Deliverable D8.2b
Proof-of-Concept – Report on Validation

Authors
THALES
MoMa
ATOS
TIE
SIEMENS
CINI

Due date of deliverable: 30/06/2010
Actual submission date: 30/06/2010

This work is licensed under the Creative Commons Attribution-Share Alike 3.0 License.
To view a copy of this license, visit http://creativecommons.org/licenses/by-sa/3.0/ or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.

This work is partially funded by EU under the grant of IST-FP6-034763.
### Change History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Status</th>
<th>Author (Partner)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>14/03/2010</td>
<td>Draft</td>
<td>Thales</td>
<td>1st version of document layout</td>
</tr>
<tr>
<td>0.2</td>
<td>24/05/2010</td>
<td>Draft</td>
<td>Thales</td>
<td>Update with five PoCs Template ready to be fulfilled by PoC’s owner</td>
</tr>
<tr>
<td>0.3</td>
<td>31/05/2010</td>
<td>Draft</td>
<td>Thales</td>
<td>Updated contribution</td>
</tr>
<tr>
<td>0.31</td>
<td>02/06/2010</td>
<td>Draft</td>
<td>Thales</td>
<td>Reviewed by Pascal B. Addressing remarks of Pascal Bisson</td>
</tr>
<tr>
<td>0.4</td>
<td>15/06/2010</td>
<td>Draft</td>
<td>Siemens</td>
<td>Contribution of Siemens (Evelyn Pfeuffer)</td>
</tr>
<tr>
<td>0.4.1</td>
<td>18/06/2010</td>
<td>Draft</td>
<td>Thales</td>
<td>Update the template according to D8.1c (number of PoCs)</td>
</tr>
<tr>
<td>0.5</td>
<td>27/06/2010</td>
<td>Draft</td>
<td>MoMa CINI ATOS TIE</td>
<td>Contributions of MoMa (Angelo Gaeta) CINI (Andrea Janes) ATOS (Francisco Javier Nieto De-Santos) TIE (Vadim Igorevich Chepegin)</td>
</tr>
<tr>
<td>0.5.1</td>
<td>28/06/2010</td>
<td>Draft</td>
<td>Thales</td>
<td>Reviewers of Pascal Bisson</td>
</tr>
<tr>
<td>0.5.2</td>
<td>01/07/2010</td>
<td>Draft</td>
<td>Thales</td>
<td>Addressing the Pascal’s comments by all partners</td>
</tr>
<tr>
<td>0.5.3</td>
<td>02/07/2010</td>
<td>Draft</td>
<td>Thales</td>
<td>Addressing the comments of Mike Fischer (Peer Review)</td>
</tr>
<tr>
<td>0.6</td>
<td>09/07/2010</td>
<td>Final</td>
<td>Thales</td>
<td>Final version</td>
</tr>
</tbody>
</table>
EXECUTIVE SUMMARY

This deliverable provides a detailed assessment of each of the Architectural choices and/or patterns demonstrated through the second set of Proof-of-Concepts selected and set-up (so called PoCs Phase II/Year 2). This report is an input to NEXOF-RA Architecture Team (led by CA and WP7 lead) to further shape NEXOF Reference Architecture specifications.
### Document Information

<table>
<thead>
<tr>
<th>IST Project Number</th>
<th>Acronym</th>
<th>NEXOF-RA</th>
</tr>
</thead>
<tbody>
<tr>
<td>FP7 – 216446</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Full title</th>
<th>Project URL</th>
<th>EU Project officer</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Deliverable Number</th>
<th>Title</th>
<th>Proof-of-Concept – Report on Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>D8.2b</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Work package Number</th>
<th>Title</th>
<th>Proof-of-Concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Date of delivery</th>
<th>Contractual</th>
<th>Actual</th>
</tr>
</thead>
<tbody>
<tr>
<td>30/06/2010</td>
<td>30/06/2010</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Status</th>
<th>Nature</th>
<th>Abstract (for dissemination)</th>
</tr>
</thead>
<tbody>
<tr>
<td>final</td>
<td>Report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Demonstrator</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td></td>
</tr>
</tbody>
</table>

| Keywords | |
|----------| |

| Authors (Partner) | |
|-------------------| |
| THALES, MoMa, TIE, ATOS, SIEMENS, CINI | |

<table>
<thead>
<tr>
<th>Responsible Authors</th>
<th>Email</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Phong CAO, Pascal BISSON</td>
<td><a href="mailto:phong.cao@thalesgroup.com">phong.cao@thalesgroup.com</a></td>
<td><a href="mailto:pascal.bisson@thalesgroup.com">pascal.bisson@thalesgroup.com</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Partner</th>
<th>Phone</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>THALES</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS

EXECUTIVE SUMMARY ........................................................................................................... 3

TABLE OF CONTENTS ............................................................................................................... 5

TABLE OF TABLES ....................................................................................................................... 8

TABLE OF FIGURES ....................................................................................................................... 9

1 INTRODUCTION ....................................................................................................................... 11

  1.1. Purpose and scope ............................................................................................................. 11
  1.2. Document overview ......................................................................................................... 11

2 ASSESSMENT OF PoC8: NoSQL STORAGE - MoMA ................................................................. 12

  2.1 PoC’s selection ..................................................................................................................... 12
  2.2 Pattern: Not only SQL (No SQL) Storage ........................................................................... 12
      2.2.1 Scope of the PoC with respect the Reference Architecture ........................................ 13
      2.2.2 Evaluation methodology ............................................................................................... 13
      2.2.3 Evaluation results .......................................................................................................... 15
      2.2.4 Conclusion ...................................................................................................................... 19
  2.3 Report of Assessment of Requirements (Objectives) of Scenario ...................................... 21
      2.3.1 Link between requirements and quality attributes ....................................................... 21
      2.3.2 Evaluation of requirements ............................................................................................ 21
      2.3.3 Conclusion ..................................................................................................................... 22

3 ASSESSMENT OF PoC9: PERMIS RBAC - THALES ................................................................. 23

  3.1 PoC’s selection ..................................................................................................................... 23
  3.2 Pattern: Authorization Pattern ............................................................................................ 23
      3.2.1 Scope of the PoC with respect the Reference Architecture ........................................... 24
      3.2.2 Evaluation methodology ............................................................................................... 28
      3.2.3 Evaluation results .......................................................................................................... 30
      3.2.4 Conclusion ...................................................................................................................... 33
  3.3 Report of Assessment of Requirements (Objectives) of Scenario (S19) .............................. 35
      3.3.1 Link between requirements and quality attributes ....................................................... 39
      3.3.2 Evaluation of requirements ............................................................................................ 39
      3.3.3 Conclusion ..................................................................................................................... 41

4 ASSESSMENT OF PoC10: SERVICE DISCOVERY - ATOS .................................................. 42

  4.1 PoC’s selection ..................................................................................................................... 42
  4.2 Patterns: Service Discovery and Service Matchmaking and Ranking ................................. 42
      4.2.1 Scope of the PoC with respect the Reference Architecture ........................................... 44
      4.2.2 Evaluation methodology ............................................................................................... 44
5 Assessment of PoC11: Supporting Service Matchmaking and Ranking Through Runtime Services Monitoring - CINI .............................................................. 57

5.1 PoC’s selection ............................................................................ 57
5.2 Pattern: Monitoring in Enterprise SOA ...................................... 57
   5.2.1 Scope of the PoC with respect to the Reference Architecture ... 58
   5.2.2 Evaluation methodology ......................................................... 59
   5.2.3 Evaluation results ................................................................. 61
5.3 Pattern: Service Matchmaking and Ranking ................................ 64
   5.3.1 Scope of the PoC with respect to the Reference Architecture ... 64
   5.3.2 Evaluation methodology ......................................................... 65
   5.3.3 Quality attributes provided to WP7 for this pattern ................. 66
   5.3.4 Evaluating quality attributes by using Pattern Quality Attributes Tree ... 67
   5.3.5 Evaluation results ................................................................. 68
5.4 Report of Assessment of Requirements (Objectives) of Scenario ... 70
   5.4.1 Link between the PoC scenario (scenario S1) and quality attributes ... 70
   5.4.2 Evaluation of requirements ..................................................... 71

6 Assessment of PoC12: Investigation of Patterns Composability Based on the Patterns for Service Discovery - TIE .............................................................. 73

6.1 PoC’s selection ............................................................................ 73
6.2 Patterns for Discovery concern: composition and quality characteristics .... 73
   6.2.1 Scope of the PoC with respect to the Reference Architecture ... 74
   6.2.2 Evaluation methodology ......................................................... 77
   6.2.3 Evaluation results ................................................................. 79
   6.2.4 Conclusion ........................................................................... 86
6.3 Report of Assessment of Requirements (Objectives) of Scenario ...... 87
   6.3.1 Link between requirements and quality attributes ................... 87
   6.3.2 Evaluation of requirements ..................................................... 88
   6.3.3 Conclusion ........................................................................... 88

7 Assessment of PoC13: Semi-automatic Service Composition at Design Time - ATOS 89

7.1 PoC’s selection ............................................................................ 89
7.2 Patterns: Assisted Composition Designer and Semantic Annotation Composition ... 89
7.2.1 Scope of the PoC with respect the Reference Architecture..........................90
7.2.2 Evaluation methodology ........................................................................91
7.2.3 Evaluation results ..................................................................................92
7.2.4 Conclusion............................................................................................96
7.3 Report of Assessment of Requirements (Objectives) of Scenario ................96
  7.3.1 Link between requirements and quality attributes...............................97
  7.3.2 Evaluation of requirements ..................................................................97
  7.3.3 Conclusion............................................................................................99
8 Assessment of PoC14: Cloud migration enabled by OSGi and exemplified by databases - SIE.................................................................100
  8.1 PoC’s selection .......................................................................................100
  8.2 Pattern: Cloud migration enabled by OSGi - step one ..............................100
    8.2.1 Scope of the PoC with respect the Reference Architecture..................100
    8.2.2 Evaluation methodology ....................................................................103
    8.2.3 Evaluation results ..............................................................................105
    8.2.4 Conclusion..........................................................................................109
  8.3 Report of Assessment of Requirements (Objectives) of Scenario ..........110
    8.3.1 Link between requirements and quality attributes..............................110
    8.3.2 Evaluation of requirements ................................................................110
    8.3.3 Conclusion..........................................................................................112
9 Presentation of Results of the Evaluation ....................................................113
10 Conclusion ................................................................................................122
References ....................................................................................................123
Glossary .........................................................................................................124
Annexe 1 ......................................................................................................128
TABLE OF TABLES

Table 1: Tests for the No SQL Storage PoC ................................................................. 14
Table 2: No SQL Storage Pattern Trade-off ................................................................. 14
Table 3: Evaluation of requirements of NoSQL Storage PoC8 ....................................... 22
Table 4: Components of the scope of the PERMIS RBAC PoC .................................... 25
Table 5: Authorization Pattern Trade-off .................................................................. 29
Table 6: Link between requirement and quality attributes ......................................... 39
Table 7: Evaluation of requirements of PERMIS RBAC Storage PoC9 ....................... 40
Table 8: Service Discovery Pattern Trade-off ............................................................. 45
Table 9: Service Matchmaking and Ranking Pattern Trade-off .................................. 45
Table 10: Results for the response time metric of Service Discovery ......................... 48
Table 11: Evaluation of requirements of Service Discovery PoC10 ............................. 52
Table 12: Monitoring in Enterprise SOA Pattern Trade-off ....................................... 59
Table 13: Matchmaking and Ranking pattern Trade-off ............................................. 66
Table 14: Evaluation of requirements of supporting service matchmaking and ranking through runtime services monitoring PoC11 .......................................................... 72
Table 15: Composition of Service Discovery Pattern Trade-off .................................. 78
Table 16: Technical details of a test environment ....................................................... 79
Table 17: Guideline of template-based discovery pattern ......................................... 80
Table 18: Guideline of the conclusion drawn from the test ...................................... 84
Table 19: Guidelines of the conclusions of interoperability and composability of patterns .......................................................... 85
Table 20: Link between requirement and quality attributes ..................................... 88
Table 21: Evaluation of requirements of investigation of patterns composability based on the patterns for service discovery PoC12 ......................................................... 88
Table 22: Assisted Composition Designer Pattern Trade-off ...................................... 92
Table 23: Semantic Annotation Composition Pattern Trade-off ................................ 92
Table 24: Tables of the time spanned by the experiments ......................................... 94
Table 25: Evaluation of requirements of semi-automatic service composition at design time PoC13 .......................................................... 97
Table 26: Components of the scope of Pattern Cloud migration enabled by OSGi ..... 101
Table 27: Cloud migration enabled by OSGi Pattern Trade-off .................................. 104
Table 28: Link between requirement and quality attributes ..................................... 110
Table 29: Evaluation of requirements of cloud migration enabled by OSGi and exemplified by databases PoC14 ........................................................................... 111
Table 30: Evaluation of requirements of cloud migration enabled by OSGi and exemplified by databases PoC14 (next) .............................................................. 111
Table 31: How to read the table of results ................................................................ 114
Table 32: Results of evaluation with ATAM approach or other approaches .............. 116
Table 33: Example of the Performance for the Not Only SQL pattern (MoMa) ............ 117
Table 34: Example of the Privacy of Authorization pattern (Thales) ............................ 118
Table 35: Example of the Performance for Monitoring in Enterprise SOA pattern (CINI) .......................................................... 119
Table 36: Example of the determination of Sensitivity Points: Operational costs and migration costs (Siemens) ................................................................................. 121
Table of Figures

Figure 1: Scope of the No SQL Storage PoC .................................................................................. 13
Figure 2: Utility Tree .................................................................................................................. 15
Figure 3: Workloads of the YCSB (from [7]) ............................................................................. 15
Figure 4: Observed behaviour for the write operations ................................................................. 16
Figure 5: Observed behaviour for the read operations ................................................................. 17
Figure 6: Elastic Speed-up .......................................................................................................... 18
Figure 7: Impact of the replication factor on performance and availability ................................. 19
Figure 8 Authorization Components .......................................................................................... 24
Figure 9: Scope of the PoC PERMIS RBAC – Authorization Pattern ........................................... 25
Figure 10: Authorization sequence diagram – use of the set of components .................................. 28
Figure 11: Evaluating quality attributes of Authorization pattern with ATAM Approach ............ 29
Figure 12: Example of trade-off point of a cryptographic key ....................................................... 32
Figure 13: Example of sensitivity point of the confidence of TTP or of a cryptographic key ......... 33
Figure 14: Architecture of ICT Compliances .............................................................................. 35
Figure 15: Different steps of the loan origination process in the ICT Compliances ...................... 38
Figure 16: Assessment report of Authorization requirement ......................................................... 40
Figure 17 Service Discovery Components .................................................................................. 43
Figure 18 Architecture used in the PoC ....................................................................................... 43
Figure 19: Scope of the PoC – Service Discovery ....................................................................... 44
Figure 20: Measures taken for the utility tree ............................................................................. 46
Figure 21: Response time evolution ............................................................................................. 48
Figure 22: Ontology for functional aspects .................................................................................. 53
Figure 23: Ontology for inputs and outputs .................................................................................. 54
Figure 24: WSML query for discovering services ....................................................................... 54
Figure 25: Example of service description ................................................................................... 55
Figure 26: The monitoring process .............................................................................................. 57
Figure 27: Scope of the PoC with respect to the NEXOF-RA ....................................................... 58
Figure 28: Evaluating quality attributes using the ATAM approach ........................................... 60
Figure 29: Sensitivity point: Monitoring intensity impact on availability ........................................ 62
Figure 30: Sensitivity point: Monitoring intensity impact on performance ..................................... 63
Figure 31: Trade-off point: Monitoring intensity ......................................................................... 63
Figure 32: The matchmaking and ranking process ...................................................................... 64
Figure 33: Scope of the PoC with respect to the NEXOF-RA ........................................................ 65
Figure 34: Evaluating quality attributes using the ATAM approach ........................................... 68
Figure 35: Sensitivity point: Memory amount and maximum scalability .................................... 69
Figure 36: Sensitivity point: Monitoring intensity impact on performance ..................................... 70
Figure 37: Scope of the PoC with relation to the RA ................................................................. 75
Figure 38: Component diagram of the PoC implementation ......................................................... 76
Figure 39: Web service description languages ............................................................................ 77
Figure 40: Valuating quality attributes with ATAM Approach .................................................... 78
Figure 41: complexity of a query vs. Complexity of an interface ................................................. 80
Figure 42: PR-Graphs for discovery of exact matching services .................................................. 81
Figure 43: PR-Graphs for discovery of similar services ............................................................... 82
Figure 44: Sensitivity Point: growing number of requests being processed by the system at the same time causes a crash ................................................................. 83
Figure 45: Sensitivity Point: growing number of requests being processed by the system at the same time causes a crash ................................................................. 83
Figure 46: Template-based discovery, Multi-phase service discovery and Matchmaking patterns in one system ................................................................. 84
Figure 47: Trade-off point: performance vs. complexity ................................................................. 85
Figure 48: E-commerce scenario ................................................................................................. 87
Figure 49: Semantic Annotation Composition Components ........................................................ 90
Figure 50: Scope of the PoC – Semi-automatic Service Composition at Design Time .................. 90
Figure 51: Measures taken for the utility tree ............................................................................ 93
Figure 52: Crisis management PoC ............................................................................................ 95
Figure 52: Initial workflow for the composition process .............................................................. 97
Figure 52: Final workflow obtained ........................................................................................... 98
Figure 53: Scope of the PoC – Pattern Cloud migration enabled by OSGi - step one ..................... 101
Figure 54 Structure diagram showing the PoC implementation .................................................. 103
Figure 55: Evaluating quality attributes with ATAM Approach ............................................... 105
Figure 56: Sensitivity point: Operational costs ........................................................................ 106
Figure 57: Operational Costs dependent on Hosting ................................................................. 107
Figure 58: Sensitivity Point: Migration costs ............................................................................ 107
Figure 59: Sensitivity Point: Migration costs due to technology learning ................................... 108
Figure 60: Trade-off point ......................................................................................................... 108
Figure 61: Assessment report of Migration requirement ............................................................. 112
Figure 62: Example of using the ATAM approach .................................................................... 128
1 INTRODUCTION

1.1. Purpose and scope

The purpose and scope of this deliverable is to report on the PoCs validation of each of the architectural choices and/or patterns demonstrated through each of the Proof-of-Concepts selected and set-up for Year 2.

This deliverable directly follows D8.1c “Proof-of-concept released” where each of the PoCs Phase II/Year II have been presented and released as software. Whereas D8.1c was focusing on Phase II PoCs’ content description, D8.2 is here focusing on reporting PoCs’ results in a way it could be exploited by the architecture team of NEXOF-RA (led by CA and WP7 lead) to further shape Reference Architecture specifications through integration of results (focus on Architectural choices and/or patterns) duly validated.

As such this document report on assessment of each of the architectural patterns/choices conveyed by the PoCs phase II which were selected with respect to quality attributes, scenarios those patterns claim and following in this a recommendation coming from the NEXOF-RA Reference Architecture team itself (WP10, WP7 & CA). [These quality attributes being specified in the section [Non-Functional Qualities] of the architectural pattern description promoted by WP7 and used by WP8 on other WPs as input. Patterns have been demonstrated and assessed in the context of scenarios derived from the work performed by WP10.

These validation results are for each architectural pattern based on three main steps: evaluation methodology, evaluation results and conclusion. First, an evaluation methodology is proposed for each pattern (here ATAM is promoted since relevant). Second, the sensitivity or the trade-off points are determined. Last, a conclusion of the validation results is reported.

1.2. Document overview

This deliverable is structured as follows: Section 1 is the document introduction, Section 2 to Section 5 reports on the findings regarding the assessment of each of the PoC set-up during PoC Phase II with a clear focus on assessing the architectural patterns they convey with respect to the claimed quality attributes (as expected by WP7). As such Section 2 reports on the findings regarding the assessment of the NoSQL Storage PoC on Scalability, availability and performance ACPs, Section 3 reports on the findings regarding the assessment of the Access Control PoC ACPs, Section 4 reports on the findings regarding the assessment of the PoC on Service Discovery ACPs. Section 5 reports on the findings regarding the assessment of the PoC on Supporting Service Matchmaking and Ranking Through Runtime Services Monitoring ACPs. Section 6 reports on the findings regarding the assessment of the Investigation Of Patterns Composability Based On The Patterns For Service Discovery ACPs. Section 7 reports on the findings regarding the assessment of the Semi-Automatic Service Composition At Design Time ACPs. Section 8 reports on the findings regarding the assessment of the Cloud Migration Enabled By OSGi and Exemplified By Databases ACPs. Section 9 presents all results under a table. Last, Section 10 concludes this deliverable.
2 ASSSESSMENT OF PoC8: NoSQL STORAGE - MoMa

This section reports the validation results of the No SQL storage PoC. After a short recap of the motivation leading to selection of this PoC (done in subsection 2.1), the rest of this section presents the validation results of the non functional qualities of the pattern (subsection 2.2) and of the functional part of the pattern (subsection 2.3)

2.1 PoC’s selection

The process leading to the selection of this PoC is described in D8.1.c. In summary, this PoC validates a pattern to attain high availability and scalability in cloud systems that, among the other things, can offer a solution to an open challenge that many enterprises will face in the coming years: the foresaw increase of digital data¹ that should be stored and managed in the future internet.

Scalability, availability and performance are relevant requirements and common to the majority of scenarios analysed in the project and to several emerging scenarios such as the ones relating to social applications, on-line gaming, and other.

In the cloud domain the NEXOF-RA project presents some alternatives leveraging on replication as mechanism to attain those features. Among these alternatives, the validation team has decided to set up and execute a PoC validating the No SQL storage patterns for the following reasons:

- it represents an emerging trend in cloud computing fostering data storages systems based on non-relational models. This trend is currently gaining momentum mainly due to its use in the system infrastructures of important companies such as Google, Yahoo, Facebook, Twitter etc. and after the recent announcement of Digg² and other well-known web sites to move towards this model

- the results of this PoC and the ones available of first phase PoCs validating the non-functional patterns for database replication in multi-tier systems can be used by the validation team and/or architects to argue about suitability of business cases and/or applications to the different approaches (e.g. non-relational VS relational)

- Yahoo Research Group (http://research.yahoo.com/) has recently benchmarked some systems based on No SQL approaches. The documented results can be used as a reference to understand if the PoC implements a correct behaviour.

2.2 Pattern: Not only SQL (No SQL) Storage

The pattern validated in the PoC is the No SQL Storage. It is described in the D7.5.b.

Basically, it presents the structure of a new trend in storage system based on distributed hash tables that allows providing elastic storage to many web applications deployed in cloud environments.

¹ A recent IDC outlook estimates an explosion of digital data over internet in the next ten years, see http://www.emc.com/collateral/demos/microsites/idc-digital-universe/iview.htm
² http://about.digg.com/node/564, march 2010
Up to now, relational databases have been the standard components used for storing persistent data. However, current relational databases present scalability problems when deployed in cloud environments. Therefore, new storage components have arisen to match the requirements of cloud applications.

These components have been categorized under the name of Not only SQL (NoSQL) systems. The main features of these components are a schema-free internal structure, data partition and distribution and data replication, what allows achieving high availability and horizontal scalability.

2.2.1 Scope of the PoC with respect the Reference Architecture

The following picture maps the PoC with the NEXOF-RA structure:

![Diagram showing the scope of the NoSQL Storage PoC](image)

**Figure 1: Scope of the No SQL Storage PoC**

In summary, the PoC relates to the following parts of the NEXOF-RA:

- Top level pattern: PaaS
- Abstract design pattern: Not only SQL (NoSQL) Storage
- Abstract Component: No SQL Storage
- Concrete component: Azure Storage tables
- Reference model: Service concern
- Guidelines and principle: the principle for development of specification (i.e. D7.2) and the principle for patterns validation via PoC (e.g. D8.0.c)

Strictly speaking no standard is adopted in the PoC, but it uses API to access the database.

2.2.2 Evaluation methodology

The methodology is based on the combination of scenario generation techniques to derive assessment criteria and metrics, and execution of tests to evaluate the systems behaviour against the criteria.

The tests executed for this PoC are devoted to evaluate Performance and Scalability and they have been executed using the Yahoo! Cloud Serving Benchmark (YCBS, [7]). A specific interface has been developed to allow execution of tests against the PoC.
The main focus has been on evaluating the impact and cost of the Replication for performance (focusing on the latency of requests when the system is under load) and scaling (focusing on the ability to scale elastically).

The following table presents the tests executed.

<table>
<thead>
<tr>
<th>TestID</th>
<th>Objective</th>
<th>Description of the technique</th>
<th>Expected Result/Behaviour</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Performance</td>
<td>Throughput is increased until the system reaches saturation and latency is measured. The test is executed for different configurations and different kinds of workloads consisting of different mix of read and update/insert operations</td>
<td>For a fixed configuration, as the throughput increases, it is expected an increase of the latency to a single request.</td>
</tr>
<tr>
<td>2</td>
<td>Scalability – Scale out</td>
<td>A number of servers are loaded with data and a workload is executed. Latency is measured. The same tests are executed when additional servers are added with a larger amount of data.</td>
<td>The latency is a measure of the perceived quality of service for the client. Response time should be constant as the number of servers, amount of data and throughput increases proportionally</td>
</tr>
<tr>
<td>3</td>
<td>Scalability – Elastic speed-up</td>
<td>A number of servers are loaded with data and a workload is executed. When the workload is running, new servers are dynamically (e.g. on demand) added and loaded. Operations are distributed also to the new servers and performance is observed.</td>
<td>Good elastic property should allow performance improvement when new servers are added but the impact of replication has to be evaluated. A period of disruption is expected until the system is reconfiguring</td>
</tr>
</tbody>
</table>

Table 1: Tests for the No SQL Storage PoC

2.2.2.1 Quality attributes provided to WP7 for this pattern

The following table presents the quality attributes specified by the producers of the patterns. Additional information is available in the D7.5.

<table>
<thead>
<tr>
<th>Quality attribute</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td>+</td>
</tr>
<tr>
<td>Performance (Efficiency)</td>
<td>+</td>
</tr>
<tr>
<td>Scalability</td>
<td>+</td>
</tr>
<tr>
<td>Consistency</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2: No SQL Storage Pattern Trade-off

Basically, the pattern claims to positively affect the availability, scalability and performance while negatively affect the consistency. It is worth mentioning that in some cases the so-called eventual consistency [8], a special form of weak consistency, can be assured.

2.2.2.2 Evaluating quality attributes by using Pattern Quality Attributes Tree

As reported in the D8.1.c, from analysis of the scenarios, the following quality attributes utility tree can be derived.
In order to evaluate some of the quality attributes of the pattern, we have performed several tests using the benchmark YCSB and a specific interface for our system.

This benchmark generates different types of workloads to execute performance and scalability tests. The following picture, taken from [7], presents the workloads of the benchmark:

<table>
<thead>
<tr>
<th>Workload</th>
<th>Operations</th>
<th>Record selection</th>
<th>Application example</th>
</tr>
</thead>
<tbody>
<tr>
<td>A—Update heavy</td>
<td>Read: 50% Update: 50%</td>
<td>Zipfian</td>
<td>Session store recording recent actions in a user session</td>
</tr>
<tr>
<td>B—Read heavy</td>
<td>Read: 95% Update: 5%</td>
<td>Zipfian</td>
<td>Photo tagging; add a tag is an update, but most operations are to read tags</td>
</tr>
<tr>
<td>C—Read only</td>
<td>Read: 100%</td>
<td>Zipfian</td>
<td>User profile cache, where profiles are constructed elsewhere (e.g., Hadoop)</td>
</tr>
<tr>
<td>D—Read latest</td>
<td>Read: 95% Insert: 5%</td>
<td>Latest</td>
<td>User status updates; people want to read the latest statuses</td>
</tr>
<tr>
<td>E—Short ranges</td>
<td>Scan: 95% Insert: 5%</td>
<td>Zipfian/Uniform*</td>
<td>Threaded conversations, where each scan is for the posts in a given thread (assumed to be clustered by thread id)</td>
</tr>
</tbody>
</table>

The workloads are representative of traditional applications using cloud data storage systems.

We have limited our tests to the execution of workloads B and C that are representatives of common operations in the collaborative learning scenario. In particular, workload B can emulate the operation, common in collaborative learning activities, of document (e.g. learning resources) tagging by learners, and workload C simulates user profile caching.

2.2.3 Evaluation results

For all the tests executed the experimental set up consists of four servers running Windows 2008 Sever. 3 servers are Xw800 dual Xeon 3.06 GHz, 2.5 GB RAM, disk 73.4 GB U_320 and have been used for replication nodes. A separate server has been used to run the client for the tests, i.e. benchmark and interface.
2.2.3.1 Evaluation results on the performance

The evaluation of the performance quality attribute has been performed via the execution of the two workloads above mentioned to measure latency as the offered throughput (# of operations / second) increases for different configurations.

The following pictures present Latency vs Throughput curves resulting from the execution of the testID number 1 for different replication factors R, where R=0 means a single server without replication and is considered as a base for comparison, R=2 means two replicated servers, and R=3 means three replicated servers.

In this test, for each of the three configuration (i.e. R=0, R=2, R=3), the offered throughput is increased until the measured throughput stopped increasing. From that point, additional load (in terms of operations / sec) only resulted in an increase of average latency.

The Figure 4 refers to the execution of a workload consisting of a 95% read and 5% of Insert or Updates, and shows the average latency of the write operations. The throughput is the total operations per second, including read and writes. The objective has been to understand the impact of replication on the performance of the system, when the replication factor is increased.

![Average Latency vs Throughput](image)

**Figure 4 : Observed behaviour for the write operations**

The observed behaviour shows that the average latency increases as the replication factor increases. A relevant increase in average latency can be observed passing from a non replicated configuration to a replicated one. Next, additional replicas (e.g. from R=2 to R=3) only provides a little overhead. The motivation is mainly due to two reasons: i) the overhead of the replication mechanism does not impact the R=0 configuration, and ii) the fact that in the PoC implementation the insert operation on a node waits until at least one replica answers. This allows at least one replica is consistent.
The Figure 5 refers to the execution of a workload read-only, showing the average latency of the read operations. Also in this case, the purpose has understood the impact of the replication on read operations, when the operations can be spread among more replicas.

![Figure 5: Observed behaviour for the read operations](image)

The observed behaviour shows an increase in terms of performance when the replication factor increases, due to the fact that read operation can be distributed. A little increase can be perceived also before the baseline curve (R=0) reach saturation, mainly for a replication factor of 3.

### 2.2.3.2 Evaluation results on Scalability

Regarding the scale out capacity, from Figure 5 it is possible to observe that the average latency remains quite constant as the throughput increase at least until each configuration reach its peak throughput.

It is possible to derive also the scale – out measured as the how many times a particular number of replicas multiply the max throughput supported of a non-replicated setup (R=0). For the workload read-only, the actual max throughput without observing a deep negative impact on performance is around 7,81 op / sec for non replicated set-up, around 14,52 op / sec for R=2, and around 18,80 op / sec for R=3, leading to a scale-out factor of 1,80 for 2 servers and a scale-out factor of 2,40 for three servers. Similar scale out factors can be derived if we consider the peak throughputs of the workload 95% read and 5% writes.

For the elastic speed-up capacity, resulting from the execution of the testID number 3, we have followed the approach proposed in [7] but, also in this case, with the objective of understanding the eventual impact of replication. The workload 95% read and 5% write has been executed with 2 servers. The offered throughput has been set to the 80% of the measured throughput with 3 servers and, at certain time (~ 230 sec.), an additional server
has been added. When the new server has been added, we started to load it and distribute operations also on the new server.

The results are shown in the next figure for the read and write operations. The read operations do not suffer of the replication overhead, while the write operations presents this overhead.

![Diagram showing latency for read and write operations]

**Figure 6: Elastic Speed-up**

The figures shows that a wide variance of the latency is observable for the read operations, when adding the new server and loading it, before observing a performance improvement due to availability of a new node.

For the write operations, it is possible to observe the impact of replication. When a new node is added, a variance of the latency can be observed but the system does not achieve better latency with respect to the previous configuration, and it stabilises to the value measured for this configuration in the Performance tests.

On the elastic speed-up, as written, the purpose has been to only to understand if implementing a replication approach can lead to negative elastic properties, and we have observed that it is not the case since the system is able to stabilise even with the replication.

As general considerations, from our experience, data repartitioning and self-healing mechanism should be first class operations to provide good elastic features, and additional tests are required to better understand this behaviour.

### 2.2.3.3 Consideration on Availability

For the availability quality attribute, tests have not been executed. Anyway, availability is directly related to the number of tolerated failures and to the time to recover from a failure. Increasing the replication factor will increase the number of tolerated failure.

The real objective is, anyway, the minimisation of mean time to recovery from a failure that would allow keeping at the minimum the number of replicas, minimising thus the overhead of the replication.

### 2.2.3.4 Sensitivity and Trade-off

From the tests results, the validation team derives the following considerations about sensitivity and trade-off points for replication:

- **Sensitivity 1: The perceived latency is sensitive to replication.** Replication has an overhead in particular in write operations. In particular, extra work to replicate data (mainly in the approach we adopted that waits until a certain number of replicas...
answers before concluding the write operation) hurts performance. By the other side, replicas allow spreading operations among more nodes. This aspect, in our observations, positively impacts in case of read operations.

- **Trade-off 1**: *Changing the replication factor may have impact on performance and availability.* Our observations shows that the challenge for an architect is to fix a suitable latency for a class of applications and understand the better configuration (e.g. number of replicas, number of servers) to achieve a specific throughput. But a trade-off exists with availability that is directly related to the number of replicas: a replicated system may support a number of failures equal to the number of replicas minus one.

The following picture shows the trade-off that can be derived: once defined a range of suitable latency for the class of applications and a range of availability (as % of uptime for a specific application/service), a proper replication factor and number of servers have to be identified as architectural decision.

![Diagram showing trade-off between performance and availability](image)

**Figure 7: Impact of the replication factor on performance and availability**

On the elastic speed-up the main result that can be observed from our tests, is that replication does not hurt elastic behaviour of a system developed according the No SQL pattern. As written, additional tests would be required to better evaluate the impact of recovery and self-healing mechanisms on the elastic speed-up and availability. It is, in fact, expected a positive impact since self-healing mechanism can reduce either the time required to stabilise a system when adding a new node and the time to recovery from a failure. This may add, anyway, additional overhead to a replicated system leading to another trade-off among elasticity, availability and performance.

### 2.2.4 Conclusion

The main objective for this PoC has been to understand the behaviour of a system based on the No SQL Storage pattern and, in particular, to understand the impact of replication on performance and scalability. The measured values of latency and throughput are not comparable with results achieved by benchmarking systems such as Cassandra, HBASE, PNUTS presented in [7] since these systems are optimised from several perspectives (e.g. I/O, caching, partitioning strategies, etc).

Comparison from a quantitative point of view the PoC with other systems was outside the scope of our tests that have had also a different focus (i.e. evaluation of the impact of the
replication) with respect to the ones reported in [7]. The behaviour observed in the tests is anyway similar to other No SQL storage systems benchmarked, showing the correctness of NEXOF-RA No SQL storage pattern decisions.

From a technical perspective, the validation team observed that the system shows scalability features but in terms of performance the replication has a cost on the average latency for application requiring write operations on a database. This is also due, as written, to the fact that the PoC implements a replication process where the Partition Selector performs replication just after receiving the store message, and waits until at least one replica answers. In this way, all the replicas receive the data and perform the required actions to store data, but the Partition Selector waits until at least a certain number of replicas answers. This includes overhead in terms of latency but support consistency among a set of replicas.

Anyway, as general trend, replication brings an overhead that can impact the latency for operation accessing databases. The challenge is thus to define a suitable latency for applications to be offered and set-up the infrastructure (\# of servers and replication factors) for the throughput to be offered while preserving the latency.

Besides scalability, to implement an elastic behavior also reliability is required. To this purpose architects must pay particular attention in implementing, as first-class mechanisms, recovery and self-healing mechanisms.

From a business perspective, there are evidences that No SQL storage systems are adopted by leading ICT players such as Google, Amazon, Facebook.

No SQL Storage systems are competitive for companies with unpredictable storage demands and, besides this aspect, business benefits coming from replication are becoming clear for service providers that, leveraging on replication, can offer Service Level Agreement indicating the 99.9% of availability of their service offers\(^3\).

In general, if the application presents requirements relating to data set growing quickly and doesn't present strict consistency need, these solutions are suitable since allows scalability and fault tolerance. As the Brewer’s CAP theorem evidences [9], relaxing consistency, it is possible to improve availability and partition tolerance, and this is basically what the majority of No SQL storage systems do.

Even if a weak form of consistency can be assured, if an application presents business requirements such as consistency, enforcing of data rules, complex reporting, and needs to attain availability and scalability, the patterns validated in first phase activities are preferable since they are designed for relational databases.

The challenge is to understand application needs and choose the most appropriate system and/or a combination of systems. This includes also the possibility to combine in-house data management systems (e.g. to support analytical or transaction processing) with cloud storage offered as a service for persistent storage and data backup, allowing thus companies to reduce the total cost of ownership of their infrastructures.

\(^3\) It is the case, for instance, of Google that exploiting replication (http://www.google.com/apps/intl/en/business/details.html) can offer a SLA with guarantee of 99.9% of availability of the Google Apps Services.
2.3 Report of Assessment of Requirements (Objectives) of Scenario

Performance, availability and scalability are non functional requirements common to several scenarios. As previously reported, these requirements have been evaluated via the execution of workloads representative of common operations in collaborative learning scenario (S5), Software as a Service CRM scenario (S21), as well as other scenarios of D10.1. In particular, the following requirements from D10.1 are related to these scenarios:

- Scalability (L1.8), that (as evidenced in the scenario S21) shall contain the element of elasticity;
- Service availability (L2.4),
- Scalable performance and throughput (L6.3)

This PoC provides a partial implementation of a storage system, and offers only a part (i.e. creation of databases, read and write operations) of the functionalities that are necessary to implement the requirements of the scenarios presented in D10.1. It is thus not possible a complete evaluation of the scenarios’ requirements but it is possible to associate to requirements some objectives and define check points measurable with this PoC, i.e. with the functionalities offered by this PoC, allowing at least a partial assessment of the requirements.

It is the case for instance of the requirement “R11 Services integration by semantic mash-up” that relates to S5. Semantic mash-up, in the context of the collaborative learning scenarios, has to support the identification of proper services / contents to integrate for a collaborative environment session. Thus a semantic mash-up server should be able to understand the “preferences” of a learner and match those preferences with the metadata of available services and contents. To address this requirement, mash-up servers have to access databases, typically via API, from different location. Availability is required to allow 24/7 operations and offers the possibility to spread data access operations among replicas.

Another case is the requirement “R29 Distributed Architecture”, relating to S21, that requires capability offered by this PoC to support creation and elastic addition of new nodes.

The two above presented requirements, as many others, basically need the capability to create a database resource and access it. Besides, the capability to add and remove resources on demand it is a useful objective to achieve for several requirements.

In the following we basically focus on an evaluation of the functionalities provided by this PoC to create and access a replicated database.

2.3.1 Link between requirements and quality attributes

As reported in D10.1, besides the collaborative learning and SaaS CRM scenarios, there are several scenarios linked to the one or more of quality attributes evaluated in this PoC, i.e. scalability, availability and performance

2.3.2 Evaluation of requirements

To partially assess the requirements of the above mentioned scenarios, we identify the following check-points:

- For scalability (L1.8) the check points are the assessment criteria presented in the previous sections, i.e. scale out and elastic speed-up
For scalable performance and throughput (L6.3) the check points are the assessment criteria presented in the previous sections, i.e. latency and throughput.

For service availability (L2.4), the identified check points are creation of databases with replicas, data access, and data storage with replicas.

As previously written, these check points are representative of functionalities required also by other requirements such as R29 Distributed architecture and R11 Service Integration by semantic mash up.

<table>
<thead>
<tr>
<th>Requirements / Objective</th>
<th>Check point</th>
<th>How to verify the check point is reached?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scalability (L1.8)</td>
<td>Scale out</td>
<td>Via execution of tests, see section “Evaluation Results on Scalability”</td>
</tr>
<tr>
<td></td>
<td>Elastic Speed-up</td>
<td></td>
</tr>
<tr>
<td>Scalable performance and throughput (L6.3)</td>
<td>Latency</td>
<td>Via execution of tests, see section “Evaluation Results on Performance”</td>
</tr>
<tr>
<td></td>
<td>Throughput</td>
<td></td>
</tr>
<tr>
<td>Service Availability (L2.4)</td>
<td>Creation of a new (data) resource with replicas</td>
<td>Direct inspection of the infrastructure</td>
</tr>
<tr>
<td></td>
<td>Read data</td>
<td>Direct inspection</td>
</tr>
<tr>
<td></td>
<td>Insert data</td>
<td>Direct inspection of the replicas</td>
</tr>
</tbody>
</table>

Table 3: Evaluation of requirements of NoSQL Storage PoC8

2.3.3 Conclusion

In this section, an evaluation of the functionality provided by the PoC has been reported.

This evaluation completes the non-functional one presented in the previous sections.

The achievement of this PoC is a first step towards the fulfilling of those requirements relating to performance, availability, and scalability (including elastic features).

This PoC implements the basic mechanisms of the most currently adopted cloud storage systems and our tests have mainly shown the impact and cost of the adoption of replication mechanism.

As written, from a functional perspective, this PoC currently offers basic functionalities that are required in several project scenarios, and our evaluation has been focused on the capabilities offered to create and access a database with replicas. The functionalities evaluated are such to fulfil several requirements of the project, and considerations similar to the ones reported in the non-functional evaluation sections can be reported here.

We refer in particular to the importance of the trade-off among number of replicas and suitable latency for a particular class of applications. This trade-off has a direct impact also on the user requirements that relate to the quality attributes evaluated. Software architects willing to design such replicate and distributed storage systems should pay attention on this.
3 ASSESSMENT OF PoC9: PERMIS RBAC - THALES

This section reports the validation results of PoC called PERMIS Role Base Access Control (RBAC). This validation was performed in the context of a scenario adapted to the IoS (eBanking – loan origination process) originating from COMPAS project a NESSI Compliant project.

3.1 PoC’s selection

This PoC has been selected and setup based on a proposal coming from WP4 and took over by Thales based on work performed and achieved by WP4 in close cooperation with WP7 and WP10. The Authorization pattern is part of the Security in ESOA, Cloud and IoS patterns as reported in D7.x.

3.2 Pattern: Authorization Pattern

Authorization functionality should provide a feature to determine whether an entity can access to the given protected resource in order to execute an operation.

The authorization is a key enabler for E-SOA, Internet of Services and Cloud computing environments since it enables a simple entity, a service consumer or a service provider to capitalize on the infrastructure while controlling the access to the resource or pieces of information it connects. According to the context where it is used, this relates to privacy or confidentiality of the information. It always relies on identity management and previous authentication of the entity.
The components depicted here have already addressed the problem of E-SOA for what concerns Authorization. Similar issue do also exist in IoS and Cloud Computing and we can refer here to statement such as in National Institute of Standards and Technology (NIST) SP 800-37: “Security Authorization federal agencies and support contractors to facilitate Security authorization is the successful application and demonstrate compliance with the Federal of the risk management framework (RMF) process”. As for Cloud Computing environments the demand is also clear (e.g. how can we authorize a service provider to access resources and/or also request an access to its resources?). To resolve interoperability issues such as the one quoted before, the authorization model should be based on standards such as XACML Specification, PERMIS⁴.

### 3.2.1 Scope of the PoC with respect the Reference Architecture

The objective of the PERMIS RBAC PoC is to implement the pattern which is specified in WP7 (D7.5), have it addressing the (authorization) requirement as specified in WP10 (D10.1), and to assess it from the QA attributes this pattern claims. The figure hereafter gives an overview of the mapping of the PERMIS RBAC PoC components to each element/constituent of the NEXOF RA structure.

---

⁴ PERMIS (PriviEge and Role Management Infrastructure Standards) is a sophisticated policy-based authorisation system that implements a set of standards. It was developed after a European project (http://www.openpermis.org/). It implements an enhanced version of the U.S. National Institute of Standards and Technology (NIST) (http://www.nist.gov) standard Role-Based Access Control (RBAC) model and it is used in a number of other European projects such as TAS3 (http://www.tas3.eu/).
Before describing all elements, here is the relationship between abstract components and concrete components.

<table>
<thead>
<tr>
<th>Abstract component</th>
<th>Concrete component</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy Enforcement Point (PEP)</td>
<td>PEP (Web Server CXF and Alert Manager)</td>
</tr>
<tr>
<td>Accounting</td>
<td>Secure Audit</td>
</tr>
<tr>
<td>PDP</td>
<td>PERMIS PDP</td>
</tr>
<tr>
<td>Digital Signature</td>
<td>XML Signature of RAMPART</td>
</tr>
<tr>
<td>Key Management</td>
<td>Key Managers of RAMPART</td>
</tr>
<tr>
<td>Delegation</td>
<td>DIS (Delegation Issuing Service - PERMIS)</td>
</tr>
<tr>
<td>SOA Framework</td>
<td>OpenLDAP</td>
</tr>
<tr>
<td>N/A</td>
<td>AXIS 2</td>
</tr>
</tbody>
</table>

Table 4: Components of the scope of the PERMIS RBAC PoC

The achievement of the PoC has used other components which are necessary in the validation context but not requested for the pattern implementation.
The SOA Framework helps the development of this pattern.

AXIS 2 is used to simulate the consumer.

**Description of each element**

**WS-Security** (Web Services Security) is a communication protocol providing means for applying security to Web services.

**XACML v2** (OASIS eXtensible Access Control Markup Language) is a declarative access control policy language implemented in XML and a processing model. It is used in the PoC as a request/response protocol between the Policy Enforcement Point and the Policy Decision Point.

**SAML 2.0** (Security Assertion Markup Language) is a version of the SAML OASIS standard for exchanging authentication and authorization data between security domains. SAML 2.0 is an XML-based protocol that uses security tokens containing assertions to pass information about a principal (usually an end-user) between an identity provider and a web service. SAML 2.0 enables web-based authentication and authorization scenarios including single sign-on (SSO).

**SOAP** (Simple Object Access Protocol) is a protocol specification for exchanging structured information in the implementation of Web Services in computer networks. It relies on eXtensible Markup Language (XML) as its message format, and usually relies on other Application Layer protocols (most notably Remote Procedure Call (RPC) and HTTP) for message negotiation and transmission. SOAP can form the foundation layer of a web services protocol stack, providing a basic messaging framework upon which web services can be built.

**LDAP** (Lightweight Directory Access Protocol) is an application protocol for querying and modifying directory services running over TCP/IP. OpenLDAP is a software which supports the LDAP.

**PERMIS** (PriviLege and Role Management Infrastructure Standards) is a sophisticated policy-based authorisation system that implements a set of standards. It was developed after a European project (http://www.openpermis.org/). It implements an enhanced version of the U.S. National Institute of Standards and Technology (NIST) (http://www.nist.gov) standard Role-Based Access Control (RBAC) model and it is used in a number of other European projects such as TAS3 (http://www.tas3.eu/). PERMIS supports the distributed assignment of both roles and attributes to users by multiple distributed attribute authorities, unlike the NIST model which assumes the centralised assignment of roles to users. PERMIS provides a cryptographically secure privilege management infrastructure (PMI) using public key encryption technologies and X.509 Attribute certificates to maintain users' attributes. PERMIS does not provide any authentication mechanism, but leaves it up to the application to determine what to use. PERMIS's strength comes from its ability to be integrated into virtually any application and any authentication scheme like Shibboleth (Internet2), Kerberos, username/passwords, Grid proxy certificates and Public Key Infrastructure (PKI).

As a standard RBAC system, PERMIS main entities are an authorisation policy, a set of users, a set of administrators (attribute authorities) who assign roles/attributes to users, a set of resources that are to be protected, a set of actions on resources, a set of access control rules, and optional obligations and constraints. The PERMIS policy is eXtensible Markup Language (XML)-based and has rules for user-role assignments and role-privilege
assignments, the latter containing optional obligations that are returned to the application when a user is granted access to a resource. A PERMIS policy can be stored as either a simple text XML file, or as an attribute within a signed X.509 attribute certificate to provide integrity protection and tampering detection. User roles and attributes may be held in secure signed X.509 attributes certificates, and stored in Lightweight Directory Access Protocol (LDAP) directories or Web-based Distributed Authoring and Versioning (WebDAV) repositories, or they may be created on demand as Security Assertion Markup Language (SAML) attribute assertions.

The PERMIS authorisation engine encompasses two components: a Credential Validation Service that validates users' roles according to the user-role assignment rules, and the Policy Decision Point (PDP) that evaluates users' access requests according to the role-permission assignment rules (or access control rules). Access to a resource depends upon the roles/attributes assigned to the user, and the role-permission assignments, which can contain constraints based on the user's access request (e.g. "print less than 10 pages") and the environment (e.g. time of day). PERMIS can work in either push mode (the user attribute assignments are sent to PERMIS by the application) or in pull mode (PERMIS fetches the attribute assignments itself from LDAP/WebDAV repositories or SAML attribute authorities). PERMIS is an open source project and the Java source code can be downloaded from http://www.openpermis.org.

**X.509 v4** is an ITU-T standard for a public key infrastructure (PKI) for single sign-on (SSO) and Privilege Management Infrastructure (PMI). X.509 specifies, amongst other things, standard formats for public key certificates, certificate revocation lists, attribute certificates, and a certification path validation algorithm. It relies on cryptography.

RAMPART components are already described in the Security PoC Phase I of D8.1b and D8.2a.

**Secure Audit** or SAWS is a Secure Audit Service that can log any messages. It can run as either a standalone web service called SAWS (Secure Audit Web Service) or as a Java Secure Audit Trail Service (JSATS) that is called via a Java API. In either case, the log file that is created is cryptographically protected against tampering, which makes it a sound platform for storing all sorts of audit trails securely. SAWS/JSATS also has a Viewing Tool to let you check the integrity of saved log files, and to view their contents.

The abstract components are materialized together with the concrete components which participate to the implementation of the PoC.

The abstract design patterns are represented directly by the pattern.

This pattern is applicable to the Top-Level patterns: E-SOA, Cloud and IoS.

The implementation design patterns are represented with these three main activities (to build a complete chain, we need more details of flow exchange. And then we must integrate more actors on the chain):

- A user requests access to a protected resource
- PEP checks the security policy towards a PDP
- According to his/her role, a user is granted to access to the resource

The following sequence diagram which is already specified in the WP7 [Authorization Pattern] explains how an authorization is implemented. This diagram shows the use of the set of abstract components which are described in the previous paragraph.

Figure 10: Authorization sequence diagram – use of the set of components

3.2.2 Evaluation methodology
The pattern is evaluated according to quality attributes it claims. These quality attributes are stated in the filled-in template description of the architectural pattern (according to a shared and agreed template provided by WP7). The stated quality attributes in this architectural pattern description are evaluated using the ATAM approach. More specifically the quality attributes utility tree is used to show and analyze the results of the implementation of each quality attribute.

3.2.2.1 Quality attributes provided to WP7 for this pattern
The Authorization pattern was specified by WP4 and communicated to WP7. In the context of its specification, this pattern has been identified to support certain defined quality attributes. These quality attributes affect the quality of the architecture where each pattern is integrated. Evaluating these quality attributes allows to know if the resulting architecture where the pattern has been applied meets the expected requirements.

[The significations of plus (+), minus (-) or neutral (0) are given in the specification of this pattern for D7.5]
3.2.2.2 Evaluating quality attributes by using Pattern Quality Attributes Tree

The assessment criteria are partially based on the ATAM methodology [6], which has been adapted to our specific need that is to evaluate, quantify and compare the tradeoffs between architectural choices. The ATAM methodology was originally developed to assist architectural decisions by taking into account early in the design process the quality attributes.

<table>
<thead>
<tr>
<th>Privacy</th>
<th>+</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accountability</td>
<td>+</td>
</tr>
</tbody>
</table>

**Table 5: Authorization Pattern Trade-off**

![Diagram of Authorization Pattern Trade-off]

**Figure 11: Evaluating quality attributes of Authorization pattern with ATAM Approach**

Please report to the Annex 1 for the details of using ATAM approach. The following description is just a report of results of the analysis.

The resource protected by the Role Based Access Control (RBAC) and also PERMIS RBAC (an implementation of this pattern in the PoC Phase II) will be accessible only by those actors who will be granted access rights to them. For this reason, it shall increase the **Privacy** of the resource and the restriction of the accesses to it to only those who have to access it. In this case, the Privacy, used for the large description, can be restrained to the **Confidentiality**.

In cryptography, X.509 is an ITU-T standard for a public key infrastructure (PKI). X.509 specifies, amongst other things, standard formats for public key certificates, certificate revocation lists, attribute certificates, and a certification path validation algorithm. Today, the
recommended length for keys used by the format X509 certificates is 1024 bits. This key length meets the needs of applications such as e-commerce. Its robustness guarantees the integrity of the Role stored on this certificate. By this way, the Privacy is still increased.

The Accountability enforces imputability, guarantying the transparency of the operations. SAWS is a Secure Audit Service that can log any messages. It can run as either a standalone web service called SAWS (Secure Audit Web Service) or as a Java Secure Audit Trail Service (JSATS) that is called via a Java API. In either case, the log file that is created is cryptographically protected against tampering, which makes it a sound platform for storing all sorts of audit trails securely. SAWS/JSATS also has a Viewing Tool to let us check the integrity of saved log files, and to view their contents.

### 3.2.3 Evaluation results

The evaluation results rely on evaluations for which quantitative indicators are not applicable in this context/field where trust and confidence is at stake. Privacy evaluation is even subjective. Privacy criteria, since they rely on the context of usage, are not commonly defined yet. Beyond, privacy evaluation, which is by nature an end-user evaluation, often rely on the confidence the end-user will give to a trustable third party which is a provider of a privacy service. We must ask ourselves: how to measure the trust, which itself is subjective and is often in touch to personal experience or that of a third party to whom we trust.

The Authorization is a set of complex processes which guarantees the access to a resource for the right entity.

We can see it with the Digital Signature which guarantees the integrity. Digital Signature uses the certificate which is delivered by the Certification Authority. This signature proves who the author of the data is. This proof is based on the best practices recommended by the PERMIS for the Role based on Certificate.

This process of Authorization is supported by three main security activities:

- PERMIS RBAC (an implementation of authorization)
- Trusted Third Party
- Secure Audit using Cryptography

The cryptography uses the private key to encrypt the data. This encryption using this key is considered as trustworthy as the key has not been broken.

The quality of a key is its ability to resist to an attack or any practice of code breaking or cracking the code. While a key is not broken, it may be considered of sufficient quality.

To verify that a message has been signed by a user with his/her secret key and has not been modified by the receiver, we must only know the corresponding public key. The proof of the non-modification of a message guarantees the message Integrity.
In cryptography, a Trusted Third Party (TTP) is an entity which facilitates interactions between two parties who have trust in the third party. In TTP models, the relying parties use this trust to secure their own interactions.

In the Authorization process, a certification authority (CA) generates a digital identity certificate used by the Role based Certificate. The CA then becomes the Trusted-Third-Party in this certificates issuance. By using the Role Based Certificate, the authorization process offers the best protection for the Privacy.

We can have more or less confidence. This perception is subjective, proven or not proven by facts, but there is no unit of measurement to evaluate it. We can just position it on a scale of values that we ourselves graduated according to the idea that we do trust.

The proof of Secure Audit permits to guarantee the traceability that supports the Accountability quality attribute. The log made by the Secure Audit is also encrypted. As consequence, the Secure Audit guarantees both Accountability and Non-Repudiation. This is not the objective of this pattern but the Non-Repudiation complements the Accountability quality attribute.

**Computational cost and best practices (Performance)**

The asymmetric algorithms known thus far are relatively big resources consumers compared with most symmetric key algorithms. Therefore they are mainly used to protect secret keys or encryption of small amounts of data, such as a hash.

Although symmetric encryption is faster than asymmetric encryption, the last is more secure. The programmer chooses what to use and must consider both speed and security. Large amounts of data are difficult to encrypt by using asymmetric key encryption due to the performance overhead. One other major problem with this type of encryption is key management. In many organizations, a public key infrastructure is implemented and used for revoking, distributing, and managing certificates.

This study can be done with the real need of performance.

In this case, the balance with different parameters must be trade-off:

- key length in bits to use to encrypt the data,
- cryptographic algorithm,
- type of processor (math support),
- asymmetric, symmetric, key session.

In the security PoC, this study is not done because the costs of it are important according to the objectives.

**Trade-off point of the encryption algorithm**

We can take an example of comparison between two different algorithms. One time, we are confident on these keys. This example shows only how we can balance between these algorithms (asymmetric and symmetric). According to the security expertise, the Asymmetric
algorithm is slower with the big volume. This example is simplified and reduced to two algorithms. In certain cases, it’s more complex to find the trade-off. Ease of key distribution should be taken into account. All the time, the choice of a security algorithm is defined according to the security needs.

![Diagram of trade-off point of a cryptographic key]

**Figure 12: Example of trade-off point of a cryptographic key**

This example is a concrete determination of the trade-off of the encryption algorithm.

Regarding the cryptographic algorithm and the type of key, to achieve the privacy quality attribute, the choice made in the security PoC is clearly a RSA algorithm used in the Public Key Infrastructure used for the trusted third parties. The whole PERMIS technology is based on attribute certificates used to sign the association of the roles to the users by a trusted third party. In that technology, the size of the attributes to be signed is small, which make them perfectly adapted to an asymmetric signature done by means of a Public Key Infrastructure.

To achieve the accountability quality attribute, the PERMIS Secure Audit Web Service combines two types of functionalities:

- Integrity of the audit trail is guaranteed by means of symmetric algorithms for the Hash function. The supported algorithms are MD5, SHA-1, SHA-256, SHA-384 and SHA-512.
- Authentication of sender of the audit log is guaranteed by the signature of the Secure Audit Web Service which provides also non-repudiation. This is performed by means of the DSA algorithm.

**Determination of Sensitivity Points**

After evaluating subjectively the results, we ought to determine the sensitivity point of TTP (Trusted Third Party) or of Cryptographic Key.
The above picture shows that the level of confidence on the TTP or on the cryptographic key follows two linear phases. The first phase is when the TTP is considered as trustable. Once, we lost the trust on the TTP or on the cryptographic key for many reasons, then we enter phase 2 where the Loss of confidence point in the TTP is rapidly met.

For example, a cryptographic key could be breakable according to the means deployed and time spent to break it. To this moment, we can trust a cryptographic key for a given time-to-live period after which the key must be replaced. The moment of lost of confidence should determine the time at which we should stop using it. The above picture is valuable for the both quality attribute: integrity and accountability [Figure 24].

**Determination of trade-off points of TTP or Cryptographic key**

For this PoC, we recommend either a 1024-bits RSA key length for demonstrators or a 2048 bits RSA key length for application that will run several years. 1024-bits RSA keys are reputable to be vulnerable by brute force attack in the next years, since 2048-bits RSA keys should resist to brute force attacks until 2030 according to our current predictions of the computation speed evolution.

**3.2.4 Conclusion**

The PERMIS RBAC PoC has been proposed to answer requirements in the context of Internet of Services (IoS) and Cloud.

The ATAM approach by using the quality attributes pattern tree highlights clearly how the quality attributes have reached their target. It means that this approach supports the specification of the quality attributes inside the Authorization pattern.

The two quality attributes which have been demonstrated are the Privacy attribute and the Accountability attributes. As shown, privacy, tied to Authorization functionality, is one of the dimension for the adoption of the IoS pattern. Accountability seems to be one of the key
dimensions for the Cloud adoption. This shows how the PoC contributes efficiently to the top-level patterns.

The subjective approach permits at least to determine the sensitivity and trade-off points. As consequence of the evaluation based on the subjectivity, these points are only the approximate data.
3.3 Report of Assessment of Requirements (Objectives) of Scenario (S19)

Inspired from CobiT (Control Objectives for Information and related Technology), this is audit standards which are dedicated to the information systems.

After the results of WP10, a scenario allows extracting a certain number of requirements. A requirement can be assimilated to an objective to reach. In this case, controlling of objectives is done via the defined check points.

**S19: ICT Compliances scenario (domain: banking, process considered: Loan origination).** This scenario comes from COMPAS project and has been revisited in order to serve the demonstration and validation of Architectural pattern on Authorization in an IoS context. It can also be seen as resulting from a joint work between COMPAS (NCP) and NEXOF-RA (NSP).

As already reported in D8.1c, the following picture is the reminder of the functional architecture of the ICT Compliances scenario.

![Figure 14: Architecture of ICT Compliances](image)

This scenario is the best way to confront the pattern in terms of functional requirements to the real world.
The following description of the different steps of the scenario permits to identify what activities are concerns for the authorization.

**Description of each step of loan process**

**Step 1: Alice goes to the bank**

Bryan connects to the application “Credit- Salaried Young”, a software application for processing loan requests from young salaried professionals.

The above web service application is protected by proxy SSO LemonLDAP. The access is done via the proxy, by using the module Liberty Alliance of LemonLDAP for the authentication. The LemonLDAP portal, configured with the authentication Liberty Alliance module’s, receives the request of Alice. On this portal, two Identity Providers are referred - one is reserved to the bank clerks, the other is dedicated to the customers.

Bryan checks the Customer Information File to retrieve Alice’s data.

Bryan processes Alice’s confidential data.

Bryan’s feedback on Alice’s credentials is extremely positive at this stage. These comments are opportunely signed by Bryan with this certificate, stored in the LemonLDAP directory, and reported in both the Customer Information File and the process log. The loan origination process can move to the next phase.

**Step 2: The bank double checks the credit worthiness of Alice**

The access to the necessary information providing the level of assurance required is regulated by appropriate collaborative services. These services behave accordingly to precise security policies defined and enforced through an authorisation infrastructure that relies on pre-existing trust relationships between the data owner and the bank.

In selecting the post-processing clerk, the bank encounters the separation of duty requirement imposing a mutual exclusion constraint on the tasks of credit broker and post processing. In our scenario, this results in preventing Bryan to be involved in the post processing phase. David is chosen for this task and due to the fact that the amount of the loan enquired by Alice does not exceed 1 million Euros, James, David’s advisor is not required to intercede.

David proceeds in the post-processing phase by querying, in this order, the Credit Bureau and the internal rating application.

The internal scoring application assigns a low risk level to the Alice’s application and, once David has stored this information, the loan origination process move to the third phase. In the circumstance of a negative scoring result, the application would have enforced the completion of the request assessment to the David’s manager (James).

Assessing the loan risk using an internal information, the process must assume the control of internal interactions for what concerns the security aspects (e.g., the data exchanged must be kept confidential, authorization policies need to be considered for accessing to Alice private data, etc).
**Step 3:** The bank calculates the price for the bundled product (loan)

David has to choose the most appropriate bundled product for Alice in the database products. He queries the Pricing Engine service to compute a price for the bundled product (notice that this query does not need the real identity of the customer to be executed). The result in terms of, e.g., original price, customer segment special conditions, customer company special conditions, asset limit for price, is then returned to David and proposed to Alice.

In doing these operations the process ensures the provision and treatment of anonymous data on the side of the pricing engine service, protects the communication between the enquiring service used by David and the Pricing Engine service to preserve the integrity of the data, and provides appropriate log mechanisms to guarantee the transparency of the price calculation process.

**Step 4:** The bank and Alice sign the form

The contract is digitally signed using the respective secure signature creation device (SSCD) of Alice and Gerald. A copy of the contract is printed for Alice’s documentation. The process exploits an appropriate security protocol to enforce the non-repudiation of the signature. It should also ensure that an adequate certification authority and a time stamping system are used during the signature process.

The contract is signed; Gerald updates the bank information system with the remaining data and provides to Alice a formal document stating that the amount of the loan will be transferred on Alice’s bank account in one week.

The different steps of the loan origination process are depicted in Figure 1:
Figure 15: Different steps of the loan origination process in the ICT Compliances
This scenario allows defining a requirement named Access Control which is contained in the L5.1:

- Access Control contained in the L5.1

As reported in the requirement at the fit criterion section. The measurement is defined in different check points:

- Entity
- Resources
- Operation
- Audit

The control is done in order to verify if the three parameters are respected regarding to the defined policies (ACL for example).

3.3.1 Link between requirements and quality attributes

Just stated here the link between QAs and requirements (work done by WP10)

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Is linked to Quality Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Control</td>
<td>Privacy</td>
</tr>
<tr>
<td>Access Control</td>
<td>Accountability</td>
</tr>
</tbody>
</table>

Table 6: Link between requirement and quality attributes

3.3.2 Evaluation of requirements

The requirements are exposed in certain check points. These check points must be verified by the implementation.

According to the ICT Compliances scenario, we authorize only the access of Bryan (Credit Broker role) to the Credit Bureau in order to request for information about the payment incidents.
Table 7: Evaluation of requirements of PERMIS RBAC Storage PoC9

As defined in this table, the missing thing is to test the different possible cases to verify the system fail.

Here is the following picture which sums from this table.

Figure 16: Assessment report of Authorization requirement
3.3.3 Conclusion

The achievement of this pattern brings a high added value for the Access Control requirement of D10.1.

The assessment of functional requirements completes the previous evaluation based on quality attributes.

The assessment of requirements proposed by this document can’t be applied generally. It needs a context and defined activities.
4 ASSESSMENT OF PoC10: SERVICE DISCOVERY - ATOS

Atos assesses a PoC called Service Discovery that demonstrates and validates some patterns related to service discovery in an IoS domain mainly, by using a tool which exploits semantics, although it is applicable to E-SOA as well, as the main difference depends on one of the external components.

4.1 PoC’s selection

These PoC has been selected and setup based on a proposal coming from WP2 and took over by ATOS based on work performed and achieved by WP2 in close cooperation with WP7. According to WP6, Service Discovery is one of the NEXOF-RA main concerns and most of the scenarios require, at some stage, to develop a service discovery, in order to finish the implementation of a process implementation or to look for services at runtime.

The Service Discovery IT has identified several patterns related to the search of services, in order to support users and developers in looking for services in an effective, quick and easy way. The patterns evaluated in the PoC are the following:

- **Service Discovery**, as the high level pattern
- **Service Matchmaking and Ranking**, as a way to implement the Service Discovery pattern

There is another pattern involved as well: **Multi-phase Discovery**. In the case of this PoC, due to the tools used, we will consider that only one phase is carried out during discovery. But, as we consider it interesting, we compare the results obtained with the service discovery tools developed in INFRAWEBS, which are able to perform more than one phases for refining the search. For doing so, the data provided in [ref.] has been used.

4.2 Patterns: Service Discovery and Service Matchmaking and Ranking

Service Discovery is a key functionality which, in principle, could be done manually by users, using a list of available services. The problem is that, as the number of services increases (a lot) it is almost impossible to perform this action without the support of a tool which can filter the results, giving as result a short list of final candidates.

This functionality is necessary in any environment related to services, as E-SOA and Internet of Services, since in both domains there will be a lot of services which can be used and, in any case, users need to perform the best choice in line with their requirements. Even if they want only to integrate one service in an application or they want to create a complex service composition, users will need support for finding the services which best fits with what they need.

While Service Discovery defines, in an abstract way, the main components involved in the discovery, Service Matchmaking and Ranking provides a more concrete architecture representing the core of any discovery engine. This last pattern is also quite abstract, although its main components can be mapped directly to implementation solutions. In the case of the PoC, experiments are performed using a semantic based approach.

The components presented in this section can be used for both E-SOA and IoS domains, as they are abstract and allows different implementations (such as using semantics). They can be more effective in E-SOA, in concrete application domains, as better and richer ontologies...
can be defined (reducing the scope of the reasoning and not requiring mediation between ontologies), keywords can be limited and the registry is controlled by the organization centralizing the activity.

**Figure 17 Service Discovery Components**

These components represent the whole Service Discovery pattern, containing all the elements necessary. So the PoC aims at providing tools which can reach the implementation level of the main components, especially the SDEngine, which represents the core component for filtering services, by means of an EspecializedSDEngine provided by SOA4ALL tools.

**Figure 18 Architecture used in the PoC**
Those tools will implement the EspecializedSDEngine by using the architecture shown in Figure 16.

### 4.2.1 Scope of the PoC with respect the Reference Architecture

![Figure 19: Scope of the PoC – Service Discovery](image)

We have focused on the usage of SOA4ALL tools for evaluating the patterns, working in close cooperation with SOA4ALL members, as a way to evaluate their architecture as well. The main abstract component corresponds to both patterns under evaluation. The Service Discovery represents the high level solution for performing discovery, which is, basically, formed by a specialized service discovery engine based on semantics. Evaluation results coming from INFRAWEBS have also been used as reference, for comparing approaches, as it also works with semantics and can perform multi-phase discovery.

About the standards involved, as SOA4ALL is based in WSMO for describing services, so it is the main standard to be used in semantics. Moreover, SPARQL is highlighted as the language to send semantic.

### 4.2.2 Evaluation methodology

The patterns are evaluated according to quality attributes they claim (the same for both patterns). These quality attributes are stated in the filled-in template description of the architectural pattern (according to a shared and agreed template provided by WP7). The
stated quality attributes in this architectural pattern description are evaluated using the ATAM approach. Especially the quality attributes utility tree is used to show and analyze the results of the implementation of each quality attribute.

4.2.2.1 Quality attributes provided to WP7 for these patterns

The Service Discovery and the Service Matchmaking and Ranking patterns are specified with the work done on WP2. In the context of specification, these patterns have been identified to support certain defined quality attributes, although their main purpose is to support concrete functionalities for users. These quality attributes affect the quality of the architecture where each pattern is integrated. Evaluating these quality attributes permit to know the potential effects in the resulting architecture.

[The significations of plus (+), minus (-) or neutral (0) are given in the specification of this pattern for D7.5]

In this case, only attributes related to those evaluated in the PoC are listed:

<table>
<thead>
<tr>
<th>Service Discovery</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buildability</td>
</tr>
<tr>
<td>Modifiability</td>
</tr>
<tr>
<td>Interoperability</td>
</tr>
<tr>
<td>Reliability</td>
</tr>
<tr>
<td>Performance</td>
</tr>
</tbody>
</table>

Table 8: Service Discovery Pattern Trade-off

<table>
<thead>
<tr>
<th>Service Matchmaking and Ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buildability</td>
</tr>
<tr>
<td>Modifiability</td>
</tr>
<tr>
<td>Interoperability</td>
</tr>
<tr>
<td>Reliability</td>
</tr>
<tr>
<td>Performance</td>
</tr>
</tbody>
</table>

Table 9: Service Matchmaking and Ranking Pattern Trade-off

4.2.2.2 Evaluating quality attributes by using Pattern Quality Attributes Tree

The assessment criteria are partially based on the ATAM methodology [6], which has been adapted to our specific need that is to evaluate, quantify and compare the tradeoffs between architectural choices. The ATAM methodology was originally developed to assist architectural decisions by taking into account early in the design process the quality attributes.
We use the ATAM utility tree for matching the quality attributes (buildability, modifiability, interoperability, reliability and performance) with metrics to be used in the evaluation. The utility tree for the patterns is presented in [D8.1c p39 Figure 10: PoC 10 Assessment Criteria and metrics] and a detailed description may be found in the same deliverable. Other metrics have been used to evaluate the architecture in terms of functional requirements fulfilment and for determining whether the trade-offs are so important with respect to the solution itself. For instance, it is possible that performance is much more affected because of the computational requirements of the solution rather than because of a choice in the architecture.

In order to evaluate some of the quality attributes addressed by the patterns, we have performed several experiments with the SOA4ALL Discovery Tool, defining an ontology for the Crisis Management domain and creating several services descriptions which have been published in a repository.

4.2.3 Evaluation results

After carrying out the planned experiments, the metrics defined in the utility tree have been measured. They can be seen in Figure 18.

![Utility Tree](image)

**Figure 20: Measures taken for the utility tree**
We have evaluated the first prototype of service discovery engine provided by SOA4All, which is a rather simple discovery prototype, according to the number of developed services, but that relies on existing libraries and on the WSML2Reasoner service, which has been improved in SOA4All project as well.

SOA4All Service Discovery engine executes a matchmaking process that compares WSMOLite service descriptions (of WSDL-based services) stored within a repository hosted by the file system. This repository is loaded in the WSML2Reasoner engine that processes the WSML queries created by the Service Discovery engine out of the searching criteria posed by the final user. Current SOA4All Service Discovery engine supports a searching criteria consisting of:

- Functional classifications as subconcepts of WSMOLite FunctionalClassificationRoot
- Preconditions and effects as WSMOLite Conditions set upon inputs and outputs

Current prototype neither support WSMOLite NonFunctionalProperties (Requirements, Preferences) nor contextual sensibility. Those features will be implementing in next prototype.

It is possible to see the complexity of the solution. Even if it has not too many classes, the size of the tool is quite high, as it requires a lot of libraries for operating. Moreover, this amount can be increased if we take into account the amount of knowledge necessary for being able to work with the tool.

In the contrary, modifiability metrics show that it is quite easy to extend the tool in order to increase its functionality, as already planned.

In the case of INFRAWEBS tools [11], the size of the package is 22Mb for the main tools, without including the knowledge base necessary for operating with the tools. This tool provides multi-phase discovery, but there is no evidence that it provides concrete mechanisms for adding new algorithms, although it is possible to create new components which implement new approaches.

We have performed a set of experiments that executes a service discovery invocation over a repository of semantic service descriptions ranging from 100 to 1000 descriptions.

A typical WSMOLite description includes concepts such as service, operation, message and properties such as hasFunctionalClassification, hasOperation, hasInputMessage, hasOutputMessage, etc.

Search criteria consist of functional classification: HumanResourceManagement, MilitaryInformationGathering, and a precondition setting that the input should be of type MilitaryInformation.

The experiments performed invoke the service discovery engine linked to a WSML2Reasoner service that is connected to a repository of service descriptions which number changes after each experiment, from 100 to 1000 service descriptions. The Service Discovery engine returns the complete description for all those services matching the given criteria.

Results for the response time metric are given in next table.
Table 10: Results for the response time metric of Service Discovery

<table>
<thead>
<tr>
<th>N# Services</th>
<th>Time spent (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>11.3</td>
</tr>
<tr>
<td>200</td>
<td>24.9</td>
</tr>
<tr>
<td>300</td>
<td>25.22</td>
</tr>
<tr>
<td>400</td>
<td>25.45</td>
</tr>
<tr>
<td>500</td>
<td>26.03</td>
</tr>
<tr>
<td>600</td>
<td>25.6</td>
</tr>
<tr>
<td>700</td>
<td>29.2</td>
</tr>
<tr>
<td>800</td>
<td>30.5</td>
</tr>
<tr>
<td>900</td>
<td>34.4</td>
</tr>
<tr>
<td>1000</td>
<td>34.8</td>
</tr>
</tbody>
</table>

On the one hand, the experiment execution time is very sensitive to the number of matched services, since for each of them, service discovery has to issue WSML queries to the reasoned in order to retrieve the complete description. This is observed in the transition from 100 to 200 services, and from 600 to 700 where the number of matched services changed.

On the other hand, the figures obtained shows a slow increase of the execution time with the number of services within the repository, but with the relative small number of services within the repository we have not stressed the service discovery performance since indeed that stress test would not test the service discovery but the WSML2Reasoner. Indeed, we have corroborated that WSML2Reasoner requires a long time to load a repository of 10000 services when it is executed in a typical laptop, giving problems.

Those results should be taken then as qualitative, since increasing the volume of the repository stress the reasoned but not the service discovery.

This last statement about the reasoner is very important for reliability. In normal operation, the classes related only to service discovery do not fail. Even if the tool interoperates well
with the registry and with an external reasoner, any problem in one of these components is a big problem for the discovery tool, as it is not capable of generating any result. This means that the reliability is decreased because the pattern depends too much in external components. In the case of the tool used, it does not include mechanisms for recovering from such a failure, so it would be necessary to provide an approach for contacting alternative registries and reasoners. In the case of E-SOA, it is possible to control these external elements and recover the system, but in an IoS environment, where we have no control over external reasoners and registries, it is not possible that we are not able to recover the system.

Service discovery is accurate in the sense that returned candidate services match the searching criteria. Whether the candidate services satisfied the intended criteria or not depends on the quality of the search criteria, that is, the user ability to express her needs or requirements by selecting proper functional classifications, preconditions and effects. Furthermore, the accurate of the service discovery depends on a proper selection of the search criteria on the similar basis used by the service providers in order to provide accurate service descriptions.

In the case of INFRAWEBS [11], it is based on semantics as well, although it applies first a keyword based filtering, which can increase performance. In [11] there are some measures about discovery activities with the tools. In similar conditions to our experiments, INFRAWEBS tool has a start-up time of 0.9sec and a total discovery time of 2.4sec using 25 services. When using 250 services, it takes between 3 and 25 seconds to finalize the discovery, depending on the complexity of the query, so the results can be similar in some cases.

4.2.4 Conclusion

The main conclusion after the PoC evaluation is that one of the main factor which affects performance is the number of services available in the registry and the reasoner used. In this sense, there is a relationship with the accuracy of the results are related. Using complex queries gives more accurate results, but this requires more time to process the query, as the reasoner needs more computational resources.

As the performance measures show, the influence of the amount of services available, ontology concepts, the complexity of the query, etc... is so important that trade-offs in performance related to calling an intermediate component (for managing interoperability or for managing several matching algorithms) are insignificant. The main problem is in the solution itself, as reasoning requires a lot of computation and memory.

An important trade-off we have found is related to scalability regarding the number of services available in the registry and potential candidates. We found a problem when trying to use the tool with 10000 services (the tool failed), which may imply problems in IoS environments. This is because of the reasoner and the way it works, so the selection of this component is very important.

In the case of using remote repositories, it is possible that the performance decreases because of network problems. This affects clearly reliability as well, as when connection with registries fails, it is not possible to obtain candidates. This can be solved in E-SOA environments, where the company can control its registry, but in IoS environments, if there is
not a mechanism for contacting with several registries, the discovery tool cannot recover from the failure.

Finally, buildability and modifiability are related, although buildability can depend as well in other factors (such as the complexity of the solution, the features provided and the information needed to operate the tools). But, usually, it is no so difficult to achieve good modifiability levels by doing a bit more complex the design and the initial development. We can consider that including extension mechanisms is worthwhile when the mechanisms do not imply too much development (no more than the application of a classic ‘strategy’ pattern) and requires only a few adaptations of existing components.

With these observations in mind, we suggest to include mechanisms for improving the reliability related to the registry and the reasoner, trying to improve scalability as well (by enabling the usage of threads and distributed reasoners, for instance). Moreover, we support the decision of including components and classes in the architecture which enable the usage of different algorithms and the extension of functionalities.
4.3 Report of Assessment of Requirements (Objectives) of Scenario

The scenario selected for performing the evaluation is the S10: Crisis Management System of Systems. As there is no a concrete scenario which is only focused on the discovery of services (it is something which is necessary only in concrete stages of the scenarios), we have selected a scenario which requires several heterogeneous systems interacting, and which provides a good guidance about the steps performed during the scenario. Those steps are very helpful in order to understand the domain well enough for defining an ontology and for generating service descriptions for simulated services which could be discovered.

The main requirements found in NEXOF-RA related to service composition are:

- R3 – Service Discovery Mechanisms
- R34 – Service Discovery

Related to these, the main functional requirements of the system fulfilled by the patterns should be:

- SR3 - How can a service be discovered?
- SR3.1.2 - How can a service that satisfies client requirements be found (searching)?

In order to check whether these requirements are fulfilled, some check points have been defined:

- Domain ontology: is a key element which has to exist, so knowledge can be represented
- Services available: it is necessary that a registry is populated with several services (represented by their descriptions)
- Query: users will send their queries for discovering services, so it is necessary one for launching the process
- Candidate Services: as a mean to demonstrate that some result has been received

These check points will be evaluated in order to determine if the stated requirements can be fulfilled.

4.3.1 Link between requirements and quality attributes

As the requirements stated are functional requirements about basic features of a service-centric system, there is no a clear relationship with the quality attributes defined in the patterns.

Accuracy is an aspect which can be considered important in this sense, although it is not considered one of the quality attributes in the patterns. But, as it is related to performance, it has been measured in the experiments, as shown in the previous section. It is a good metric to evaluate the level of fulfilment of the requirements.
4.3.2 Evaluation of requirements

The mentioned check points have been evaluated in order to determine if the patterns are able to fulfil them. The following table shows the results:

<table>
<thead>
<tr>
<th>SR3 &amp; SR3.1.2</th>
<th>Domain ontology</th>
<th>Two ontologies were defined</th>
</tr>
</thead>
<tbody>
<tr>
<td>Services available</td>
<td>A set of more than 1000 descriptions were created and stored in the repository</td>
<td></td>
</tr>
<tr>
<td>Query</td>
<td>Some queries were defined for the experiments</td>
<td></td>
</tr>
<tr>
<td>Candidate services</td>
<td>The service discovery tool gave a list of candidate services according to the queries sent</td>
<td></td>
</tr>
</tbody>
</table>

Table 11: Evaluation of requirements of Service Discovery PoC10

In a first stage, the team generated two ontologies according to the Crisis Management domain. The ontologies are based in the concepts and activities described in the WP10 scenario plus some ideas about potential data to be used. One of the ontologies is focused on the different activities which can be performed in the domain, while the other ontology is related to the inputs and outputs of those activities in the same domain.

Both ontologies were designed with the Web Service Modelling Toolkit v2.0. The first ontology can be seen in the next figure. It is based on three main branches: Information Management, Airport Management and Crisis Management.
The second ontology is built using two main concepts in mind: Primitive Type and Complex Type. The first one represents the simpler data types, while the second represents the data types which are obtained using primitive types in complex data structures.
Using a tool for automatically generating similar service descriptions, the team developed up to 10000 services descriptions and stored them in the registry set up for the experiments. Figure 23 shows an example of a service description.

Then, a query was developed according to some of the activities defined in the scenario, in order to simulate an expected query in such a situation. The WSML query shown here is based only in the functional classification and it is not too complex.

```wsml
?x memberOf _"http://www.wsmo.org/ns/wsmo-lite#Service" and
?x["http://www.wsmo.org/ns/wsmo-lite#hasFunctionalClassification" hasValue {?cls0, ?cls1}] and ?cls0 subConceptOf
"http://www.nexofra.eu/airport_crisis_management#HumanResourceManagement" and
?cls1 subConceptOf
"http://www.nexofra.eu/airport_crisis_management#MilitaryInformationGathering"
```

Figure 23: Ontology for inputs and outputs

Figure 24: WSML query for discovering services
Figure 25: Example of service description

As explained in section 4.2.3, the accuracy was good in general during the experiments, when it is possible to find candidates. When we were using less than 200 services, no candidates were found, but from 200 to 1000 candidate services are always found. According to the accuracy measured in the utility tree, it is clear that there are several aspects which influence in the level of fulfilment of users' requirements, in terms of reception or not of candidates. Candidates received are always meaningful, although sometimes there might be problems finding them, but taking into account the observations given in section 4.2.3, it is possible to obtain better results.

4.3.3 Conclusion

The achievement of these patterns is an important step towards the achievement of those requirements related to service discovery in the NEXOF-RA, so users can search for services when they are developing services compositions or when they want just to find a service to integrate it in other applications.

The assessment of functional and system requirements, in conjunction with the accuracy metric obtained before, completes the previous evaluation based on quality attributes. It is demonstrated that it is possible to implement patterns which are oriented to fulfilling these requirements.

This approach requires some previous work (such as the definition of ontologies), but it is a step which only is necessary once, and the ontologies can be reused many times in the same domain.

It is possible to fulfil the requirements, obtaining good candidate services with semantic based queries, but it is necessary to define correctly the queries, the ontology and the
services descriptions in order to achieve the full potential of this kind of approach. The problem is that these descriptions and queries depend on humans who may have different point of views, so the accuracy of descriptions and queries is really important.
5 ASSESSMENT OF PoC11: SUPPORTING SERVICE MATCHMAKING AND RANKING THROUGH RUNTIME SERVICES MONITORING - CINI

5.1 PoC’s selection

This PoC has been selected and the setup based on a proposal coming from WP10 and performed by CINI. The two evaluated patterns are part of the ESOA top level pattern.

The patterns evaluated in this PoC are the following:
- Service Matchmaking and Ranking
- Monitoring in Enterprise SOA

5.2 Pattern: Monitoring in Enterprise SOA

The monitoring pattern should provide features to constantly watch the behaviour of services and take future actions based on historical data. Monitoring is used to check if the used services guarantee certain quality of service (QoS) to their consumers, and to make accurate management decisions. The pattern specifies three levels of monitoring.

1. Application/Service level: in which the functional software components are monitored by means such as Service Level Agreements (SLAs).
2. NEXOF Compliant Infrastructure (NCI) level: monitoring the software components that constitute the NEXOF infrastructure such as runtimes, registries.

---

Figure 26: The monitoring process
3. Computational Resource Infrastructure level: manage the hardware and software resources that constitute the substrate on which the NCIs run such as virtual machines, hosts.

In the figure above we depict the main interactions between the provider/consumer and the monitoring component. The different components are depicted as boxes while the interactions are depicted as arrows connecting the components. The monitoring component is directly connected to the registry and every time new services are registered data starts to be collected. The monitoring component interacts with agents on the provider and user sides, which collect information about different attributes of the invoked services such as performance, availability etc.

5.2.1 Scope of the PoC with respect the Reference Architecture

The objective of this PoC is to use monitoring to support Service Matchmaking and Ranking and so to implement the patterns Service Matchmaking and Ranking and Monitoring in Enterprise SOA specified in WP7 (D7.5). These patterns are then assessed by evaluating their effect on the QA as well as the coverage of monitoring requirements of D10.1.

The following figure shows the scope of the PoC with respect to the NEXOF-RA (from a monitoring in Enterprise SOA perspective).

![Figure 27: Scope of the PoC with respect to the NEXOF-RA](image)

This PoC implements the "Monitoring in Enterprise SOA" pattern, please see the Reference Architecture Model (D6.1) for the definition of the concept "Service monitoring" as well as the "Monitoring in Enterprise SOA" pattern for a complete description of this pattern.
We take advantage of the following standards in this PoC:

- Java, a programming language developed by Sun Microsystems
- XML (Extensible Markup Language), a set of rules for encoding documents in machine-readable form developed by the W3C
- WSDL (Web Services Description Language), an XML-based language to describe Web services
- SOAP (Simple Object Access Protocol), an XML-based protocol specification for exchanging structured information in the implementation of Web Services.

Moreover the PoC refers to those abstract and concrete components that are defined in the implemented pattern (see the "Monitoring in Enterprise SOA pattern", section 8.1 and 8.2)

5.2.2 Evaluation methodology

The pattern is evaluated according to the quality attributes it claims. These quality attributes are stated in the filled-in template description of the architectural pattern (according to a shared and agreed template provided by WP7). The stated quality attributes in this architectural pattern description are evaluated using the ATAM approach. Especially the quality attributes utility tree is used to show and analyze the results of the implementation of each quality attribute. We developed JUnit\(^5\) test cases to evaluate the quality attributes stated by the PoC.

5.2.2.1 Quality attributes provided to WP7 for this pattern

In the context of their specification, the patterns have been identified to support certain defined quality attributes. These quality attributes affect the quality of the architecture where each pattern is integrated. Evaluating these quality attributes permits to know if the resulting architecture where the patterns have been applied meets the expected requirements.

[The significations of plus (+), minus (-) or neutral (0) are given in the specification of these pattern for D7.5]

<table>
<thead>
<tr>
<th>Monitoring in Enterprise SOA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
</tr>
<tr>
<td>Maintainability</td>
</tr>
<tr>
<td>Performance</td>
</tr>
<tr>
<td>Adaptability</td>
</tr>
</tbody>
</table>

Table 12: Monitoring in Enterprise SOA Pattern Trade-off

5.2.2.2 Evaluating quality attributes by using Pattern Quality Attributes Tree

The goal of the PoC is to demonstrate the feasibility of the objectives by means of a running and working implementation. For the evaluation of the quality attributes we used the method of feasibility prototyping.

\(^5\) JUnit, http://www.junit.org
The hereafter described assessment criteria are derived from the rationale for the effects on quality attributes defined within the pattern descriptions (section 4.1 of the pattern description):

- **Availability** is increased because the monitoring component can identify failing services as well as "can monitor its own components. In this way, a failure on an agent implementing a particular role can be detected."

- **Availability, Maintainability, Adaptability** is "eased by decomposing the Monitoring Tool into loosely coupled sub-components. Each sub-component fulfils one functionality (playing a concrete role in the monitoring tasks), so it is possible modify the implementation of a particular functionality without affecting the rest of the components."

- **Performance** "the monitoring tasks may affect negatively (degrade) the performance of the services provided by an E-SOA"

---

Figure 28: Evaluating quality attributes using the ATAM approach
5.2.3 Evaluation results

The evaluation results are based on executing different scenarios and collect the relevant information to validate the quality attributes defined in our utility tree.

We applied the method of feasibility prototyping: i.e., to develop a partial solution that to evaluate whether the system satisfies a subset of the requirements. This method is to a certain degree subjective because it depends on the context (other patterns used in combination with the here described pattern, work load, etc.).

To verify the aspects Availability and Adaptability we implemented the following test cases:

- "Basic configuration": setup system of
  - 1 service provider
  - 1 user
  - 1 monitoring component (to monitor the service provider)

- "Watchdog": setup system of
  - 1 service provider
  - 1 user
  - 2 monitoring components (1 to monitor the service provider, 1 to monitor the primary monitoring component)

- "Faulty service": setup system of
  - 1 service provider
  - 1 user
  - 1 monitoring component (to monitor the service provider)

  where the monitoring component monitors a functionality of the service provider. After 1 second the service provider is stopped, the monitoring component reports this failure.

- "Pull approach": setup system of
  - 1 service provider
  - 1 user
  - 1 monitoring component (to monitor the service provider)

  using the pull approach, user pulls monitoring data every second.

- "Push approach": setup system of
  - 1 service provider
  - 1 user
  - 1 monitoring component (to monitor the service provider)

  using the push approach, provider pushes monitoring event every second.

Maintainability is enhanced through the development of modular components that can be changed and replaced independently from the rest of the system. Excessive coupling between object classes decreases design modularity and makes the system more sensitive to the changes. Low coupling between objects and high lack of cohesion of methods [8] metrics indicate decreased encapsulation and increased complexity while the low value implies high cohesion and well design [1]. We have evaluated these two attributes by
analyzing the source code of the PoC obtaining an LCOM value between [0,1] and a CBO of <5, which indicates reuse possibilities and increased modularity in the design [14].

Performance guarantees the level of usability of the system. In this context we speak of two types of impact to performance: first, the performance decreases because the web service is tested by the monitoring component (due to an increased workload), and second, the performance decreases because every message has to pass through a client and a provider proxy.

To verify that these two attributes have a negative impact on performance, we implemented the following tests:

- "Workload":
  - 1 service provider
  - 1 user
  - 1 monitoring component (to monitor the service provider)
  
  for one minute the total number of calls without monitoring is counted, for one minute the total number of calls with simulated monitoring calls is counted.

- "Proxy":
  - 1 service provider
  - 1 user
  - 1 monitoring component (to monitor the service provider)
  
  for one minute the service is called without proxy, for one minute with both proxies

**Determination of sensitivity points: Availability and Monitoring**

The availability of a service delivered by a provider can be increased by monitoring. Increasing the intensity of monitoring, i.e., the amount of functionality that is monitored, or the frequency of monitoring, or the thoroughness of monitoring (e.g., monitoring just the existence of a functionality or also its correct result) increases also availability.

![Figure 29: Sensitivity point: Monitoring intensity impact on availability](image-url)
On the other hand, the performance of a service delivered by a provider is influenced by monitoring. Increasing the intensity of monitoring, has a negative impact on performance (see figure below).

![Figure 30: Sensitivity point: Monitoring intensity impact on performance](image)

**Determination of trade-off point: Availability and Performance**

In summary, monitoring intensity is a trade-off point: depending on the business goals a balanced level between performance and availability has to be chosen.

![Figure 31: Trade-off point: Monitoring intensity](image)
5.3 Pattern: Service Matchmaking and Ranking

The Service Matchmaking and Ranking should provide capabilities to help consumers of a SOA infrastructure to find the best services that match their needs. Due to the increase number of available services, it is mandatory to implement infrastructures to assist consumers while searching for services. To this end, Matchmaking and Ranking is a key enabler for ESOA, since it solves the problem of effectively and efficiently identifying and selecting the appropriate services that fit the consumer’s requirements.

In practice, consumers should be able to provide queries to the system in which they specify the required characteristics of a service. The system uses the consumer description to find the appropriate services and rank them based on their relevance to the required characteristics. This way the consumer receives more accurate results.

In the figure above we depict the main interactions between the consumer and the Matchmaking and Ranking component. The different components are depicted as boxes while the interactions are depicted as arrows connecting the components. The matchmaking receives users' requests and searches in the service's repository for the hits that match the request. Once the matches are found they are ranked and returned to the user.

5.3.1 Scope of the PoC with respect the Reference Architecture

The objective of this PoC is to use monitoring to support Service Matchmaking and Ranking and so to implement the patterns Service Matchmaking and Ranking and Monitoring in Enterprise SOA specified in WP7 (D7.5). These patterns are then assessed by
evaluating their effect on the QA as well as the coverage of monitoring requirements of D10.1.

The following figure shows the scope of the PoC with respect to the NEXOF-RA (from a service and matchmaking perspective).

![Figure 33: Scope of the PoC with respect to the NEXOF-RA](image)

This PoC implements the "Service Matchmaking and Ranking" pattern; please see the Reference Architecture Model (D6.1) for the definition of the concept «Service discovery» as well as the "Monitoring in Enterprise SOA" pattern for a complete description of this pattern.

We take advantage of the following standards in this PoC:

- Java, a programming language developed by Sun Microsystems
- XML (Extensible Markup Language), a set of rules for encoding documents in machine-readable form developed by the W3C
- WSDL (Web Services Description Language), an XML-based language to describe Web services
- SOAP (Simple Object Access Protocol), an XML-based protocol specification for exchanging structured information in the implementation of Web Services.

### 5.3.2 Evaluation methodology

The pattern is evaluated according to the quality attributes it claims. These quality attributes are stated in the filled-in template description of the architectural pattern (according to a shared and agreed template provided by WP7). The stated quality attributes in this
architectural pattern description are evaluated using the ATAM approach. Especially the quality attributes utility tree is used to show and analyze the results of the implementation of each quality attribute.

5.3.3 Quality attributes provided to WP7 for this pattern

In the context of their specification, the patterns have been identified to support certain defined quality attributes. These quality attributes affect the quality of the architecture where each pattern is integrated. Evaluating these quality attributes permits to know if the resulting architecture where the patterns have been applied meets the expected requirements.

[The significations of plus (+), minus (-) or neutral (0) are given in the specification of these pattern for D7.5]

<table>
<thead>
<tr>
<th>Matchmaking and Ranking pattern</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td>0</td>
</tr>
<tr>
<td>Buildability</td>
<td>+</td>
</tr>
<tr>
<td>Maintainability</td>
<td>+</td>
</tr>
<tr>
<td>Integrability</td>
<td>+</td>
</tr>
<tr>
<td>Interoperability</td>
<td>0</td>
</tr>
<tr>
<td>Modifiability</td>
<td>+</td>
</tr>
<tr>
<td>Adaptation to new operating environments (portability)</td>
<td>+</td>
</tr>
<tr>
<td>Performance (efficiency)</td>
<td>+</td>
</tr>
<tr>
<td>Recoverability</td>
<td>0</td>
</tr>
<tr>
<td>Reliability</td>
<td>0</td>
</tr>
<tr>
<td>Resource efficiency</td>
<td>+</td>
</tr>
<tr>
<td>Scalability</td>
<td>+</td>
</tr>
<tr>
<td>Security</td>
<td>0</td>
</tr>
<tr>
<td>Testability</td>
<td>+</td>
</tr>
<tr>
<td>Usability</td>
<td>+</td>
</tr>
</tbody>
</table>

Table 13: Matchmaking and Ranking pattern Trade-off

As stated in the "Service Matchmaking and Ranking" pattern, attributes such as availability, interoperability, recoverability, reliability, and security are not affected by the decisions taken, since no mechanisms have been included that take them into account. For the same reason we are not able to assess them using this PoC.
The "Service Matchmaking and Ranking" pattern description states that the positive effect on Reusability is limited to those environments where the "Service Discovery" pattern is applied. This does not apply to this PoC, this quality attribute is therefore not evaluated.

Usability is concerned with how easy it is for the user to accomplish a desired task. An evaluation of usability at architectural level resolves into the following non-functional requirements [12]:

- Using a system efficiently: resource efficiency, performance
- Minimizing the impact of errors: modifiability, testability
- Increasing confidence and satisfaction: availability, reliability.

According to the pattern description a positive influence is an effect of a positive effect of the pattern on resource efficiency, performance, modifiability, and testability, i.e., will not be evaluated directly but through these four quality attributes.

5.3.4 Evaluating quality attributes by using Pattern Quality Attributes Tree

The goal of the PoC is to demonstrate the feasibility of the objectives by means of a running and working implementation. For the evaluation of the quality attributes we used the method of feasibility prototyping.

The here described assessment criteria are derived from the rationale for the effects on quality attributes defined within the pattern descriptions (section 4.1 of the pattern description):

- **Buildability, Maintainability, Integrability, Testability, Reusability:** "the decomposition of the component into loosely coupled sub-components with clear interfaces is positive for these quality aspects". Moreover, due to "the usage of interfaces it is easy to test and reuse this component without too much effort."
- **Modifiability and Portability:** the pattern is designed so that "the core discovery engine may implement several approaches for matchmaking and ranking", i.e., it can be modified and extended with new matchmaking and ranking algorithms easily
- **Portability:** "It is possible to perform any implementation in any language", we will implement it using a portable language: Java
- **Scalability:** "the design allows an implementation of the components using threads, which would provide a good response time as the number of clients increases."
- **Performance, Resource efficiency:** "The efficiency will depend on the implementation of algorithms themselves, but the pattern offers the main components for adequate the selection of algorithms to the context of the query".
5.3.5 Evaluation results

The evaluation results are based on executing different scenarios and collect the relevant information to validate the quality attributes defined in our utility tree.

We applied the method of feasibility prototyping: i.e., to develop a partial solution that evaluates whether the system satisfies a subset of the requirements. This method is to a certain degree subjective because it depends on the context (other patterns used in combination with the here described pattern, work load, etc.).

Testability, Buildability, Maintainability, Integrability is enhanced through the development of modular components that can be changed and replaced independently from the rest of the system. Excessive coupling between object classes decreases design modularity and makes the system more sensitive to the changes. A low coupling between objects and high lack of cohesion of methods [13] indicates decreased encapsulation and increased complexity while the low value implies high cohesion and well design [1]. We have evaluated these two attributes by analyzing the source code of the PoC obtaining an LCOM value between [0.1] and a CBO of <5, which indicates reuse possibilities and increased modularity in the design [9].

To verify the aspects Modifiability, Performance, Resource efficiency, and Portability we implemented the following test cases:

- "Multiple approaches": setup system of
o 2 approaches for matchmaking and ranking
o 1 client which sends requests using either the first or the second approach

- "Fast/Slow": setup system of
  o 2 approaches for matchmaking and ranking, one with immediate response, one with a delay
  o 1 client which sends requests using either the first or the second approach

Portability is furthermore verified by implementing this PoC using a Language with high portability: Java.

Scalability in this pattern is intended as "it provides a mean to combine a lot of different approaches." as well as a good response time since "the design allows an implementation of the components using threads" (see pattern description). This was validated through an implementation based on threads as well through the following test case:

- "Threading": setup system of
  o generation of 100 different approach instances for matchmaking and ranking that take 1 minute to terminate
  o generation of 100 different client instances requesting a random service
  o verification that all clients wait about 1 minute

**Determination of sensitivity points: Memory amount and maximum scalability**

To ensure scalability using threading has to verify that the available memory is sufficient, or vice versa: the available amount of memory influences the scalability limit.

![Figure 35: Sensitivity point: Memory amount and maximum scalability](image)

On the other hand, the performance of a service delivered by a provider is influenced by monitoring. Increasing the intensity of monitoring, has a negative impact on performance (see figure below).
5.4 Report of Assessment of Requirements (Objectives) of Scenario

This PoC demonstrated the scenario S1, Service procurement. It focuses on the following (functional) requirements to be evaluated:

- SLA processing (R1)
- Monitoring and reliability (R31)
- Monitoring information (L2.3)
- Awareness of unreliable services (L3.4)
- Event based monitoring (L7.2)
- History based monitoring (L7.3)
- Assertion based monitoring (L7.4)
- Dynamic regulation of monitoring activities (L7.6)
- Service Data Collection (L7.7)
- Environmental Data Collection (L7.8)
- Dynamic renegotiation of constraints (L7.9)
- Extensible framework for data collectors, data analyzers and constraint metrics (L7.10)

5.4.1 Link between the PoC scenario (scenario S1) and quality attributes

The non functional requirements found relevant for this scenario by the stakeholders are:

- Availability
- Interoperability
- Modifiability
- Performance
- Reliability
- Security
- Usability

Figure 36: Sensitivity point: Monitoring intensity impact on performance
## 5.4.2 Evaluation of requirements

<table>
<thead>
<tr>
<th>Requirements / Objective</th>
<th>Check point</th>
<th>How to verify that the check point is reached?</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLA processing (R1)</td>
<td>Matchmaking and Ranking component has to be able to consider service levels to publish and retrieve services</td>
<td>Setup two providers that publish their services providing also the given service levels. Setup a client that can choose between the two providers depending on the required service level.</td>
</tr>
<tr>
<td>Monitoring and reliability (R31)</td>
<td>It is possible to alert a vendor of system failures</td>
<td>Setup test case &quot;Faulty service&quot;</td>
</tr>
<tr>
<td>Monitoring information (L2.3)</td>
<td>It is possible to monitor a service and to obtain the information in a timely fashion</td>
<td>Setup test case &quot;Push approach&quot;</td>
</tr>
<tr>
<td>Awareness of unreliable services (L3.4)</td>
<td>The system is aware of services that do not deliver what is expected</td>
<td>Setup test case &quot;Faulty service&quot; (to verify services published by a vendor) or &quot;Watchdog&quot; (to verify the monitoring service itself)</td>
</tr>
<tr>
<td>Event based monitoring (L7.2)</td>
<td>Monitoring based on events is possible</td>
<td>Setup test case &quot;Push approach&quot;</td>
</tr>
<tr>
<td>History based monitoring (L7.3)</td>
<td>Monitoring based on analyzing the monitoring history is possible</td>
<td>In this PoC, all collected data is stored in a local database. Setup a client and a provider that use the information stored in that database.</td>
</tr>
<tr>
<td>Assertion based monitoring (L7.4)</td>
<td>Monitoring based on assertions is possible</td>
<td>Setup test case &quot;Push approach&quot; and react only to specific events.</td>
</tr>
<tr>
<td>Dynamic regulation of monitoring activities (L7.6)</td>
<td>It is possible to change the monitoring activities on runtime</td>
<td>Setup a client and a provider where the client makes use of the service published by the provider. Before calling the service another time, modify the monitoring policy (using the &quot;configureMonitoringPolicy&quot; method, see the &quot;Monitoring in Enterprise SOA “pattern description). Now use the service another time and verify that now the collected data through monitoring adapted to the new monitoring policy.</td>
</tr>
<tr>
<td>Service Data Collection (L7.7)</td>
<td>Monitoring data is available on run-time</td>
<td>Setup a client and a provider, the client makes use of the service published by the provider. Verify that monitoring data is available on run-time</td>
</tr>
<tr>
<td>Environmental Data Collection (L7.8)</td>
<td>Data concerning the environment is collected on the client and on the provider side</td>
<td>Setup a client and a provider, the client makes use of the service published by the provider. Verify</td>
</tr>
</tbody>
</table>
that data about the execution environment is collected from the client and provider side.

| Dynamic renegotiation of constraints (L7.9) | Users can change their constraints and obtain a new service matching their constraints. | Setup a client and two providers where the client makes use of the service published by the first provider. Before calling the service another time, change the constraints required by the client so that the second service should be retrieved. Now obtain a list of services fulfilling the constraint, use the new service and verify that the constraint is respected. |
| Extensible framework for data collectors, data analyzers and constraint metrics (L7.10) | It is possible to add new interceptors and data analyzers | To add a new interceptor, add an interceptor class to the client and/or provider proxy to collect the new type of data. To add a new data analyzer, add a new analyzer class on the server side (where the monitoring component is installed) that adds a new class that interprets the collected data. |

Table 14: Evaluation of requirements of supporting service matchmaking and ranking through runtime services monitoring PoC11

5.4.3 Conclusion

The achievement of these patterns is an important step towards the achievement of those requirements related to “Supporting Service Matching and Ranking Through Runtime Services Monitoring” in the NEXOF-RA, so users can search for services when they are developing services ranking / monitoring or when they want just to find a service to integrate it in other applications.

The assessment of functional and system requirements, in conjunction with the accuracy metric obtained before, completes the previous evaluation based on quality attributes. It is demonstrated that it is possible to implement patterns which are oriented to fulfilling these requirements.
6 ASSESSMENT OF PoC12: INVESTIGATION OF PATTERNS COMPOSABILITY BASED ON THE PATTERNS FOR SERVICE DISCOVERY - TIE

TIE assesses a PoC called PoC12: The Patterns’ Composition and Effects on Quality attributes. This PoC investigates relationships between different patterns for service discovery that are applicable in a number of SOA families including Internet of Services (IoS) and Enterprise SOA. The PoC validates how alternative patterns could be used separately and together and what affects it has on the quality characteristics of the resulting system.

6.1 PoC’s selection

This PoC has been selected due to its relevance for several SOA families including IoS and ESOA. Although within NEXOF-RA a link between discovery and Cloud computing has not been made explicit it is known that Cloud needs discovery of appropriate resources/services in order to fulfill users’ demands, e.g. for searching for Storage-as-a-Service, etc…

Another point for this PoC has been gained because it provides an analysis of possible alternatives and their potential common use within one system. This paves a road towards instantiation guidelines necessary for the end users of NEXOF-RA after the end of the project.

The evaluated patterns are Service Matchmaking and Ranking, Multi-phase Discovery pattern and Template-based discovery pattern (see D8.1c for details).

6.2 Patterns for Discovery concern: composition and quality characteristics

With the emergence of Web services and the Service Oriented Architectures (SOA), processes are more and more being decoupled. Combining these technologies will create a wide network of services that collaborate in order to implement complex tasks.

However up-to-date, because the market for Web services is far from transparent, there are not many Web service search engines widely available. The ones that exist usually only search for Web services based on their UDDI registration and their WSDL description. One example is the search eSynaps engine. Seekda! tries to go further, by extracting semantics from the WSDL files, which enables runtime exchange of similar services and composition of services. Seekda! does not yet search through existing semantic Web service description files, but only makes use of the WSDL file of a Web service because currently, Web services are commonly described via narrative Web pages containing information about their operations in natural languages.

Web pages contain plain text with no machine interpretable structure and therefore engines such as Service-Finder tries to gather information about services from different sources throughout a Web 2.0 environment. The information is automatically added to a semantic model so that flexible discovery of services can be realized.

Other approach is demonstrated by GODO, which is a Goal-Driven approach for searching WSMO Web services. It consists of a repository with WSMO Goals and lets users state their goal by writing a sentence in plain English. A language analyzer extracts keywords from the user sentence and a WSMO Goal will be composed based on those keywords. The WSMO
Goal with the highest match will be sent to WSMX, an execution environment for WSMO service discovery and composition. WSMX then searches for a WSMOWeb service that is linked to the given WSMO Goal via WSMO Mediators and return the WSMO Web service back to the user. This approach makes good use of the capabilities of the WSMO framework, but it cannot be applied for other semantic languages like OWL-S and WSMO-Lite (the output of another NESSI strategic projects named SOA4ALL), which do not have such goal representation elements.

Another approach for semantic Web service discovery is based on searching for similarities among different service descriptions. Several mediation techniques to identify semantic similarities between ontologies, e.g., by using Mediation Spaces, have been developed. They mediate on data-level as well as semantic level to discover related semantic Web services according to ontologies, other semantic Web services or WSMO Goals.

Thus, it is demonstrated that there are a number of alternatives how to organize service search for the Internet of Services domain but not much guiding practical principles and comparison analysis. NEXOF-RA employs a pattern based approach of documenting, comparing and classification of architectural choices and best practices. Thus, the main motivation for this PoC is two-folded:

- Pragmatically an architect needs to see how different alternatives could compete or if possible co-exist, how they would behave in combinations with each other and so this PoC is performing comparison analysis.
- Current solutions are partial since service providers and consumers being locked by platform vendors via a set of standards and APIs. This PoC investigates interoperability of solutions based on patterns provided by NEXOF-RA.

6.2.1 Scope of the PoC with respect to the Reference Architecture

The PoC is based on several patterns that are integrated into NEXOF-RA by WP7 and horizontal investigation WPs. For the detailed descriptions of those patterns and their relationships it is recommended to study deliverables of WP7. Here it is given a short view at the components involved into this PoC biased on the NEXOF-RA model.
The following Figure 37 gives a short overview on how the implementation of the PoC is structured.

The developed framework essentially is an instantiation of a system based on the NEXOF-RA model, patterns and standards. Within this PoC authors have been building a system that fulfills business and technical requirements elicited from a number of scenarios, especially from the one from e-Commerce area by systematically implementing patterns and using standards found within NEXOF-RA documentation.

The resulting framework for service search is rather flexible and could be used in a number of domain including IoS and ESOA. It is also open to the extensions and modifications due to the use of patterns and interfaces among them.

- **WSMO.** For an initial implementation only a WSMO-Web service reader and a WSMO-Ontology reader have been implemented, but based on the modularity of the implementation, the engine can be extended with readers that can parse other Semantic Web service languages.
- **Axis2.** Apache Axis2 is a core engine for Web services. It is a complete redesign and rewrite of the widely used Apache Axis SOAP stack. Implementations of Axis2 are available in Java and C.
- **WSMO4J.** To read the WSMO files WSMO4J is used. WSMO4J is an API and a reference implementation for building Semantic Web Services and Semantic Business Process applications based on WSMO. By using WSMO4J, WSMO files can be parsed, read and written. For the Disambiguator WordNet is used through two WordNet API's, to find senses of words.
**JWordNetSim** is used to calculate the similarity between two WordNet senses. For the part-of-speech tagging the Stanford parser is used. The overall implementation is made in Java, due to the availability of external packages written for WSD and WSMO parsing and reading.

![Component diagram of the PoC implementation](image)

A point should be made concerning relationships with a standard catalogue, because languages that are used for describing services are very diverse and in order to address user requirement R5 “Service description” (for more details see NEXOF-RA D10.1) it is necessary to mention them specifically. The following picture demonstrates a result of critical analysis and classification of the available languages/technologies used for service description/annotation at the moment (June 2010).
The top layer on Figure 39 consists of semantic Web service description languages such as Web Service Modeling Ontology (WSMO) and OWL-S. These languages employ ontologies for describing the behavior of a Web service. Concepts, attributes and relations from existing ontologies and logical expressions can be used to state conditions and effects of a Web service. To provide a bridge between the syntactical languages such as WSDL and hRESTs to the semantically enriched languages such as WSMO and OWL-S, middle layer languages were defined. MicroWSMO, SA-REST and Semantic Annotations for WSDL (SAWSDL) link the concepts from the semantic descriptions with the data types for the input and output of a Web service, or with its operations, and provide methods to transform data types to semantic concepts and the other way around.

6.2.2 Evaluation methodology

The pattern is evaluated according to quality attributes claimed by their authors/contributors. Those quality attributes are coming from the documentation provided by WP7. ATAM is used as a basis methodology that has been employed by several WPs and it guides as assessment of quality attributes within PoCs. Utility tree visualises and helps in a demonstration of validation procedures and analysis of the results.

6.2.2.1 Quality attributes provided to WP7 for these patterns

Specification defines patterns’ quality attributes how they have been identified by their contributors. These attributes affect the quality of the resulting architecture when pattern are combined together. This PoC investigates the quality characteristics that are in the list of attributes of every selected pattern.
Discovery concern patterns

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interoperability</td>
<td>+</td>
</tr>
<tr>
<td>Performance</td>
<td>+/-</td>
</tr>
<tr>
<td>Reliability</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 15: Composition of Service Discovery Pattern Trade-off

Besides those attributes the question about a possibility of those patterns to work together was also investigated. This question had been resolved during a design and implementation phase of this PoC.

6.2.2.2 Evaluating quality attributes by using Pattern Quality Attributes Tree

A chosen method of prototyping gives an answer on a question concerning a composibility and interoperability of patterns. The ATAM utility trees have been used for matching the technical quality attributes with metrics to be used in the evaluation. Series of tests have been executed to answer the questions concerning interaction of quality attributes against prepared test set of service descriptions. Quality attributes and associated metrics will be derived from patterns descriptions submitted to the NEXOF-RA WP7.

![Figure 40: Valuating quality attributes with ATAM Approach](image)

Firstly, a test data set with a number of semantic service descriptions were created. Then, for testing the patterns combinations and their quality attributes, 61 test queries have been defined to check the outcomes of each of the pattern compositions. These queries represent possible sets of keywords a user might use to search for a Web services. For each query, a list of preferred Web services that are present in the repository, that should be returned as high as possible in the list of Web services provided to the user are defined.
### 6.2.3 Evaluation results

This section contains the results of the evaluation tests and prototyping. Due to the obvious differences in implementations the results of evaluation are shown in a quantitative way. The technical characteristics of the testing environment are presented in the following table.

<table>
<thead>
<tr>
<th>Number of computers</th>
<th>1, 2 (vary in different tests)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network</td>
<td>IEEE 802.3 / IEEE 802.11g</td>
</tr>
<tr>
<td>OS</td>
<td>Windows 7, 64bit</td>
</tr>
<tr>
<td>Runtime environment</td>
<td>JRE 1.6, update 20</td>
</tr>
<tr>
<td>Available RAM</td>
<td>1Gb</td>
</tr>
<tr>
<td>CPU</td>
<td>Intel Quad 8400 2.66GHz / i7Core Q820 1.73GHz</td>
</tr>
</tbody>
</table>

**Table 16: Technical details of a test environment**

#### Performance testing: user-system interaction

This performance test was not a standard one as described further on. The main point of this test was to find out a performance of a human-computer system and its productivity depending on the architectural choices made, i.e. usage of multi-phase discovery pattern alone or combined with template-based discovery pattern. The standard test with predefined queries was also conducted but the results matched the expectations, i.e. a performance of a system with just multi-phase discovery pattern was slightly higher than of a system that uses both patterns. The difference was just a time necessary to pass an information through the implementation of a template-based discovery pattern.

Since template-based discovery pattern allows creation of a user-friendly interfaces for searching services this PoC were measuring time starting from a point when a user was presented with a search goal until the moment of appearing a list of relevant services. Thus, a user had to read, understand and then formulate a query. First group of users was presented with two input fields one for verbs and another one for nouns. Another group had a reach interface with several fields and drop down menus pre-filled with expected/existing words coming from a system thesaurus.

The examples of goal presented to the users were simple ones such as “Find a service that allows you to order a food to your home”, “Find a service that allows you to calculate a current currency exchange rate”, or more intricate ones such as “Find out all services that allow to buy a pizza in your region and you would be able to pay with a X credit card type with minimum delivery time” or “Find the cheapest service that allow to compare historical data of a stock market for companies X, Y and Z”.

The test revealed the following sensitivity point as shown on the next figure.
The most important point to make here is that the more complicated queries need more elaborated and complex UIs to support them appropriately. It means more investments in UI and human-computer interaction in order to understand how to support user in the most effective way with the task he is facing.

Table 17: Guideline of template-based discovery pattern

<table>
<thead>
<tr>
<th>Number</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The usage of template-based discovery pattern implies investments into the research of domain area and human-system interaction design. The result still might not be satisfactory at some point when competing with a simple “Google-like” interface.</td>
</tr>
</tbody>
</table>

Performance testing: algorithms and their precision and recall ratio

Another aspect of performance is actual search performance (Precision and Recall) of an engine depending on a number and complexity of algorithms that it is using. To test the performances of the three matching algorithms, 61 tests have been done. The tests can be divided into two types. 33 tests have been done to measure the matching performance of the algorithms using queries that have been designed to search for Web services that are present in the repository. 28 tests have been prepared to measure the performance using queries that have been designed to search for Web services that are not present in the repository. In the latter case, a number of similar Web services from the repository have been used to test how good the algorithms can discover similar Web services.

Testing with 61 queries and three matching algorithms, provide 183 Precision-Recall graphs (PR-graphs). Because comparing so many graphs is impossible, PR-graphs consisting of average precision values for the recall points are created. This enables comparing all the different algorithms at once. However, the testing is done with lists of preferred Web services that can vary in the number of Web services they consists of. For testing, lists that contain two to four preferred Web services have been used. Because these variations in number of Web services cause different recall values, average precision values could only be calculated for queries that have the same amount of preferred Web services.
8 different PR-graphs can visualize the performances of the various matching algorithms. The four PR-graphs that are shown in Figure 42, show the average results for the exact matching tests. For each of the four different numbers of preferred returned Web services (n) a PR-graph is created. The four PR-graphs that are shown in Figure 43, show the average results for the approximate matching tests.

![Figure 42: PR-Graphs for discovery of exact matching services](image_url)
From the different PR-graphs that are shown in figure Figure 51, it is possible to see two observations. First, the Jaccard algorithm has in most cases a higher precision for the first half of the graph than the simple and the similarity algorithm. Second, all algorithms have about the same precision to provide a full recall. This means that to provide all the preferred Web services to the user, they need about the same amount of Web services to be displayed. So, the user has to scroll down an amount of services, that is the same for every algorithm, to find the last preferred service. However, according to the fact that the Jaccard algorithm provides a higher precision for a lower recall, the Jaccard algorithm provides at least some of the preferred Web services in an earlier stage to the user then the others. It can therefore be seen as the best algorithm to discover exact matching Web services.

From the different PR-graphs that are shown in figure Figure 55, we can make the observation that the similarity algorithm performs overall better for discovery of similar Web services than the Jaccard and the simple matching algorithm, as in most of the cases the precision lines of the similarity algorithm are above the line of the Jaccard algorithm.

Stress/load testing for reliability

Theoretically, it is well known from cybernetics that the more complex a system is (i.e. consists of increasing number of components with increasing number of relationships among them) the less reliable it would be. This is because of an increasing number of potential points of failure. Mathematically it is described by the following formula:
\[ P(A) = \prod_{i=1}^{n} P(A_i) \]

where \( P(A) \) is a probability of a system failure, \( P(A_i) \) is a probability of a failure of a component \( i \), and events of failure of each component are independent of each other.

The test shown that if the whole system (all services) are located on one physical computer than the failure could be caused only by exceeding a memory of a Java Runtime Environment. Such situation causes a complete crash of a system and typically needs an involvement of a personnel.

![Figure 44: Sensitivity Point: growing number of requests being processed by the system at the same time causes a crash](image)

When services were allocated on several (2) computers the failure were caused by a loss of requests, i.e. the more load the more requests were not answered.

![Figure 45: Sensitivity Point: growing number of requests being processed by the system at the same time causes a crash](image)
In both cases the results shows that the failure were not caused by an implemented system or patterns implementations. The conclusion drawn from this test in a form of guideline presented in the following table.

<table>
<thead>
<tr>
<th>Number</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Locate whenever possible components that communicate the most between each other on the same machine/web-server/application server to increase throughput and decrease potential loss of data</td>
</tr>
<tr>
<td>2</td>
<td>Encourage Intensive use of auto-build environments that run among other things extensive and comprehensive sets of unit and integration tests</td>
</tr>
<tr>
<td>3</td>
<td>Encourage use of well tested and proven third party libraries and frameworks, e.g. coming from Apache foundation or SpringSource</td>
</tr>
</tbody>
</table>

Table 18: Guideline of the conclusion drawn from the test

All those rules help seriously mitigating technical risks related to the reliability of a system.

**Interoperability and composability of patterns**

The patterns under consideration were composed and used together during a design phase using interfaces.

![Diagram](image)

Figure 46: Template-based discovery, Multi-phase service discovery and Matchmaking patterns in one system

This solution is based on the patterns provided by NEXOF-RA investigation teams via WP7 documentation process. Due to the intensive use of Interfaces and Services along with Spring DI each component’s implementation is interchangeable and interoperable as long as it is compliant with an interface.
**Determination of trade-off points**

Performance and reliability are affected by the

- Complexity of the system and
- Complexity of the queries
- Complexity of algorithms

After reaching a certain level of complexity (of a system and/or queries, algorithms) performance drastically dropping down.

![Trade-off point: performance vs. complexity](image)

**Figure 47: Trade-off point: performance vs. complexity**

The conclusions are made in form of guidelines (see the following table).

<table>
<thead>
<tr>
<th>Number</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>On-line interactive systems should use algorithms that guarantee quick response time. Current response time from an on-line system is estimated to be around 1 second in order to support a necessary Web 2.0 level of user experience.</td>
</tr>
<tr>
<td>2</td>
<td>If the discovery/selection process is performed off-line or the time is not critical (e.g. agent-agent interaction without strict time constraints) than more sophisticated algorithms could be used.</td>
</tr>
<tr>
<td>3</td>
<td>User interfaces (e.g. Template-based discovery pattern) could drastically increase a performance of a human-machine system but they need reasonable investments. After reaching some point of complexity of UI simple UIs could be again more efficient.</td>
</tr>
</tbody>
</table>

**Table 19: Guidelines of the conclusions of interoperability and composability of patterns**
6.2.4 Conclusion

The major point to be made after this PoC is that NEXOF-RA reference architecture that includes patterns descriptions, standard catalogue, etc. provides means for developing fully functional applications for a number of domains. It gives a set of possibilities that could be alternative to each other or could be combined together in order to alter the important non-functional qualities of the resulting system.

Then, the usage of modern best-practices for development such as Agile processes supported by adequate tooling could help in mitigation of a number of technical risks associated with architecture, e.g. reliability. For further information on this point please visit a web-site established by EU FP6 STASIS project at http://www.ict-development.eu/.

Although, this PoC does not explicitly measure modifiability of produced systems it is worth to mention that during an implementation it was clearly stated by an architecture and development team that SOA paradigm support of decoupling of components and separation of concerns helps creating systems which are more extendable and modifiable.

The rest of the important conclusions could be found on the appropriate sections above in a form of guideline principles.
6.3 Report of Assessment of Requirements (Objectives) of Scenario

This PoC is based on the S12 e-Commerce scenario (see D10.1) which was also served as a starting point for a scenario used within SOA4ALL, work package 9.

Scenario S12 is connected with the following (the most relevant in the context of this PoC) requirements coming from the NEXOF-RA work package 10:

- **R3**: Service discovery
- **R5**: Service description
- **R30**: Service integration and
- **R34**: Service discovery

The main goal of this scenario is to show how NEXOF-RA and particularly its parts such as patterns, standards and concrete components could be applied in combinations to in order to satisfy a real-world e-Commerce scenarios that associated with the Internet of Services type of SOA.

First of all, those underlying activities start with searching (discovery) of services that could fulfil specific needs/steps of the overall process (see Figure 48 point 1).

![Figure 48: E-commerce scenario](image)

6.3.1 Link between requirements and quality attributes

A number of quality attributes is associated with S12 scenario, e.g. performance, reliability, trust, scalability, and at the same time every pattern submitted to the NEXOF-RA WP7 has also a list of its own quality attributes. Thus, this PoC will also investigate a match of required qualities across work packages.
### 6.3.2 Evaluation of requirements

The requirements are exposed in certain check points. These check points must be verified by the implementation and performed tests.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Check point for</th>
<th>How to verify by the implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Discovery</td>
<td>Check point 1: Interoperability</td>
<td>see section “Interoperability and composability of patterns” of PoC 12</td>
</tr>
<tr>
<td></td>
<td>Check point 2: Performance</td>
<td>See section “Performance testing: algorithms and their precision and recall ratio” and “Performance testing: user-system interaction” of PoC 12</td>
</tr>
<tr>
<td></td>
<td>Check point 3: Reliability</td>
<td>See section “Stress/load testing for reliability” of PoC 12</td>
</tr>
</tbody>
</table>

Table 21: Evaluation of requirements of investigation of patterns composability based on the patterns for service discovery PoC12

### 6.3.3 Conclusion

This solution is based on the patterns provided by NEXOF-RA investigation teams via WP7 documentation process. The final system was proved to satisfy requirements elicited from a scenario coming from WP10.
7 ASSESSMENT OF PoC13: SEMI-AUTOMATIC SERVICE COMPOSITION AT DESIGN TIME - ATOS

Atos assesses a PoC called Semi-automatic Service Composition at Design Time that demonstrates and validates some patterns related to service composition in an IoS domain, by using a tool which exploits semantics.

7.1 PoC’s selection

These PoC has been selected and setup based on a proposal coming from WP2 and took over by ATOS based on work performed and achieved by WP2 in close cooperation with WP7. According to WP6, Service Composition is one of the NEXOF-RA main concerns and most of the scenarios require, at some stage, to develop a process implementation by using services composition.

The Design Time Service Composition IT has identified several patterns related to assisted composition, in order to support users in their developments. The patterns evaluated in the PoC are the following:

- **Assisted Composition Designer**, as the high level pattern
- **Semantic Annotation Composition**, as a way to implement the Assisted Composition Designer

For the pattern *Process-level Semi-Automated Composition*, we will use as reference the results of the evaluations performed in [10], as it includes good examples about the usage of ASTRO tools (which can be considered an implementation of the pattern) and can be used for comparing the approached.

7.2 Patterns: Assisted Composition Designer and Semantic Annotation Composition

Service Composition is a key functionality which, usually, is performed manually by business process developers. In order to support this activity, new approaches are defined for assisting users and reducing time of development and number of potential errors.

This functionality is necessary in any environment related to services, mainly E-SOA and Internet of Services, as composing services can be simple, but usually, it is a complex task. The problem is that automatic composition is very difficult, as it requires simulating human thinking in some aspects, and its complexity is even higher in very wide environments, such as IoS, where the knowledge about one domain is not enough.

While Assisted Composition Designer defines, in an abstract way, the main components involved in the (semi)automatic composition, Semantic Annotation Composition provides a more concrete architecture for solutions using semantics. It can be almost mapped directly to implementation classes, although it is still abstract enough for being applicable to different solutions based on semantics.

The components presented in this section can be used for both E-SOA and IoS domains, as semantics are useful in both contexts. They can be more effective in E-SOA, in concrete application domains, as better and richer ontologies can be defined, reducing the scope of the reasoning and not requiring mediation between ontologies.
These components represent the ProcessModelManager and CompositionEngine, contained in the Assisted Composition Designer pattern. So the PoC aims at applying a more concrete pattern reaching the implementation level.

7.2.1 Scope of the PoC with respect the Reference Architecture

![Figure 49 Semantic Annotation Composition Components](image)

![Figure 50: Scope of the PoC – Semi-automatic Service Composition at Design Time](image)
We have focused on the usage of SOA4ALL tools for evaluating the patterns, working in close cooperation with SOA4ALL members, as a way to evaluate their architecture as well. The main abstract components correspond to both patterns under evaluation. The Composer and the Models Manager correspond to components defined in the Assisted Composition Designer pattern, while the Reasoner is exclusive from the Semantic Annotation Composition. In fact, the Semantic Annotation Composition represents a whole Composer (as defined in the Assisted Composition Designer pattern).

In the case of the tools used, the patterns hierarchy is not exactly the same, although the components are more or less the same. In the case of SOA4ALL, the Blackboard is part of the SOA4ALL DT Composer, instead of being something external to the Composer, although the effect in the architecture is the same (it is more a matter of classes organization).

About the standards involved, as SOA4ALL DT Composer is based in WSMO and WS-BPEL, we consider that these are the main standards to be taken into account in semantic-based approaches. There are other semantic languages which can be used, but this seems to be very well positioned.

### 7.2.2 Evaluation methodology

The patterns are evaluated according to quality attributes it claims (the same for both patterns). These quality attributes are stated in the filled-in template description of the architectural pattern (according to a shared and agreed template provided by WP7). The stated quality attributes in this architectural pattern description are evaluated using the ATAM approach. Especially the quality attributes utility tree is used to show and analyze the results of the implementation of each quality attribute.

#### 7.2.2.1 Quality attributes provided to WP7 for these patterns

The Assisted Composition Designer and the Semantic Annotation Composition patterns are specified with the work done on WP2. In the context of specification, these patterns have been identified to support certain defined quality attributes, although their main purpose is to support concrete functionalities for users. These quality attributes affect the quality of the architecture where each pattern is integrated. Evaluating these quality attributes permits to know the potential effects in the resulting architecture.

[The significations of plus (+), minus (-) or neutral (0) are given in the specification of this pattern for D7.5]

In this case, only attributes related to those evaluated in the PoC are listed:
Table 22: Assisted Composition Designer Pattern Trade-off

<table>
<thead>
<tr>
<th>Assisted Composition Designer</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Buildability</td>
<td>-</td>
</tr>
<tr>
<td>Modifiability</td>
<td>+</td>
</tr>
<tr>
<td>Performance</td>
<td>-</td>
</tr>
<tr>
<td>Reusability</td>
<td>+</td>
</tr>
<tr>
<td>Integrability</td>
<td>+</td>
</tr>
</tbody>
</table>

Table 23: Semantic Annotation Composition Pattern Trade-off

<table>
<thead>
<tr>
<th>Semantic Annotation Composition</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Buildability</td>
<td>-</td>
</tr>
<tr>
<td>Modifiability</td>
<td>+</td>
</tr>
<tr>
<td>Performance</td>
<td>-</td>
</tr>
<tr>
<td>Reusability</td>
<td>+</td>
</tr>
<tr>
<td>Integrability</td>
<td>+</td>
</tr>
</tbody>
</table>

7.2.2.2 Evaluating quality attributes by using Pattern Quality Attributes Tree

The assessment criteria are partially based on the ATAM methodology [6], which has been adapted to our specific need that is to evaluate, quantify and compare the tradeoffs between architectural choices. The ATAM methodology was originally developed to assist architectural decisions by taking into account early in the design process the quality attributes.

We use the ATAM utility tree for matching the quality attributes (buildability, modifiability and performance) with metrics to be used in the evaluation. The utility tree for the patterns is presented in [D8.1c p51 Figure 17: PoC 13 Assessment Criteria and metrics] and a detailed description may be found in the same deliverable. Other metrics have been used to evaluate the architecture in terms of functional requirements fulfilment and for determining whether the trade-offs are so important with respect to the solution itself. For instance, it is possible that performance is much more affected because of the computational requirements of the solution rather than because of a choice in the architecture.

In order to evaluate some of the quality attributes addressed by the patterns, we have performed several experiments with the SOA4ALL DT Composer, defining different initial workflows and using several composition chunks, all of them based on an ontology created for the Crisis Management domain.

7.2.3 Evaluation results

After carrying out the planned experiments, the metrics defined in the utility tree have been measured. They can be seen in Figure 20.
It is possible to see the complexity of the solution. Even if it has not too many classes, the size of the tool is quite high, as it requires a lot of libraries for operating. Moreover, this amount can be increased if we take into account the amount of knowledge necessary for being able to work with the tool.

In the contrary, modifiability metrics show that it is quite easy to extend the tool in order to increase its functionality.

In the case of ASTRO tools [10], the size of the package is 11.5Mb for the main tools, without including the knowledge base necessary for operating with the tools, containing up to 750 files with code (as it performs more functionalities), although it does not seem as flexible for doing modifications which increase functionalities.

The performance of semiautomatic composition has a strong dependency on the following aspects:

- The complexity of the input model, which can be measured as the number of abstract activities (unbound activities) included in the model.
- The number of agents registered within the semiautomatic composition engine. Those agents introduce changes in the input model, optionally assisted by external helper services: service discovery, reasoner, etc.
- The complexity of the domain knowledge models, measured as the number of concepts, properties, relations, axioms, etc included in the models. Another way to measure this complexity could be the number of RDF statements (triples) included in the RDF repository accessed by the reasoned.
- The number of process fragments and SWS descriptions available in the repositories accessed by the agents and helper services.

Therefore, it is very complex to evaluate properly the performance of semiautomatic composition approach described in this PoC, due to the difficulty to estimate the complexity of those dependencies and their impact in the throughput of the experiments. That is, different experiments will offer quite diverse results.
Nonetheless, from a pure qualitative point of view, it is reasonable easy to estimate roughly the improvement in terms of performance, of a modelling task, using this semiautomatic composition approach compare to the manual modelling approach, since the performance in case of semiautomatic composition is dramatically improved (i.e from some hours to minutes). However, it is difficult to compare this semiautomatic approach with other semiautomatic approaches available since they use completely different techniques.

As some examples, we have executed an experiment for semiautomatic composition using the airport crisis management modelling process, described in this PoC, and another process modelling task, based on one SOA4ALL scenario that models an eGoverment business registration process. In both cases, we run the semiautomatic modelling task 100 times, recording the time required to model the process every 10 tasks. In both cases, the knowledge models are similar in term of concepts (i.e classes) around 100, the number of sub-processes in the repository is around 10-20, the number of SWS around 10-20. They are different in terms of the complexity of input model and subprocesses, since the crisis management process is simpler. Also the number of agents registered in the composition engine is different, since for the modelling of the later experiment we have added an additional agent.

Therefore the time spanned by the experiments are different, being around 1 sec for the former experiment (around 100 sec for the total of 100 tasks) and around 7 seconds for each modelling task in the later experiment.

<table>
<thead>
<tr>
<th>Crisis Management PoC</th>
<th>Business Registration Scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>N# Experiments</strong></td>
<td><strong>N# Experiments</strong></td>
</tr>
<tr>
<td><strong>Time spent (sec)</strong></td>
<td><strong>Time spent (sec)</strong></td>
</tr>
<tr>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>12.5</td>
<td>73.8</td>
</tr>
<tr>
<td>20</td>
<td>20.7</td>
</tr>
<tr>
<td>20.7</td>
<td>138.9</td>
</tr>
<tr>
<td>30</td>
<td>206.1</td>
</tr>
<tr>
<td>29.1</td>
<td>30</td>
</tr>
<tr>
<td>37.5</td>
<td>10</td>
</tr>
<tr>
<td>40</td>
<td>278.1</td>
</tr>
<tr>
<td>46.3</td>
<td>50</td>
</tr>
<tr>
<td>54.3</td>
<td>339.9</td>
</tr>
<tr>
<td>60</td>
<td>397.6</td>
</tr>
<tr>
<td>62.5</td>
<td>60</td>
</tr>
<tr>
<td>70</td>
<td>458.7</td>
</tr>
<tr>
<td>70.6</td>
<td>70</td>
</tr>
<tr>
<td>80</td>
<td>540.9</td>
</tr>
<tr>
<td>79.1</td>
<td>80</td>
</tr>
<tr>
<td>88</td>
<td>615.6</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

*Table 24: Tables of the time spanned by the experiments*

Obviously, these time values depends on the hardware where the experiments are execute and they should be consider only as qualitative measurements that reflect the strong differences in execution time for different modelling tasks. Nonetheless, these absolute execution times demonstrate that the time required for a manual modelling task can be dramatically cut down when the modelling task is assisted by an automatic modelling engine.

In the case of ASTRO WS-Compose Tool [10], a graph based approach, in a bookstore scenario where a complex workflow is generated, it takes, as average, 2.7 sec for the model
construction and 605.2 sec for the synthesizing the composition and providing an executable WS-BPEL file. This case is more comparable with the Business Registration Scenario in SOA4ALL in complexity, although in this case we also obtain an executable file. In any case, it is evident that the amount of time required for these operations are quite high.

Concerning the accuracy of the result model, experiments show that it strongly depends on the following factors:

- Quality of the domain specific knowledge model based (i.e. ontology) light annotations attached to the input process model, which can be used both at process level and at activity level, in order to describe them.

- Quality of the process fragments and SWS descriptions stored by the providers in the repository, which are also based on similar concepts described within the same domain knowledge models.

- Modellers and providers sharing a common understanding of the domain concepts used to annotate the input process model and the modelling artefacts: process fragments and SWS.

In other words, the semiautomatic composition engine applies matchmaking techniques based on light semantics to bind the abstract activities of the input process model to available modelling artefacts such as process fragments and SWS, but that matching strongly depends of how the activities and the modelling artefacts were annotated.

Process fragments and SWS descriptions matchmaking is very sensitive to proper functional classification, I/O descriptions and non functional properties. Good domain knowledge models improve the quality of those descriptions and the way input process models are annotated, thereby the quality of the matchmaking task.

The semiautomatic composition experience could provide better results if it provides feedback to the modeller explaining why some process fragments or SWS were used to resolve abstract activities while others were not. This information could be useful to improve the quality of the knowledge models, process fragments and SWS descriptions and input process model annotations.
7.2.4 Conclusion

The main conclusion after the PoC evaluation is that performance and accuracy of the results are related, although there are many factors affecting the performance of the solution. Using complex models give more accurate results, but more complex models require more computation, giving a higher response time.

As the performance measures show, the influence of the amount of models available, ontology concepts, the complexity of the initial workflow, etc... is so important that trade-offs in performance related to calling an intermediate component are insignificant. The main problem is in the solution itself, as reasoning requires a lot of computation and memory.

Finally, buildability and modifiability are related, although buildability can depend as well in other factors (such as the complexity of the solution, the features provided and the information needed to operate the tools). But it is clear that it is no so difficult to achieve good modifiability levels by doing a bit more complex the design and the initial development.

7.3 Report of Assessment of Requirements (Objectives) of Scenario

The scenario selected for performing the evaluation is the S10: Crisis Management System of Systems. As there is no a concrete scenario which is only focused on the development of service compositions, we have selected a scenario which requires several heterogeneous systems interacting, and which provides a good guidance about the steps performed during the scenario. Those steps are activities performed at runtime which can be modelled by a workflow.

The main requirements found in NEXOF-RA related to service composition are:

- R32 – Orchestration
- L2.5 - Workflows

Related to these, the main functional requirements of the system fulfilled by the patterns should be:

- SR5 - How can a process be realized by composing services?
- SR5.1 - How can processes be designed in terms of the services they are composed of (Orchestration, Choreography descriptions)?
- SR5.1.1 - How can a process be designed to select some of the services it composes at run-time in order to complete and satisfy QoS constraints (Dynamic composition, Dynamic Binding, Constraints Satisfaction)?

In order to check whether these requirements are fulfilled, some check points have been defined:

- Initial Workflow: as a way to determine the main activities required in the workflow
- Goals: as a way to express which kind of results should be obtained executing the workflow
- **Final workflow**: it is the final demonstration that the desired workflow can be developed as an orchestration, which includes all the activities to be performed.
- **Semantic Annotations**: as a mean to enable dynamic composition and dynamic binding during execution, as well as facilitating the modification of the workflow.

These check points will be evaluated in order to determine if the stated requirements can be fulfilled.

### 7.3.1 Link between requirements and quality attributes

As the requirements stated are functional requirements about basic features of a service-centric system, there is no a clear relationship with the quality attributes defined in the patterns.

Accuracy is an aspect which can be considered important in this sense, although it is not considered one of the quality attributes in the patterns. But, as it is related to performance, it has been measured in the experiments, as shown in the previous section.

### 7.3.2 Evaluation of requirements

The mentioned check points have been evaluated in order to determine if the patterns are able to fulfil them. The following table shows the results:

<table>
<thead>
<tr>
<th>SR5 &amp; SR5.1</th>
<th>Initial Workflow</th>
<th>An initial workflow was designed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goals</td>
<td>The workflow was annotated using the defined ontology</td>
<td></td>
</tr>
<tr>
<td>Final Workflow</td>
<td>A final workflow in LPML was obtained</td>
<td></td>
</tr>
</tbody>
</table>

Table 25: Evaluation of requirements of semi-automatic service composition at design time PoC13

In a first stage, the team generated a basic workflow, with high level activities which were annotated with semantic information representing the functionalities expected in each activity. For doing so, the ontology presented in section 4 was applied.

**Figure 53: Initial workflow for the composition process**
Then, the workflow was loaded in SOA4ALL tools, and the DT Composer was called, in order to generate a more complex workflow, which would include sub-tasks whenever necessary, according to the reasoning over the initial activities.

![Diagram of workflow](image)

**Figure 54: Final workflow obtained**

This was achieved thanks to the development of some separate service composition chunks which were used as pieces of workflows and alternatives for some higher level activities, fulfilling concrete functionalities and parameters.

Finally, according to the accuracy measured in the utility tree, it is clear that there are several aspects which influence in the level of fulfilment of users’ requirements. It is possible to obtain wrong workflows, but taking into account the observations given in section 5.2.3, it is easy to obtain accurate results.
7.3.3 Conclusion

The achievement of this pattern is an important step towards the achievement of those requirements related to workflows, orchestrations and the way to obtain them, in order to design and implement processes.

The assessment of functional and system requirements, in conjunction with the accuracy metric obtained before, completes the previous evaluation based on quality attributes. It is demonstrated that it is possible to implement patterns which are oriented to fulfilling these requirements.

It is clear that, even if it is possible to fulfil the requirements, the user satisfaction may not be reached at 100%, but this is because the approaches still depend a lot on human decisions and inputs for the tools, which is something out of scope of the patterns definition.
8 ASSESSMENT OF PoC14: CLOUD MIGRATION ENABLED BY OSGi AND EXEMPLIFIED BY DATABASES - SIE

Siemens assesses a PoC called PoC14: Cloud migration enabled by OSGi and exemplified by databases. The PoC demonstrates patterns / techniques (as, for instance, Dependency Injection, Adapters) which are principally used in the Enterprise domain and applied to problems/challenges in the cloud computing domain. The PoC validates basic principles for cloud-agnostic middleware and shows how applications can be enabled to be transparently migrated into a (vendor-specific) cloud respectively, from one vendor’s cloud to the cloud of another vendor without any changes to their business logic. The PoC shows all these issues by concentrating on migrating a local database to a cloud database.

8.1 PoC’s selection

This PoC which has been selected and setup is based on a suggestion coming from WP3 (Siemens) and on Siemens’ PoC I story (PoCI: SCA Example Motion Tracker, Pattern: OSGi-based SCA Container) (see D8.1).

The pattern evaluated in the PoC is called Cloud migration enabled by OSGi - step one.

8.2 Pattern: Cloud migration enabled by OSGi - step one

Beyond the Enterprise domain, customers and service providers see the need and advantages to bring applications and services into the cloud. Main motivation for the PoC is to provide a standard way to access cloud resources (not restricted to storage) for supporting migration into cloud and between clouds. Currently, many activities and initiatives to evolve respective standards have started. Each cloud stack layer, each stakeholder and each technology demands different requirements concerning standardization. Cloud Services will evolve more and more, driven by use cases. The APIs and the management platforms provided by different vendors differ very much. So, currently it is not clear, what they have really in common for becoming a standard. It is expected that there will be many of them.

Even if it is clear which standards will be the established ones, implementation of standards for interfaces and protocols or adapter for various interfaces or protocol can only support interoperability, but it is no guaranty. To avoid cloud vendor locking by bringing applications and services into the cloud adaptation to new operating environments (portability), Extension of capability and Interoperability are the non-functional aspects covered by the pattern.

8.2.1 Scope of the PoC with respect the Reference Architecture

From the PoC the pattern “Cloud migration enabled by OSGi – step one” is derived. This pattern is actually part of D7.4. It is an Abstract Design Pattern and refines the Top-Level Pattern “Enterprise SOA” as well the Top-Level Pattern PaaS.
Before describing all elements, here is the relationship between abstract components and concrete components.

<table>
<thead>
<tr>
<th>Abstract component</th>
<th>Concrete component</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Services</td>
<td>AXIS 2</td>
</tr>
<tr>
<td>OSGi container</td>
<td>Eclipse Equinox OSGi, Spring</td>
</tr>
<tr>
<td>OSGi components</td>
<td>Spring DI, openJPA, SimpleJPA (bundles)</td>
</tr>
<tr>
<td>Cloud Database</td>
<td>Amazon SimpleDB</td>
</tr>
<tr>
<td>Relational Database</td>
<td>MySQL</td>
</tr>
</tbody>
</table>

**Table 26: Components of the scope of Pattern Cloud migration enabled by OSGi**

The achievement of the PoC has used MySQL as examples for the abstract component relational database and Amazon SimpleDB as examples for the abstract component for cloud database. The results of the PoC and the pattern are applicable in the same way for implementation using to other examples for these two concrete.

In the PoC the involved pattern is Cloud migration enabled by OSGi – step one. This pattern is based on common APIs and an OSGi based service platform using dependency injection. It enables migration of services into cloud and between clouds and shows one fundamental step towards an OSGi Based Cloud Platform Runtime. The pattern enables access to and use of a local as well as a cloud service. As an example service the functionalities of a database are described, but data migration between databases is disregarded.
With regard to the standards catalogue used, therefore the main standard is OSGi (Open Services Gateway initiative). The Open Services Gateway Initiative (OSGi) is a component framework for Java and defines architecture for developing and deploying modular applications and libraries. It provides a dependency resolution mechanism, with version support. It also offers standardized ways to manage the software lifecycle.

As an example service the functionalities of a database are described. Therefore additional used standards are Structured Query Language (SQL), Java Data Objects (JDO) API and Java Persistence API (JPA). SQL is a database sublanguage that is used for accessing relational databases and is standardized by ISO. JDO is a Java application program interface (API) for transparent persistence. It works with both, object and relational databases as well as other types of systems. The Java Persistence API (JPA) is a Java programming language framework. It allows managing relational data in applications and to access relational databases. JPA 2.0 is also known as JSR 317. Both, JDO and JPA are standardized by the Sun Java Community Process. The Open Cloud Computing Interface (OCCI by OGF) is the related important standard which is just evolving and therefore it is listed here, too.

The pattern is related to the abstract components OSGi container, web service, message queue, relational database and cloud data base.

This pattern has been implemented with the following concrete components, which are here referenced with the relationship to the abstract components:

- **Axis 2**
  Apache Axis is an open source, XML based Web service framework. Axis 2 Axis consists of a implementation of the SOAP server, and various utilities and APIs for generating and deploying Web service applications. Axis is developed under the auspices of the Apache Software Foundation. (http://ws.apache.org/axis/cpp/index.html)

- **Apache Active MQ**
  This is an open source implementation of the JMS standard. (http://activemq.apache.org/) and has been used for the abstract component Message Queue.

- **Eclipse Equinox OSGi**
  Eclipse Equinox OSGi provides a certified implementation of the OSGi R4 core framework specification. It is open source. (http://www.eclipse.org/equinox/). Equinox has been used with additional bundles to enable logging (slf4j), dependency injection (Spring framework), database access (Java Persistence API JPA) and to access different databases (openJPA and SimpleJPA).
  The Eclipse Equinox OSGi has been used for the abstract component OSGi Container.

- **Spring**
  It is an application framework for Java and JavaEE, and offers configuration via Dependency Injection; declarative services via AOP; and packaged enterprise services. Spring in combination with JPA provides a feature which enables the usage of different datasources. Spring DI and openJPA, both as OSGi bundles, are representatives for the abstract component OSGi component.
- MySQL
  This is an open source relational database (see [http://www.mysql.com/](http://www.mysql.com/)). In the context of the PoC it represents the other local relational database (local service). MySQL has been used for the abstract component Relational Database.

- Amazon SimpleDB
  Amazon SimpleDB as example for any other cloud database has been used for the abstract component cloud database.

The following figure gives a short overview how the implementation of the PoC is structured: OSGi cloud platform (built by OSGi, Spring DI and JPA bundles), application (a simple web service accessing data sources) and datasources (MySQL, Amazon Simple DB and potential others).

8.2.2 Evaluation methodology

The pattern is evaluated according to quality attributes it claims. These quality attributes are stated in the filled-in template description of the architectural pattern (according to a shared and agreed template provided by WP7). The stated quality attributes in this architectural pattern description are evaluated using the ATAM approach. Especially the quality attributes utility tree is used to show and analyze the results of the implementation of each quality attribute.

Figure 56 Structure diagram showing the PoC implementation
8.2.2.1 Quality attributes provided to WP7 for this pattern

The specification of the pattern “Cloud migration enabled by OSGi - step one” is based on a proposal coming from WP3 (Siemens) and on Siemens’ PoC phase I story (PoC I: SCA Example Motion Tracker, Pattern: OSGi-based SCA Container). In the context of specification, this pattern has been identified to support certain defined quality attributes. These quality attributes affect the quality of the architecture where each pattern is integrated. Evaluating these quality attributes permit to know if the resulting architecture where the pattern has been applied meets the expected requirements.

[The significations of plus (+), minus (-) or neutral (0) are given in the specification of this pattern for D7.5]

<table>
<thead>
<tr>
<th>Quality attribute</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adaptation to new operating environments (Portability)</td>
<td>+</td>
</tr>
<tr>
<td>Extension of capability</td>
<td>+</td>
</tr>
<tr>
<td>Interoperability</td>
<td>+</td>
</tr>
</tbody>
</table>

Table 27: Cloud migration enabled by OSGi Pattern Trade-off

8.2.2.2 Evaluating quality attributes by using Pattern Quality Attributes Tree

The method was prototyping with parallel expert estimations of the given statements. Interoperability, service portability and extension of capability were validated by building and executing the related migration scenario (WP10). The assessment metric is qualitative by nature, i.e. whether the selected features are fully available and provided in a feasible way by implementation and demonstrating the migration into the cloud.
Please report to the Annexe 1 for the details of using ATAM approach. The following description is just a report of results of the analysis.

To guarantee the interoperability with different database services, standard database APIs are used. The Java Persistence API (JPA) is a widely used standard for database access. It is a Java programming language framework which allows managing relational data in applications and to access relational databases. JPA 2.0 is also known as JSR 317. openJPA and SimpleJPA are used as technical implementation of JPA.

Spring is a data access framework which comprises several modules that provide a range of services. In the context of this pattern the modules for inversion of control (Spring Dependency Injection DI) and data access (JPA) are used. Spring in combination with JPA allows interoperable integration with relational databases.

To facilitate portability of software components, the Open Services Gateway Initiative (OSGi) is a component framework for Java is used. It defines architecture for developing and deploying modular applications and libraries. It provides a dependency resolution mechanism, with version support. It also offers standardized ways to manage the software lifecycle. It builds on open standards such as Web services. Eclipse Equinox OSGi is used as technical solution.

To provide a means to dynamically extend the service runtime OSGi in combination with Spring DI (as OSGi component or bundle) is used. Spring DI allows to deploy and undeploy components during runtime and make the underlying SOA Framework (here: Eclipse Equinox OSGi) dynamically extendible.

8.2.3 Evaluation results

The goal of the PoC is to demonstrate the feasibility of the objectives by means of a running and working implementation. For the evaluation of the quality attributes interoperability, service portability and Extension of capability, which are identified in the ATAM’s utility tree.
(See D8.1) we used the method of feasibility prototyping: Walk-through of technical concepts with working example according PoC scenario. The evaluation results are subjective. Quantitative indicators are not applicable in this context/field. The PoC shows how applications can be enabled to be transparently migrated into a (vendor-specific) cloud respectively, from one vendor's cloud to the cloud of another vendor without any changes to their business logic. One motivation for migration into cloud is cost reduction (operational costs). The PoC aims on providing flexibility to do business with a new provider without excessive migration effort or cost. Therefore evaluation is done with respect to two main issues: portability and migration cost. The trade-off is between operational costs and migration costs.

**Determination of Sensitivity Points: Operational costs and migration costs**

In the context of the PoC two sensitivity points operational costs and migration costs are discussed. The operational costs depend on the service hosting: It can be hosted on on-premises platform or hosted in a (vendor-specific) cloud provided by an infrastructure provider (IaaS provider). In case of a cloud infrastructure service is used, the operational costs depend on the IaaS business and prize model which is offered by the IaaS provider. Cloud computing is an approach that promises to be highly agile and lower costs for consumers of cloud services.

Over time, switching from on-premises to cloud hosting can result in a dramatically reduction of operational costs. Especially, if the service owner is not tied to one IaaS provider (no vendor lock in), e.g. by contract or technology dependencies, the costs can be reduced further more by accepting better offer from another provider and eventually changing to an alternate vendor-specific cloud platform.
Migration costs depend on migration strategy and on portability and integrability of the service. Extension of capability facilitates migration by allowing to equip the service with appropriate adapters. Migration costs are much lower if the business logic is separated from infrastructure logic (which is the case for the PoC), because only one part of the service, the infrastructure logic needs to be migrated. More and more infrastructure providers offer such infrastructure logic to their customers, so that a migration is made easy. Besides, in a great number of cases infrastructure logic consists of much less code than the business logic.

Achieving separation of business logic and infrastructure logic is not for free. The PoC proposes to use OSGi and Spring to facilitate this separation. However, using such technologies requires experience and practice and this means for OSGi/Spring beginners a rather high training efforts. These follow a classical learning curve which is depicted in Figure 61. For experts the learning curve is considerable lower.
The combination of both aspects results in the complete migration costs for the PoC. Of course for each service to be migrated and its owner the costs need to be individually estimated.

**Determination of trade-off point of Costs**

Migration costs influence substantially the gain that can be expected by bringing a service into the cloud. The analysis of both, migration costs and operational costs determines the decision for any platform migration. The intersection point of the two respective curves identifies the trade-off point.

It doesn't need mentioning, that for each service migration the trade-off analysis needs to be done individually, in particular with respect to the complexity of the service and experiences of applied migration technologies.
8.2.4 Conclusion

Spring, JPA and OSGi are converging to create an environment that facilitates the design and the lifecycle management of software components. These technologies are principally used in the Enterprise domain and applied to problems/challenges in the cloud computing domain. They are exposed as reusable services. The combination of OSGi and Spring has an impact on service-oriented design and application development, in that it makes the service runtime modular by itself allowing it to be dynamically extended by new adapters (like openJPA or SimpleJPA) and via Spring DI.

Spring can complement OSGi to make it an even more attractive platform by avoiding dependencies (dependency injection and inversion of control) and providing an integration platform for several technologies (e.g. AOP, Persistence, Remoting). In the context of the PoC, it provides a flexible switching between several data sources and supports scenarios where interworking services are hosted on different platforms e.g. a service containing business logic is hosted on-premises and an infrastructure service is hosted in the cloud.

- OSGi, openJPA and Spring are really powerful, but complex. It is recommended to familiarise oneself with these projects and gain experiences before using them productively. The effort for familiarisation should not be underestimated.
- Once openJPA is running as bundle, the handling is very comfortable. It is easy to switch between different JPA implementations and to switch between different data sources.
- The combination of OSGi and Spring absolutely makes sense. It brings together the benefits of the Spring Framework and the OSGi runtime technology by integrating the OSGi modularity and service models with the Spring bean model and allows to realize bundle-based Spring context definitions and dependency injection across bundles within the OSGi runtime and components.
- openJPA supports more than 15 different databases which can be configured within minutes. So interoperability with different databases is provided.
- By using Spring it is very easy to write applications which use different databases. OpenJPA automatically creates the table with the required fields in the mySQL database.
8.3 Report of Assessment of Requirements (Objectives) of Scenario

This PoC is demonstrated via the scenario “Cloud Computing: Migration into a cloud and migration between different clouds” [come from S18 of D10.1].

It focuses on the following requirements (objectives) to be evaluated:

- R19 - Interoperability (R19)
- L5.2 - Service portability (L5.2)
- L1.11 - Extension of capability (buildability)

The PoC14 Cloud migration enabled by OSGi and exemplified by databases is based on a proposal coming from WP3 (Siemens) and on Siemens’ PoC I story (PoC6: SCA Example Motion Tracker, Pattern: OSGi-based SCA Container, reported in D8.1b). Although the pattern “Cloud Migration Enabled by OSGi – step one” is not produced within WP3, it supports some of the objectives for interoperability and portability between infrastructure providers. The PoC is linked to the WP10 scenario “Cloud Computing: Migration into a cloud and migration between different clouds”. The pattern “Cloud migration enabled by OSGi - step one” addresses the requirement Technical interoperability (R19) as stated in D10.1. It is applicable to the cloud domain even is not particularly formulated to be cloud-specific.

8.3.1 Link between requirements and quality attributes

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Is linked to Quality Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Migration into Cloud/between clouds</td>
<td>Adaptation to new operating environments (Portability)</td>
</tr>
<tr>
<td></td>
<td>Extension of capability</td>
</tr>
<tr>
<td></td>
<td>Interoperability</td>
</tr>
</tbody>
</table>

Table 28: Link between requirement and quality attributes

8.3.2 Evaluation of requirements

The requirements are exposed in certain check points. These check points must be verified by the implementation. The implementation parts, involved in the verification of the check points are shown in the overview of the PoC’s implementation (see Figure 56).

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Check point</th>
<th>How to verify by the implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Migration into Cloud/between clouds</td>
<td>Check point 1:Infrastructure application (it is a simple web application) interworks with local database as well with cloud</td>
<td>Execute migration into cloud: Via OSGi management console switch from used database MySQL to cloud database Amazon SimpleDB (and vice...</td>
</tr>
</tbody>
</table>
database

versa), the simple web service lists content of the database, adds some new data and lists the content again, the database switch is executed, again before and after the switch, the simple web service lists content of the database, adds some new data and lists the content again.

| Check point 2: Reconfiguration | The web interface of the application shows content of Amazon simple DB which is the new used cloud database. |
| Check point 3: Extension | OSGi Log makes in evidence that extension of capability is successful. |

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Check point</th>
<th>Mechanisms / Components</th>
<th>Control means</th>
</tr>
</thead>
<tbody>
<tr>
<td>Migration into Cloud/between clouds</td>
<td>Infrastructure</td>
<td>Exchange DB</td>
<td>Access DB via service</td>
</tr>
<tr>
<td>Reconfiguration</td>
<td>Reconfigure Cloud DB</td>
<td>View content of Cloud DB</td>
<td></td>
</tr>
<tr>
<td>Extension</td>
<td>Extend with new Bundle</td>
<td>OSGi Log makes in evidence that extension of capability is successful.</td>
<td></td>
</tr>
</tbody>
</table>

Table 29: Evaluation of requirements of cloud migration enabled by OSGi and exemplified by databases PoC14

In the following table the elements for the ATAM QA Tree are shown, which is depict in Figure 63.

Table 30: Evaluation of requirements of cloud migration enabled by OSGi and exemplified by databases PoC14 (next)
8.3.3 Conclusion

The achievement of the migration pattern is one step towards cloud migration enabled by OSGi. It shows basic principles for cloud-agnostic middleware and how applications can be enabled to be transparently migrated into a (vendor-specific) cloud, respectively, from one vendor’s cloud to the cloud of another vendor without any changes to their business logic. The PoC shows all these issues by concentrating on migrating a local database to a cloud database.

As already mentioned also other solutions for migration support are available, but they are focusing more or less on adapter concepts. Important requirements, such as management and performance issues, are up to now not sufficiently considered. There are also many activities and initiatives for evolving standards, all in this context. Unfortunately, it has to be assumed that there will be many of them. This rises some doubt on whether they can facilitate the migration challenges to the full extend.

Based on OSGi and Spring and the results of the evaluated PoC and pattern complementary architectural patterns for migration are expected to evolve with focus on adaption to new operating environments (portability), extension of capability, interoperability, management and performance issues, which as well, will cope with the emerging number of standards.
9 PRESENTATION OF RESULTS OF THE EVALUATION

The presentation of the overall results after the assessment based on the ATAM approach shows a global view with important features. The way to reach these features is detailed in the sections: “Evaluation methodology, Evaluation results and conclusion”. The objective of the following table goes to the essential.

The following table [Table 31] permits to understand the table of results [Table 32] and contains three boxes (addressed four main points: evaluation point, sensitivity trade-off points, risk, and added value):

- 1st box concerns the evaluation type,
- 2nd box dedicated to the results obtained with the sensitivity and trade-off points,
- 3rd box is divided in two boxes which are dedicated to the risk and to the added value.

<table>
<thead>
<tr>
<th>Evaluation type: Test, Benchmark, Subjectivity, Analytical.</th>
<th>There are many types of the evaluation. The evaluation can be objective or subjective.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The evaluation objectivity (EO) is used to measure the quality attributes. This measure is only possible to apply on the countable things. The measure can be:</td>
</tr>
<tr>
<td></td>
<td>• Test,</td>
</tr>
<tr>
<td></td>
<td>• Benchmark from the manufacturer,</td>
</tr>
<tr>
<td></td>
<td>• Analytical.</td>
</tr>
<tr>
<td></td>
<td>The evaluation subjectivity (ES) is used to give a point of view on something that we cannot measure with a rule.</td>
</tr>
<tr>
<td></td>
<td>• Subjectivity on the uncountable things (such as the confidence on the trusted third party). The reason of why these things cannot be measurable is given in details in the corresponding assessment sections.</td>
</tr>
<tr>
<td>Sensitivity and trade-off points (Y,Y)</td>
<td>Sensitivity Points: Key architectural decision that is critical for achieving a particular QA requirement. Serve as yellow flags: “Use caution when changing this property of the architecture”),</td>
</tr>
</tbody>
</table>
Trade-off Points: are the key architectural decision that affects more than one QA and is a sensitivity point for more than one quality attribute, are the most critical decisions that one can make in architecture.

The answer can be “Yes” or “No”. The 1st Y concerns the Sensitivity point, the 2nd Y for the Trade-off point.

The “Y” means that the sensitivity or trade-off point is found for the quality attribute. This judgement is based on the previous evaluation (objectivity or subjectivity).

The “N” means that it cannot be determined.

<table>
<thead>
<tr>
<th>Risk (H, M, L)</th>
<th>Added value (H, M, L)</th>
<th>Risk:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>This element concerns the implementation of the quality attribute.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Risks (High) and Non-Risks (Low)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Risks: Potentially problematic architectural decisions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Non-risks: Good architectural decisions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Medium: Difficult to state if there is a risk or not.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added value:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The importance of the quality attribute is judged High when the implementation of it gives a high added value to the architecture.</td>
</tr>
</tbody>
</table>

Table 31: How to read the table of results
The following table sums all results obtained for the patterns. Refer to [Table 31] for the aid of reading this table.

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Quality Attribute</th>
<th>Performance</th>
<th>Availability</th>
<th>Scalability</th>
<th>Interoperability</th>
<th>Privacy</th>
<th>Accountability</th>
<th>Buildability</th>
<th>Modifiability</th>
<th>Reliability</th>
<th>Maintainability</th>
<th>Adaptability</th>
<th>Integrability</th>
<th>Portability</th>
<th>Recoverability</th>
<th>Testability</th>
<th>Reusability</th>
<th>Resource efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not only SQL (No SQL) Storage</td>
<td></td>
<td>Test</td>
<td>Test</td>
<td>Test</td>
<td>(Y, Y)</td>
<td>(Y, Y)</td>
<td>(Y, Y)</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td>Authorization</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Sub</td>
<td>(Y, Y)</td>
<td>(Y, Y)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service Discovery</td>
<td></td>
<td>Test</td>
<td></td>
<td>Test</td>
<td>Test</td>
<td>Test</td>
<td>(Y, Y)</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td>Service Matching and Ranking</td>
<td></td>
<td>Test</td>
<td></td>
<td>Test</td>
<td>Test</td>
<td>Test</td>
<td>(Y, Y)</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td>Monitoring in Enterprise SOA</td>
<td></td>
<td>Test</td>
<td>Test</td>
<td>Test</td>
<td>(Y, Y)</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>H</td>
<td>M</td>
<td>M</td>
<td>H</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td></td>
</tr>
</tbody>
</table>
Table 32: Results of evaluation with ATAM approach or other approaches

The following section gives us a concrete translation of this table.
Example of the Performance for the Not Only SQL pattern (MoMa)

<table>
<thead>
<tr>
<th>Evaluation type</th>
<th>Test,</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity and</td>
<td>The perceived latency is sensitive to replication. Replication has an</td>
</tr>
<tr>
<td>trade-off points</td>
<td>overhead in particular in write operations. In particular, extra work</td>
</tr>
<tr>
<td>(Y,Y)</td>
<td>to replicate data (mainly in the approach we adopted that waits until a</td>
</tr>
<tr>
<td></td>
<td>certain number of replicas answers before concluding the write operation)</td>
</tr>
<tr>
<td></td>
<td>hurts performance. By the other side, replicas allows spreading</td>
</tr>
<tr>
<td></td>
<td>operations among more nodes. This aspect, in our observations,</td>
</tr>
<tr>
<td></td>
<td>positively impacts in case of read operations.</td>
</tr>
</tbody>
</table>

And the trade-off point:

![Performance vs Availability Graph]

Report to Figure 7

<table>
<thead>
<tr>
<th>Risk: L</th>
<th>Added value H</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Risk is considered low. It means that the implementation of this quality attribute is easy.</td>
</tr>
<tr>
<td>Added value is qualified high. In fact, it’s worth to implement this quality attribute. The high availability for data is a target to be obtained by this pattern.</td>
<td></td>
</tr>
</tbody>
</table>

Table 33: Example of the Performance for the Not Only SQL pattern (MoMa)
Example of the Privacy of Authorization pattern (Thales)

<table>
<thead>
<tr>
<th>Evaluation type</th>
<th>Subjectivity on the uncountable things (such as the confidence on the cryptographic key). The reason of why these things cannot be measurable is given in details in the corresponding assessment sections.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity and trade-off points (Y,Y)</td>
<td>Here is the picture Figure 13 which shows the sensitivity point: Quality Attribute</td>
</tr>
<tr>
<td>Decision</td>
<td>Lost of confidence of the TTP or the cryptographic key</td>
</tr>
<tr>
<td>Trade-off point Figure 12:</td>
<td>Asymmetric key</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Risk: L</th>
<th>Added value H</th>
<th>Risk is considered low. It means that the implementation of this quality attribute is easy.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Added value is qualified high. In fact, it's worth to implement this quality attribute. The integrity is a target to be obtained by this pattern.</td>
</tr>
</tbody>
</table>

Table 34: Example of the Privacy of Authorization pattern (Thales)
Example of the Performance for Monitoring in Enterprise SOA pattern (CINI)

<table>
<thead>
<tr>
<th>Evaluation type</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity and trade-off points (Y, Y)</td>
<td>The sensitivity point of Figure 30:</td>
</tr>
<tr>
<td></td>
<td><img src="image1" alt="Graph" /></td>
</tr>
<tr>
<td></td>
<td>And the trade-off point of Figure 31:</td>
</tr>
<tr>
<td></td>
<td><img src="image2" alt="Graph" /></td>
</tr>
</tbody>
</table>

**Report to**

- **Risk:** M
- **Added value:** H

Risk is considered Middle. It means that the implementation of this quality attribute is a little difficult.

Added value is qualified high. In fact, it’s worth to implement this quality attribute. The high availability for data is a target to be obtained by this pattern.

Table 35: Example of the Performance for Monitoring in Enterprise SOA pattern (CINI)
Example of the determination of Sensitivity Points: Operational costs and migration costs (Siemens)

<table>
<thead>
<tr>
<th>Evaluation type</th>
<th>Subjectivity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The method was prototyping with parallel expert estimations of the given statements. Interoperability and portability were validated by building and executing the integration scenarios. The assessment metric is qualitative by nature, i.e. whether the selected features are fully available and provided in a feasible way by SCA implementation. (The PoC, from which the pattern is derived, used the TUSCANY SCA container.)</td>
</tr>
</tbody>
</table>

Sensitivity and trade-off points (Y,Y)

The sensitivity point of Figure 58 and Figure 61

![Graph showing Operational Costs per Use vs. Intensity of Use with On-premises: more or less fix-costs and Cloud: pay-per-use-model]

And the trade-off point of Figure 62

![Graph showing Complexity vs. Migration Costs with Learning curve for OSGi/Spring, experts vs. beginners]
Apache Tuscany is used as technical solution. Of course, any other SoA Framework which provides an OSGI-based implementation can be chosen to implement the pattern. The risk lies in this selection: availability of such a technical solution and its feature completeness (which features really work) and if the features are provided in a feasible way.

Added value:
To provide a means to dynamically extend the service runtime a modular framework is used as the architectural baseground for the SCA-compliant service runtime. The Open Services Gateway Initiative (OSGi) is a component framework for Java and defines architecture for developing and deploying modular applications and libraries.

Table 36: Example of the determination of Sensitivity Points: Operational costs and migration costs (Siemens)
10 CONCLUSION

For PoC Phase II a number of new architectural patterns have been assessed through PoCs still promoting the use of ATAM methodology to have it done. As for PoC Phase I, architectural patterns demonstrated through PoCs were assessed according to Quality Attributes they claimed thus meeting the demand coming from the Architecture team (WP7). As a result of the teaming engaged for PoC Phase II, between WP8 and WP10 additional reporting was also provided for what concerns assessment of the (functional) requirements of the scenarios selected for the PoCs (at least the ones demonstrated through PoCs). This has contributed to better align the work of the two WPs while extending the assessment coverage and thus reporting. In view of the work performed and the spectrum (E-SOA, IoS, Cloud) which is now one of the patterns WP8 managed to demonstrate and assess through PoCs, it is clear that WP8 contributed valuable and actionable results to WP7 in order for it to make use of them in the context of reference architecture specifications for NEXOF. This without mentioning the value of the results reported here for what concerns the work undertaken by WP14 in the context of Instantiation Guidelines.
REFERENCES


[4] [Chorus]: Decision of the Group Operations committee to set up a common Group Reference System


GLOSSARY

**Amazon**: is an American-based multinational electronic commerce company. Headquartered in Seattle, Washington, it is America's largest online retailer, with nearly three times the Internet sales revenue of the runner up, Staples, Inc., as of January 2010.[3]


**Esynaps**: eSynaps Search web service

[http://www.webservicelist.com/api/web_services_programming_development_esynaps_search_api_websitedesign37744.html](http://www.webservicelist.com/api/web_services_programming_development_esynaps_search_api_websitedesign37744.html)

**Facebook**: is a social networking website launched in February 2004 and operated and privately owned by Facebook. Users can add people as friends and send them messages, and update their personal profiles to notify friends about themselves. Additionally, users can join networks organized by workplace, school, or college. The website's name stems from the colloquial name of books given to students at the start of the academic year by university administrations in the US with the intention of helping students to get to know each other better. Anyone age 13 or older can become a Facebook user.

For more information [http://en.wikipedia.org/wiki/Facebook](http://en.wikipedia.org/wiki/Facebook)

**Google**: is a multinational public cloud computing, Internet search, and advertising technologies corporation.


**IoS**: Internet of Services, pattern specified in the WP7


**Non-repudiation** is the concept of ensuring that a party in a dispute cannot repudiate, or refute the validity of a statement or contract. Although this concept can be applied to any transmission, including television and radio, by far the most common application is in the verification and trust of signatures

**Ontology**: In computer science and information science, an ontology is a formal representation of the knowledge by a set of concepts within a domain and the relationships between those concepts. It is used to reason about the properties of that domain, and may be used to describe the domain.

In theory, an ontology is a "formal, explicit specification of a shared conceptualisation".[1] An ontology provides a shared vocabulary, which can be used to model a domain — that is, the type of objects and/or concepts that exist, and their properties and relations.


**OSGi framework** is a module system and service platform for the Java programming language that implements a complete and dynamic component model, something that does not exist in standalone Java/VM environments. Applications or components (coming in the form of bundles for deployment) can be remotely installed, started, stopped, updated and uninstalled without requiring a reboot; management of Java packages/classes is specified in great detail. Life cycle management is done via APIs which allow for remote downloading of management policies. The service registry allows bundles to detect the addition of new services, or the removal of services, and adapt accordingly.
OWL-S Web Ontology Language (OWL)
http://www.w3.org/2004/OWL/

PaaS: Platform as a Service, pattern specified in the WP7

PKCS refers to a group of Public Key Cryptography Standards devised and published by RSA Security.

Public Key Infrastructure (PKI): is a set of hardware, software, people, policies, and procedures needed to create, manage, store, distribute, and revoke digital certificates.

QoS: Quality of Service

RSA Data Security Inc was assigned the licensing rights for the patent (which expired in 2000) on the RSA asymmetric key algorithm and acquired the licensing rights to several other key patents as well (e.g., the Schnorr patent). As such, RSA Security, and its research division, RSA Labs, were interested in promoting and facilitating the use of public-key techniques. To that end, they developed the PKCS standards. They retained control over them, announcing that they would make changes/improvements as they deemed necessary, and so the PKCS standards were not, in a significant sense, actual industry standards, despite the name. Some, but not all, have in recent years begun to move into 'standards track' processes with one or more of the standards organizations (notably, the IETF PKIX working group).

Reasoners
The WSML2Reasoner framework has a flexible architecture that allows the easy integration of existing reasoning components. It first maps WSML to either generic Datalog or OWL (Web Ontology Language), and then uses a plug-in layer that translates to the specific internal representation of a single reasoning engine. In the following sections, we briefly discuss a few of the reasoning engines currently integrated and tested within the WSML2Reasoner framework.
For more information http://tools.sti-innsbruck.at/wsml2reasoner/

REST: Representational State Transfer (REST) is a style of software architecture for distributed hypermedia systems such as the World Wide Web. The term Representational State Transfer was introduced and defined in 2000 by Roy Fielding[1][2] in his doctoral dissertation. Fielding is one of the principal authors of the Hypertext Transfer Protocol (HTTP) specification versions 1.0 and 1.1.
http://en.wikipedia.org/wiki/Representational_State_Transfer

Seekda: is an Austrian spin-off company[1] that specializes in developing and marketing Web Services oriented products, particularly a Web based search engine allowing to locate public Web Services and innovative IT solutions for the eTourism sector.
http://en.wikipedia.org/wiki/Seekda

SOA4All: Service Oriented Architectures for All (SOA4All) is a Large-Scale Integrating Project funded by the European Seventh Framework Programme, under the Service and Software Architectures, Infrastructures and Engineering research area.
For more information: http://www.soa4all.eu/
**Spring**: Spring, the leading platform to build and run enterprise Java applications. Led and sustained by SpringSource, Spring delivers significant benefits for many projects, increasing development productivity and runtime performance while improving test coverage and application quality.

For more information [http://www.springsource.org/](http://www.springsource.org/)

**Timestamping Authority (TSA)** can resemble to Certificate Authority (CA). TSA certifies the timestamp on a document. TSA plays a role of third confident party. The timestamp put on the document can be revoked if you want to verify the exactitude of this timestamp.

**Trusted timestamping**: is the process of securely keeping track of the creation and modification time of a document. Security here means that no one, not even the owner of the document, should be able to change it once it has been recorded provided that the timestamper's integrity is never compromised.

**TTP**: Trusted Third Party.

**UUDI**: Universal Description, Discovery and Integration (UDDI, pronounced Yu-di:) is a platform-independent, Extensible Markup Language (XML)-based registry for businesses worldwide to list themselves on the Internet. UDDI is an open industry initiative, sponsored by the Organization for the Advancement of Structured Information Standards (OASIS), enabling businesses to publish service listings and discover each other and define how the services or software applications interact over the Internet. A UDDI business registration consists of three components:


**WSDL**: Web Services Description Language (WSDL, pronounced 'wiz-del') is an XML-based language that provides a model for describing Web services. The meaning of the acronym has changed from version 1.1 where the D stood for Definition.

For more information: [http://en.wikipedia.org/wiki/Web_Services_Description_Language](http://en.wikipedia.org/wiki/Web_Services_Description_Language)

**WSML2Reasoner** is a highly modular framework that combines various validation, normalization and transformation algorithms that enable the translation of ontology descriptions in WSML to the appropriate syntax of several underlying reasoning engines.

**WSMO** or Web Service Modeling Ontology is a conceptual model for relevant aspects related to Semantic Web Services. It provides an ontology based framework, which supports the deployment and interoperability of Semantic Web Services.

The WSMO has four main components:

- **Goals** - The client's objectives when consulting a Web Service.
- **Ontologies** - A formal Semantic description of the information used by all other components.
- **Mediators** - Connectors between componentes with mediation facilities. Provides interoperability between different ontologies.
- **WebServices** - Semantic description of Web Services may include functional (Capability) and usage (Interface) descriptions.
D1.2.1 deliverable provides a grounding mechanism that will be used in SOA4ALL WSMO-based service descriptions.

For more information: http://www.soa4all.eu/component/remository/?func=fileinfo&id=14

**WSMX 1.0: A Further Step Toward a Complete Semantic Execution**

For more information: http://www.soa4all.eu/component/remository/?func=fileinfo&id=146
ANNEXE 1

Description of the methodology
The Trusted Timestamping is the pattern which has already specified in the WP7. This pattern is influenced positively or negatively by certain quality attributes that it should support.

The + means that the pattern is sufficiently supports the quality attributes (so that the architectural choices in the pattern are valid with respect to this attribute.)

The - that the pattern is insufficiently supports the quality attributes (so that the architectural choices in the pattern are invalid with respect to this attribute.)

The next boxes are the quality attributes.

In the next step, the asking to the following questions “what or when”: what is used to achieve the quality attribute? For example, to guarantee the integrity, a protocol is used.

(H,L): H means that the achievement of this quality attribute brings a high added value to the architecture of the pattern. In another term, the importance of the attribute contributes to the success of the system.
The second letter “L” means that the setup or configuration is more easy (low). It also means the risk to achieve this attribute (level of complexity). These qualifiers can take L (low), M (medium), H(high).

The achievement of this quality attribute by using a protocol, the Digital Signature is the mechanism which can guarantee the integrity implementation. The X509 and 1024 bits are the technical solutions of the Digital Signature.