucial factor for the performance of the system as it could have a major impact on the reduction of the search space, and thus the execution time, of the queries to be processed.

*Discovering Suitable Resources:* The discovery of the appropriate sources that should be queried for a given query pattern is of major importance for the SemaGrow solution. There are two important requirements for the discovery process: First, it is the process that eliminates the need for querying all the underlying federated repositories. Furthermore, it should incorporate the necessary operations that guarantee the discovery of the sources that hold content relevant to the given query fragments.

*Decomposing and Rewriting Queries:* The queries submitted by the client applications should be decomposed in query patterns in such a way that a good load balancing for the different sources is achieved. Additionally, query rewriting should be limited as possible, as it introduces an additional overhead on the execution of the described workflow.

*Transforming Queries in Different Schemas:* The query transformation service is responsible for querying sources that follow a different schema than the one of the original query (when necessary). In this case, the issues that are raised are related to the accuracy of the proposed transformations. The transformation process will have to include data sources that provide relevant content, without introducing actually non-relevant repositories to the process.
3. MEASURING THE EFFICIENCY OF DISTINCT SEMAGROW COMPONENTS

Taking into account the aforementioned critical points of the query execution workflow, we describe the metrics that will be computed by the SemaGrow components involved in these points in order to acquire a good indication of their individual efficiency. Table 1 summarizes the core SemaGrow research outcomes, their correspondence with the major components of the SemaGrow Stack and the metrics that will be used for evaluating the success of the respective outcomes.

<table>
<thead>
<tr>
<th>SemaGrow Research Outcomes</th>
<th>Relevant Stack Components</th>
<th>Performance Measures</th>
</tr>
</thead>
</table>
| Novel indexing algorithms that support the efficient storage and retrieval of data summaries that concisely describe instance-level metadata about the different sources federated under the SemaGrow infrastructure. | SemaGrow POWDER Triple Store, Resource Discovery Component | a) Size of the data summaries as a function of the total size of the federated repositories  
        b) Overhead of the method as a function of the time it would take to query all repositories  
        c) Accuracy of the source selection in predicting which sources hold data that satisfy a given query. |
| An extension of state-of-the-art query decomposition and rewriting methods that will enable complex queries in one schema to be broken down into sub-queries, each in a (possibly) different schema. | Query Decomposition Component, Query Transformation Service, Ontology Alignment Component | a) Load balancing between sources  
        b) Minimizing rewriting  
        c) F-score of the tuples retrieved when compared with the tuples retrieved by hand-crafted queries. |
| The integration of a variety of state-of-the-art schema alignment methods under a novel architecture for the prior selection of the most appropriate method or methods for a given schema pair, the synthesis of multiple methods into a unified alignment, and the posterior evaluation of alignment quality. | Ontology Alignment Component | a) Precision and Recall  
        b) Performance of the alignment synthesis versus the performance of the best of the components being synthesised |

Table 1: Performance Measures for the Core SemaGrow Research Outcomes

The following subsections provide details on the performance measurement for each of the aforementioned research outcomes.

3.1 Indexing Algorithms

3.1.1 Storage and Retrieval of Data Summaries

In order to estimate the efficiency of the POWDER Triple Store as a means to produce meaningful and manageable data summaries, the system will measure the size of the produced data summaries as a fraction of the total size of the federated repositories.

\[
\text{Compression} = \frac{\text{Size(Summary)}}{\sum_{i=1}^{n} \text{Size(Rep}o_i)}
\]
3.1.2 Source Selection

The Resource Discovery Component of the SemaGrow stack is responsible for selecting the sources holding data that satisfy a given query pattern. A first metric for its efficiency is time-centric and is expressed as the overhead of the resource discovery process compared to the time it would take to query all repositories:

\[
Gain(q) = \frac{T_{\text{selection}}}{T_{\text{complete}}}
\]

Where \( q \) is a given query pattern, \( T_{\text{selection}} \) is the time dedicated to the resource discovery process and \( T_{\text{complete}} \) is the time needed for querying all the underlying repositories with the query pattern.

Furthermore, it is important to compute the accuracy of the resource discovery process, by determining the correctness and the completeness of the set of proposed data sources. Correctness corresponds to the traditional precision metric and is defined as the percentage of the proposed data sources that actually hold relevant content for a given query pattern.

\[
Precision(q) = \frac{\text{Count}(R_q)}{\text{Count}(S_q)}
\]

Where \( R_q \) is the subset of the set of proposed sources \( S_q \) that returned a non-empty result set for the query pattern \( q \).

\[
R_q = \{ s \in S_q : \text{Result}(s, q) \neq \emptyset \}
\]

Completeness is defined as the fraction of the proposed data sources in relation to the amount of the data sources that hold content relevant to the examined query pattern.

\[
Recall(q) = \frac{\text{Count}(S_q)}{\text{Count}(T_q)}
\]

Where \( T_q \) is the set of sources that hold content relevant to the query pattern \( q \).

3.2 Query Decomposition and Rewriting

Success will be measured in terms of selecting the optimal decomposition, achieving good load balancing between sources while at the same time minimizing rewriting, especially when the source schema is structurally distanced from the query schema. It is evident that different decompositions will lead to the execution of different queries over a different subset of the underlying federated repositories. Hence, testing the quality of the selected decomposition should take into account the following parameters.

1. **Load Balancing**: Load balancing will be measured by calculating the reduction in execution time when applying different decomposition choices, comparing the execution time of the selected decomposition with the execution time of other possible decompositions.

2. **Minimize Rewriting**: Query rewriting should be kept to the minimum amount that guarantees completeness, since the rewriting process introduces ambiguity and approximations. To this end, the different possible decomposition results will be evaluated with respect to the confidence on the rewriting patterns, as delivered by the Schema Mappings repository.

3. **Precision and Recall**: Ultimately, the results retrieved by the execution of the query fragments will be compared with the results retrieved from hand-crafted, targeted queries, in order to evaluate the integrity of the selected decomposition.
3.3 Ontology Alignment

The results of the SemaGrow Ontology Alignment Component will be evaluated using the traditional Precision and Recall metrics.

What will be added is the comparison between the results of the synthesis process and the results of the best of the underlying individual alignment methods, in order to estimate the benefits of using and combining different alignment methods.

Furthermore, an important aspect for the Ontology Alignment module is its scalability. To this end, we will compute the overhead of the synthesis process compared to the underlying methods, calculating the time needed for synthesizing the results of the different methods as a fraction of the execution time of the slowest individual method.

\[
\text{Overhead}(O_1, O_2) = \frac{T_{\text{Synthesis}}}{\max_{\text{methods}} T_{\text{method}}(O_1, O_2)}
\]
4. MEASURING THE EFFICIENCY OF THE SEMAGROW STACK

4.1 Overall Performance

The overall efficiency of the SemaGrow Stack is, to a great extent, dependent on the query response time for the queries submitted by the client applications. We can distinguish two different performance criteria for the SemaGrow solution:

1. Responsiveness, measured as the time to retrieve the first query result and time between successive query results
2. Throughput, measured as the time to retrieve all query results.

The queries used for these measurements will be derived by combining elicitation from domain experts with analysis of the query logs of the currently deployed services as well as of the logs of the early SemaGrow deployments.

4.2 Scalability

The scalability of the SemaGrow system, with respect to the data volume and the number of providers / collections, will be continuously tested as the amount of available data increases. Furthermore, we will test the performance of the system for the projected data volume in 2020, using realistic artificially generated data.

The data projection tool (see Figure 2) is based on the distributions observed over the existing data collections. The statistical analysis builds the probability distributions for the different data elements, which are subsequently used by a Monte Carlo simulation module that generates simulated data. The product of the Monte Carlo simulation is compiled by the Simulated Data Builder in cohesive data collection files. Finally, these files are triplicated and are ready to be injected in a Triple Store and be used for testing.

![Diagram of Data Collection Process](image)

**Figure 2**: Generator of Realistic Project Datasets

Following the described design, and in order to initiate experiments on large-scale artificial data at an early stage of the project, generator modules have been implemented for the LAFLOR, Ariadne, AgLR, Trees4Future and Eururalis collections. We will proceed to develop generators for every data collection used within SemaGrow, and test the system as the volume and number of collections increase. The presented approach for building large-scale realistic simulated datasets based on real-world data collections was well-received by the experts members of the ESG group.

As a result of the data injection and generation process, the Scalability & Robustness Experimental Methodology should evolve to cover the needs and findings of the undergoing experiments. Therefore, the testing methodology will be...
refined in sync with the progress of the SemaGrow components. To this end, two additional (not originally foreseen) versions of the methodology will be delivered, in M18 and M24 of the project, reporting on the overall findings and presenting the necessary changes on the rigorous testing components of SemaGrow.
5. CONCLUDING REMARKS

The present report showcases the main considerations for evaluating the efficiency and performance of the SemaGrow solution, and provides an initial description of the testing methodology to be followed, defining the relevant metrics for each component as well as the overall system. Furthermore, we present our approach for testing the scalability of the proposed architecture, and report on the design and development of the modules that will handle the production of large-scale simulated data that will be used for this purpose.

In the following period, we will proceed with the implementation and integration of the first version of the automated testing components that will be included in the system in order to provide estimates of its efficiency. Besides implementing the proposed methodology, the components must satisfy certain functional requirements in order to guarantee the accurate, continuous and evolving evaluation of the system’s performance. These are summarized in the following table.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unobtrusiveness</td>
<td>The measurement of the relevant metrics should not interrupt or interfere the execution of the actual process tested</td>
</tr>
<tr>
<td>Uptime</td>
<td>The testing components should be active for all queries, having an uptime of 99.9%</td>
</tr>
<tr>
<td>Reporting</td>
<td>The results of the testing components should be made available on demand via a clearly defined API and should also be stored permanently in an appropriate data structure</td>
</tr>
</tbody>
</table>

Table 2: Requirements for the Automated Rigorous Testing Components

The experiments over real and simulated data will lead to revisions and refinements of the rigorous testing methodology and, as a result, updates on the respective components. These revisions will be reported in updated versions of this document. The first one will be delivered in M18 of the project, while the final version of the methodology and the associated testing components will be delivered in M24.