NEXOF-RA

NESSI Open Framework – Reference Architecture

IST- FP7-216446

Deliverable D8.0.c

Processes, Principles and Selection Criteria behind PoC

Authors

Angelo Gaeta (MoMA)

Due date of deliverable: 30/06/2010

Actual submission date: 30/06/2010

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

Change History

Version	Date	Status	Author (Partner)	Description
D8.0.c V0.1	19/04/2010	Draft	MoMA	Re-worked version according reviewers comments and recommendations
D8.0.c V0.2	28/04/2010	Draft	MoMA	Update
D8.0.c V0.3	30/04/2010	Draft	MoMA	Update
D8.0.c V0.4	05/05/2010	Draft	MoMA	Update
D8.0.c V0.5	07/05/2010	Draft	MoMA	Update
D8.0.c V0.6	10/05/2010	Draft	MoMA	Update
D8.0.c V0.7	14/05/2010	Draft	MoMA	Update according to the plenary meeting comments
D8.0.c V0.8	26/05/2010	Draft	MoMA	First Stable version
D8.0.c V0.9	11/06/2010	Final	MoMA	Version for Internal Review
D8.0.c V1.0	28/06/2010	Final	MoMA	Addressed internal reviewers comments (Mike Fisher (BT), Herman Postema (LOGICA))

EXECUTIVE SUMMARY

A Proof-of-Concept (PoC) is defined as a (set of) software artefact(s) used to validate some patterns of the NEXOF-RA specifications [38].

This deliverable defines the principles and the process to validate the NEXOF-RA specification via identification, set-up and execution of PoCs.

Practical experience in software architecture evaluation recommends limiting the number of qualities to be evaluated. Following this recommendation, the scope of a single PoC should be limited to the validation of few patterns and quality attributes even if, in general, there can be several patterns addressing a scenario's requirements.

Some of these patterns may be more appropriate for a specific domain, others may present a different trade-off among quality attributes, etc.

To consider this heterogeneity of cases, given the limited scope of a single PoC, the proposed process fosters the identification and definition of a set of PoCs that can allow comparison of patterns across domains or according to their similarity.

In addition, some clear principles guide the validation process.

Requirements and scenarios provide the baseline for validation of patterns via PoCs. Problem domains to be investigated and patterns to be validated are identified on the basis of scenario analysis. Backward traceability from PoCs to requirements and scenarios has to be ensured in a PoC. Scenario generation techniques are used to derive assessment criteria and metrics for PoCs.

Comparison of available alternatives in a specific problem domain is fostered in the process allowing the identification and selection of PoCs whose results can be used by the validation team to argue about similarities, differences and business cases for which a particular pattern is preferable to others.

Validation coverage allows the balancing of validation activities across different domains and different elements of the reference architecture, as well as to ensure validation from different perspectives besides pure technical ones.

Independence of the reviewers is promoted and, whenever possible, ensured.

The proposed process grounds its most critical steps in widely adopted and recognised best practices, industrial methods, and standards. Moreover, it is generic and complete enough to be applicable beyond the NEXOF-RA project. As such, it is a key element for sustainability of work engaged through committed projects of the NESSI technological platform, and it is extensible to support additional analysis on the validated patterns.

Document Information

IST Project Number	FP7 – 216446	**Acronym**	NEXOF-RA
Full title	NESSI Open Framework – Reference Architecture		
Project URL	http://www.nexof-ra.eu		
EU Project officer	Arian Zwegers		

Deliverable	**Number**	D8.0.c	**Title**	Processes, Principles and Selection Criteria behind PoC
Work package	**Number**	8	**Title**	Proof of concept

Date of delivery	**Contractual**	30/06/2010	**Actual**	30/06/2010
Status			final ☑	
Nature	Report ☑ Demonstrator ☐ Other ☐			
Abstract (for dissemination)	This deliverable reports the principles and the process to validate the NEXOF-RA specification via identification, set-up and execution of Proof-of-Concepts, defined as a (set of) software artefact(s) used to validate some patterns of the NEXOF-RA.			
Keywords	Proof of Concept, Validation, Architectural Choice, Architectural Pattern			

Authors (Partner)	Angelo Gaeta (Mo.M.A.)			
Responsible Author	Angelo Gaeta		**Email**	agaeta@momanet.it
	Partner	Mo.M.A.	**Phone**	+39 089 964364

TABLE OF CONTENTS

1 INTRODUCTION

This document presents the principles and the process behind the validation activities via Proof-of-Concept (PoC).

The document is structured as follows.

Section 2 gives an overview of the assessment and validation activities in the context of the NEXOF-RA project [1]. The purpose is to show relationships among the different project teams involved in the assessment and validation activities, to present the main purpose of the assessment and validation activities, and to motivate the common methodological ground.

Section 3 defines a PoC and its purposes.

Section 4 and related subsections present the principles behind the validation activities, and a process to identify, define, set-up and execute PoCs according to the defined principles. These sections are the core of the document and each step of the process is clearly described in terms of the rules, practices and approaches adopted.

Section 5 provides additional information and motivation for the practices and approaches adopted in the most critical steps of the process.

Section 6 presents the key features of the process and gives information on its extensibility and adaptability outside the NEXOF-RA context.

Section 7 lastly draws conclusions.

1.1 Terminology

In this document, Assessment and Validation are defined to indicate two complementary processes:

> Assessment: this refers to the process of evaluating (part of) the NEXOF-RA specifications against the requirements by means of analysing architectural documentation. It is focused on the confirmation by examination that specified requirements have been fulfilled.
>
> Validation: this refers to the process of evaluating (part of) the NEXOF-RA specifications against the requirements (non-functional and/or functional) by means of execution of Proof-of-Concept. It is focused on the confirmation by provision of objective evidence that specified requirements have been fulfilled.

Other terms adopted in this document are based on the IEEE Glossary of Software Engineering Terminology [18], IEEE Standard for a Software Quality Metrics Methodology [17], and IEEE Standard for Software Test Documentation [19].

1.2 Acronyms

ACP – Architectural Choice and Pattern

ALMA – Architecture Level Modifiability Analysis

ARID – Active Reviews for Intermediate Designs

ATAM – Architectural Trade-off Analysis Method

CA – Chief Architect

CBAM – Cost Benefit Analysis Method

E-SOA – Enterprise Service Oriented Architecture

FAAM – Family Architecture Assessment Method

IoS – Internet of Services

IT – Investigation Team

NCI – NESSI Compliant Infrastructure

NSP – NESSI Strategic Project

OCC – Open Construction Cycle

OSP – Open Specification Process

PoC – Proof of Concept

PM – Project Manager

RA – Reference Architecture

RM – Reference Model

SAAM – Software Architecture Analysis Method

SR – Scenarios and Requirements

WP – Work Package

2 AN OVERVIEW OF THE ASSESSMENT AND VALIDATION PROCESS IN THE CONTEXT OF NEXOF-RA

The following picture provides an overview of the assessment and validation process in the context of the NEXOF-RA project. The picture identifies the different teams[1] involved in the activities and how the assessment and validation process relates to the instantiation process described in [40].

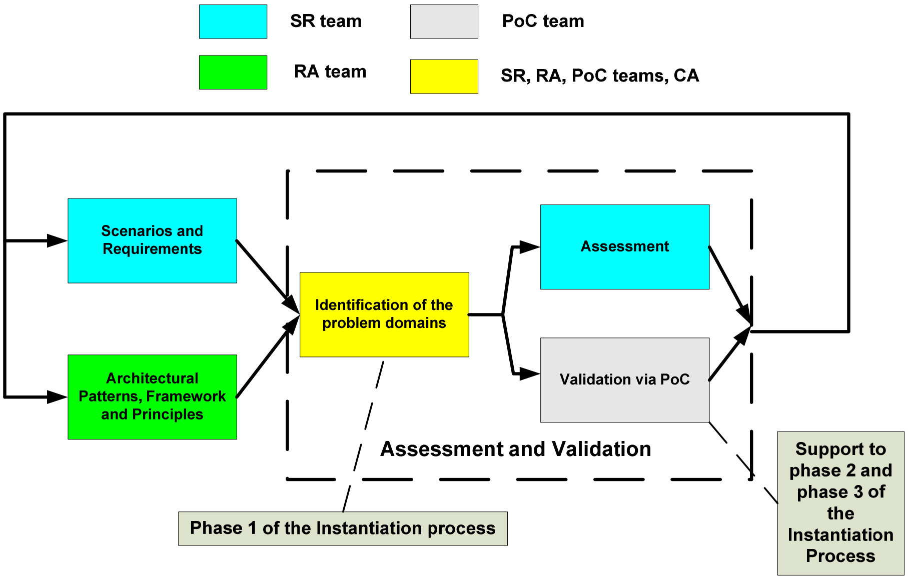

Figure 1: Overview of the Assessment and Validation process

The starting point is a set of scenarios and requirements identified by the SR team [36], and a set of patterns [38] together with the architectural principles they follow [37] defined by the RA team.

On the basis of these inputs, a cross-team activity (graphically shown with a yellow rectangle) is undertaken to identify problem domains and patterns in light of relevant scenarios' requirements and/or specific architectural problems[2].

A problem domain is a particular area of application that needs to be investigated to solve a problem. Problem domains can be, for example, "High Availability and scalability" and "Security".

Patterns provide solutions for common problems. For each problem domain, the applicable patterns, i.e. patterns that address the requirements of the scenarios linked to a problem domain, are identified.

The output of this cross-team activity is a set of tables, one for each problem domain identified. An example is shown in Table 1. Consensus among the

[1] The Reference Architecture (RA, WP7), the Proof-of-Concept (PoC, WP8) and the Scenario & Requirements (SR, WP10) teams. CA stands for Chief Architect.

[2] This activity can be executed following the principle and criteria defined in the NEXOF-RA instantiation guidelines. In particular, the identification of problem domains and sub-domains can follow phase 1 of the instantiation process described in [40]

involved teams has to be found on what are the most relevant requirements and, to this purpose, a Win-Win Negotiation process [8] could be followed.

	Pattern A	Pattern B	…
R1: Scalable performance and throughput	X	X	
R2: High Availability		X	
…			

Table 1: Patterns addressing requirements related to a problem domain

For each one of the identified problem domains, then, (some of) the patterns are:

- assessed against the requirements by the SR team, according to the process described in [39]
- validated via the PoCs[3], according to the principles, process and the criteria presented in section 4 and related subsections of this document. With full agreement of the CA and WP7 lead, the focus of this validation via PoC will be mainly, but not only, on the quality attributes and trade-off of the selected patterns.

The two processes complement each other and share the adoption of scenario generation techniques to derive assessment criteria and metrics. They differ in terms of artefacts used to evaluate the reference architecture specifications, with the first one focusing only on reference architecture documentation and the second one mainly aimed at provision of objective evidences via execution of tests on the PoCs.

It is worth mentioning that, in some cases, the outputs of the PoCs may be considered as additional architectural documentation to be used in the assessment process.

The results of the assessment and validation activities are formalised in reports and can be used by potential NEXOF-RA stakeholders (i.e. specification producers, software architects, other NESSI strategic projects) to better understand the applicability of the patterns in a specific scenario or context. Moreover, the assessment and validation helps to improve the quality of the reference architecture, not solely by identification of bottlenecks, risks or sensitivity points but also by providing additional information and considerations (e.g. suitable business cases) on the different alternatives in a specific problem domain.

[3] Validation via PoC involves quantitative evaluation of quality attributes as well as identification of trade-off, risks and sensitivity points. These are relevant steps of phase 2 and phase 3 of the Instantiation process [40]

2.1 The purpose of Assessment and Validation

The overall goal of the process is to assess and validate some key patterns, mainly the ones that solve common and challenging problems in the three project domains, i.e. Enterprise SOA, Cloud Computing and Internet of Services.

The purpose is to gather qualitative and quantitative information allowing a software architect: *i)* to understand, given a particular context (domain, scenarios, requirements), the consequences of some architectural decisions with respect to the quality attributes (non-functional requirements) of a system, and *ii)* under which circumstances (business cases, database models, infrastructure set-up) a particular architectural decision is preferred with respect to alternative ones[4].

2.2 The common ground for assessment and validation: ATAM

Assessment and validation activities are based on a common foundation that is the Architectural Trade-off Analysis Method (ATAM) [2] and, in particular, the adoption of scenario generation techniques (promoted by ATAM) to derive assessment criteria and metrics.

ATAM is a scenario-based method, representing an evolution of the Software Architecture Analysis Method (SAAM) [3], which fits well with the overall purpose of the NEXOF-RA assessment and validation process.

The ATAM method was developed to provide a principled way to evaluate the fitness of a software architecture with respect to multiple competing quality attributes: modifiability, security, performance, availability, and so forth.

ATAM is one of the most widely adopted industrial practices for evaluating software architecture and the adoption of ATAM as a common foundation is also motivated by the fact that(like other Scenario-based evaluation techniques[5]) it fits well in the architecture definition phase, i.e. the phase addressed in NEXOF-RA.

Figure 2 graphically maps different methods and techniques for architecture evaluation and validation [4] [5] in the main phases of the architecture development process. As the next sections will clarify, PoCs combine prototypes (i.e. they are software artefacts) and scenario based evaluation techniques (i.e. assessment criteria and metrics are derived using scenario based techniques).

[4] As the following sections of the document clarify, besides validating quality attributes of the patterns, the PoC process fosters pattern comparisons

[5] A review and comparison among five well adopted scenario based evaluation techniques (i.e. SAAM, ATAM, CBAM, ALMA, FAAM) is provided in [9]

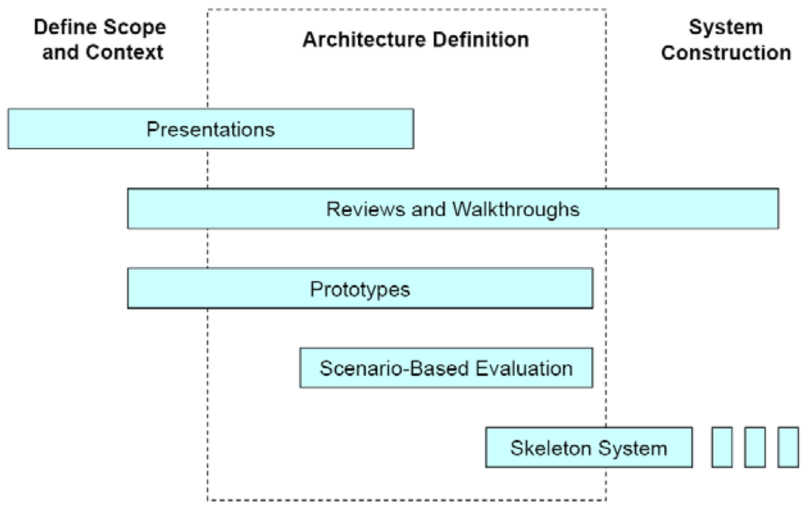

Figure 2: Architecture Validation techniques and methods (from [6])

Figure 3 provides an overview of the differences between some scenario-based evaluation techniques. The picture clearly shows that SAAM / ATAM (focusing on evaluation of quality attributes against design decisions) are more generally applicable than ALMA / FAAM (focusing mainly on product line and business information systems) or CBAM (focusing on cost-benefits analysis).

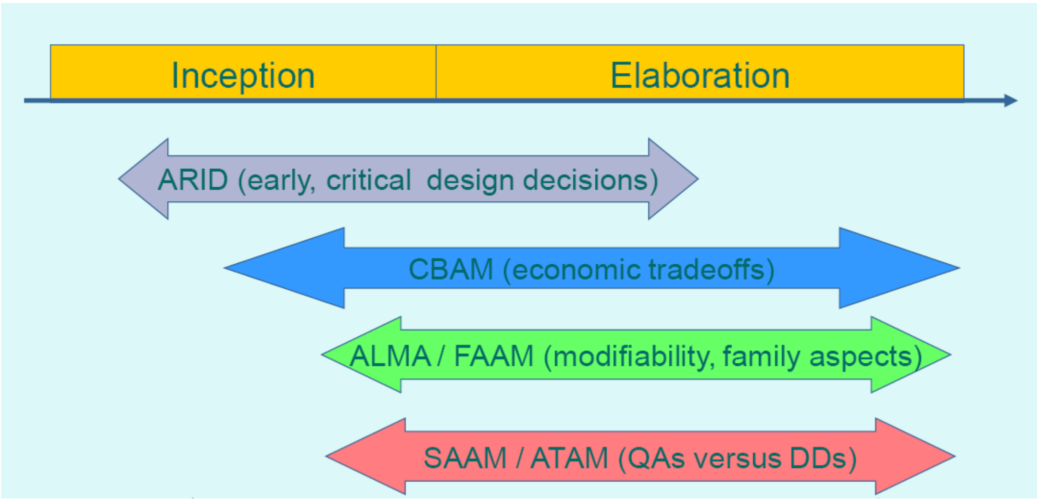

Figure 3: Main focus of Scenario-based evaluation techniques (from [7])

In NEXOF-RA the main ATAM steps adopted are the ones of the so-called *Investigation and Analysis* phase (cf. section 3 of [2]) in particular step 5 – Generate quality attribute utility tree, and step 6 – Analyze architectural approaches.

The generation of quality attribute utility trees is used to identify a common set of assessment criteria & metrics. The architectural patterns are then assessed and validated against these assessment criteria and analysed to identify architectural risks, sensitivity and trade-off points.

3 PoC: DEFINTION AND PURPOSE

A Proof-of-Concept (PoC) is defined as a (set of) software artefact(s) used to validate some key architectural patterns of the NEXOF-RA. In particular, the main focus of a PoC is on the validation of a pattern's claim about quality attributes.

Practical experience in software architecture evaluation [4] recommends limiting the number of qualities to be evaluated. In line with this, the scope of a single PoC should therefore be limited to validation of few patterns and quality attributes even if, in general, there can be several patterns that can address a scenario's requirements. Some of these patterns may be more appropriate for a specific domain, others may present a different trade-off among quality attributes, etc.

To consider this heterogeneity of cases, given the limited scope of a single PoC, it is recommended whenever possible to identify and define a set of PoCs that can allow comparison of patterns which address the same requirements or problems.

The concept of *related* PoCs is defined:

- Two or more PoCs are *related* if they validate different aspects of a pattern or a different combination of patterns addressing the same requirements / problems.

Referring to Table 1, for example, related PoCs can be defined to validate respectively Pattern A and Pattern B. Another case of related PoCs is where two PoCs validating the same pattern in different contexts[6] or domains (e.g. Cloud and IoS).

The situation is anyway more complex. Pattern A in fact may be a composition of patterns[7] and this fact can lead to the identification of additional related PoCs.

The next section presents a process that, given a particular problem domain, allows a meaningful set of related PoCs to be identified that (besides validating quality attributes of the patterns) can serve as a way to compare patterns across domains or according to their similarity. The process also gives rules to define, set-up and execute the PoCs.

An illustrative example is interpolated in the text to clarify how most critical activities of the process can be executed.

[6] In the first phase of activities, it is the case for example of the two PoCs validating the Gray-box Replication pattern in two different network contexts: LAN and WAN.

[7] This is generally indicated by the *IsPartOf* NEXOF-RA relationship [37]

4 THE POC PROCESS: PRINCIPLE AND ACTIVITIES

The PoC process supports the identification, definition, set-up and execution of software artefacts allowing validation of patterns and comparison of different alternatives.

The general principles behind the definition of this process are summarised below:

- *Requirements and scenarios.* Requirements specification provides a baseline for validation of patterns via PoC. Backward traceability from PoCs to requirements and scenarios has to be ensured in a PoC. Scenario generation techniques are used to derive assessment criteria and metrics.

  This principle guides the identification of the problem domains and patterns to be investigated (see section 2) that is the basis of our procedure to identify related PoCs (see sections 4.1 and 4.2), the definition of the objectives, assessment criteria and tests of the PoCs (see section 4.3), and the reporting of results (see section 4.5).

- *Limit the number of patterns and qualities under validation in a PoC.* As written, this principle derives from analysis of practical experiences in software architecture evaluation.

  In a PoC this helps in focusing the objectives and purposes of the PoCs.

- *Comparison of available alternatives in a specific problem domain.* The value of a PoC goes further than the validation of quality attributes of a pattern. For each of the problem domains to be investigated, the process fosters the identification of related PoCs, supporting comparison among alternatives.

  This principle guides the activities on analysis and comparison of patterns to be validated, and identification and selection of related PoCs (see sections 4.1 and 4.2).

- *Validation coverage.* In combination with the previous one, this principle aims to support a proper coverage of the validation activities (even if few patterns for each problem domain are investigated) by:
  - balancing validation activities across different domains and different elements of the reference architecture (e.g. different key concerns)
  - ensuring that the objectives of a PoC are able to validate the impact of patterns and architectural choices in multiple cases thus allowing the validation team to derive conclusions from different perspectives besides pure technical ones.

  This principle guides the activities on analysis and comparison of patterns to be validated, and identification and selection of related PoCs (see sections 4.1 and 4.2) as well as the activities on PoC definition (see section 4.3).

- *Independence of the reviewers.* This is a general principle that should guide every validation process. In our process, this is translated in the independence between PoC owners and pattern producers.[8]

The following picture shows the flow of activities of the PoC process. The picture also evidences the sections of this document explaining each activity.

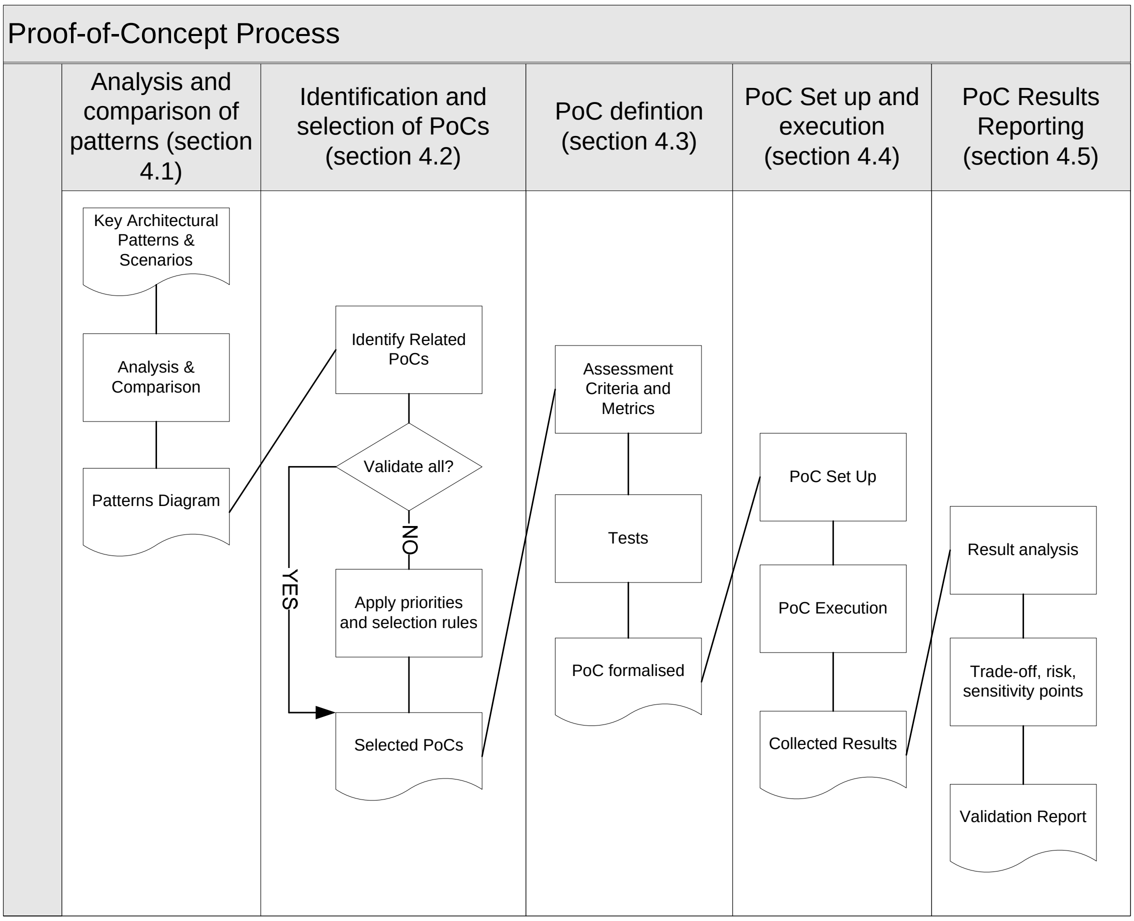

Figure 4: PoC process flow diagram

The starting point of the process is a set of patterns of a problem domain, identified on the basis of the most relevant requirements and scenarios.

The process consists of the following activities:

- **Analysis and comparison of Patterns,** whose purpose is to provide the validation team with a clear picture of the patterns of a problem domain: the (sub)problems they solve, their relationships, the consequences (i.e. system's quality attributed affected).

[8] In the context of the NEXOF-RA project (and not only), there can be difficulties in applying this principle due to the presence of project partners working in both WP1-4 and WP8. When there is no possibility to assign PoCs to organisations that are different from the pattern producers, a way to reduce the "lack of objectivity" problem may be to assign the validation activities to personnel not involved in the production of patterns but who have sufficient knowledge about the pattern.

- **Identification and selection of PoCs,** whose purpose is to identify a list of related PoCs and to select a meaningful sub-set of them to allow comparison among the available alternatives.
- **PoC definition,** whose purpose is to formalise (for each of the selected PoCs) objectives, scenario, assessment criteria, metrics and tests for the validation. A PoC is defined to validate a pattern or a combination of patterns against the defined criteria. Alternative patterns or alternative combinations of patterns may be validated through related PoCs. The main sub-activities are:
  - **Assessment Criteria & Metrics,** whose purpose is to define a set of assessment criteria and metrics for the patterns to be validated.
  - **Test planning and definition,** whose purpose is to define tests to validate the patterns against the assessment criteria.
- **PoC Set up and execution,** whose purpose is to set up and execute a PoC to accumulate evidence of the validation.
- **PoC results reporting,** whose purpose is to present the results of the validation in a way that could be useful to NESSI Stakeholders to make informed decisions.

The following subsections detail each activity and describe the approaches and techniques behind each one of them.

4.1 Analysis and Comparison of the Architectural Patterns

To identify an appropriate set of related PoCs, the validation team must have a clear picture of the patterns to validate. This activity is thus devoted to the analysis and comparison of patterns of a particular problem domain

To this purpose, the validation team can produce a pattern diagram annotating the pattern relationships map created by the reference architecture team [38] with additional information (e.g. quality attributes, comments, etc.).

An example of a pattern diagram for a problem domain is provided in Figure 5[9]. A patterns diagram may be further refined to cover only part of the problem domain. The picture presents some of the NEXOF-RA relationships defined in [37] and annotates some patterns with the quality attributes defined in the NEXOF-RA Quality Model [35].

[9] A pattern diagram may also be prepared for complex systems showing the possible combination of patterns (and their relationships) to design a complex system. Nothing change with respect to the proposed process.

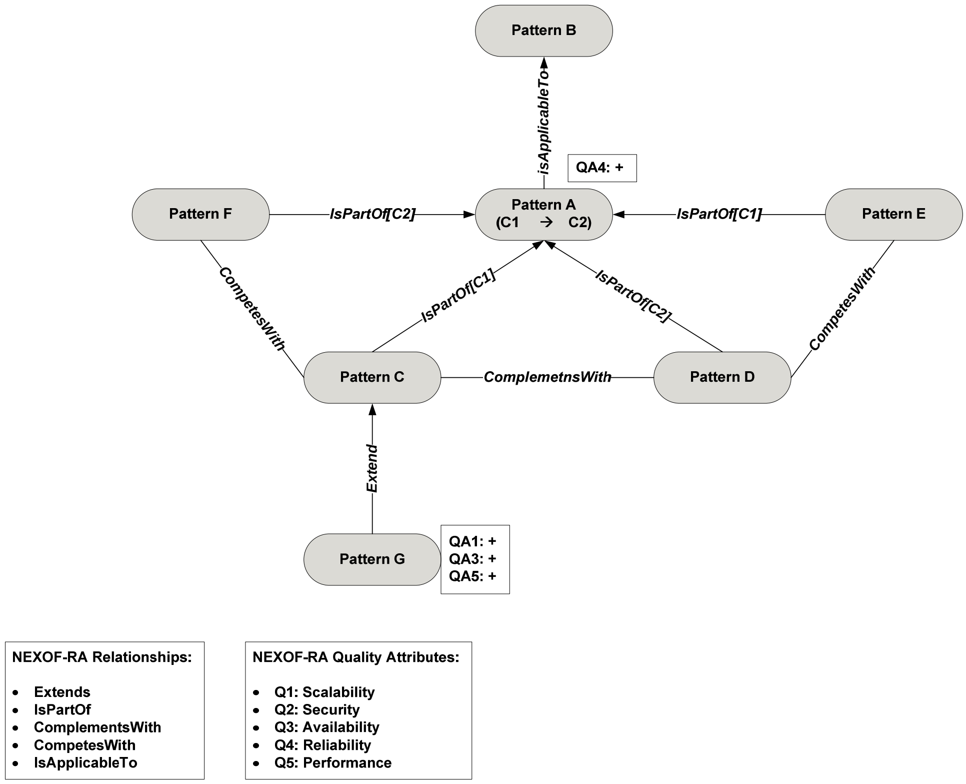

Figure 5: Patterns Diagram for a problem domain

The pattern diagram shows the relationship between Pattern A and Pattern B: Pattern A is applicable to the solution provided by Pattern B and its application positively impacts the Reliability quality attributes. In addition, the pattern diagram shows that Pattern A is composed of two parts, i.e. C1 and C2 and that:

- Pattern C is an alternative to pattern E (i.e. both can be used to design part C1 of pattern A), as well pattern D is an alternative to pattern F (i.e. both can be used to design part C2 of pattern A).
- The concurrent adoption of pattern C and pattern F to compose pattern A is discouraged since they provide two mutually exclusive parts refinement of pattern A (i.e. CompetesWith relationship). A similar argument is valid for pattern E and pattern D.
- The concurrent adoption of pattern C and pattern D is encouraged since they provide two specific and complementary parts of the pattern A (i.e. ComplementsWith relationship).

From the above, if the purpose is the validation of Pattern A the two main options in front of the validation team are: pattern C → pattern D and pattern E → pattern F with the first one encouraged by the architects that have produced the patterns[10]. In addition, the diagram shows that pattern G extends pattern C,

[10] Since there is only CompetesWith relationship between C and D (not between E and F)

i.e. can be used anywhere C can be used but makes different architectural choices.

A pattern diagram such as the one exemplified in Figure 5 shows the different alternatives in a particular problem domain that can be validated via the identification of a set of related PoCs.

4.1.1 On analysis and comparison with state of the art patterns

The proposed approach based on adoption of annotated pattern diagrams to analyse and compare patterns in a problem domain can be used in a simple and cost-effective way when the system of patterns share the same principles, criteria and language, as the NEXOF-RA patterns do.

This is not the general and most common case since patterns and pattern languages are written by different architects and authors. Besides, there are general purpose patterns [20] [21] as well as domain-specific ones [22] [23] having rich interdependencies to other general-purpose or domain-specific patterns.

To allow analysis, comparison and systematic selection among different patterns, including state of the art patterns written by different authors and coming from different source, an approach that can be followed is proposed in [10]. The approach is based on two steps.

The first step is basically similar to our approach but leverages the definition and adoption of a formal pattern language grammar to homogenise the different languages, and derive pattern sequences from the grammar.

In case of complex decision, the approach defined in [10] foresees a second step involving a further analysis with the support of instruments such as the patterns-based design space analysis (cf. section 5 of [10]).

The approach presented in [10] is suitable for comparison and selection between NEXOF-RA and state of the art patterns since it explicitly considers the quality goals of the patterns and their variants and, thus, appears to be appropriate given the project focus on quality attribute based architecture definition [26].

4.2 Identification and Selection of PoCs

This activity is devoted to the identification of related PoCs and the selection of a meaningful sub-set of them. To this purpose, the pattern diagram is an instrument that the validation team can use to identify the set of related PoCs.

In general, a number of related PoCs can be identified from analysis of the relationship among patterns. Criteria for selection can be derived from analysis of patterns assumption and consequences (e.g. trade-off claimed by the patterns).

From the example of Figure 5, the validation team can identify two related PoCs from the "*A IsApplicableTo B*" relationship, validating respectively pattern A and pattern B. Next, the validation team can identify different PoCs validating different alternatives to compose Pattern A.

The identified PoCs validate different solutions claiming different trade-offs among the quality attributes. Architectural considerations on the patterns validated via the PoC (e.g. if the pattern is cross-cutting, if the pattern is applicable across domains, etc.) have to be annotated by the validation team since this information can also support the selection process. An overview of the objectives of the PoCs has to be annotated.

The validation team can list the identified PoCs in a table such as the one proposed in Table 2, that is provided for the example of Figure 5. In addition to the information mentioned above, the table presents a column with considerations and comments of the validation team:

PoCs to validate patterns in a problem domain									
Related PoCs	**Patterns validated**	**Trade-off claimed by the patterns**					**Architect considerations**	**Objective of the PoC**	**Validation team comments**
		QA1	**QA2**	**QA3**	**QA4**	**QA5**			
PoC1	Pattern B	+		+			Applicable in cloud and E-SOA domains	Evaluate the impact of the pattern on designing available and scalable: *i)* multi-tier systems (in E-SOA), *ii)* multi-tenant systems (in Cloud) using different approaches to support multi-tenancy. Evaluate the scale-out in both the cases	It is suggested to develop two different PoCs to validate the patterns in the two domains. The two PoCs may be used to compare across-domains
PoC2	Pattern C→ Pattern D	+		+	+		Sequence for Pattern A (suggested) Pattern A is a Cross-cutting pattern applicable to pattern B to improve reliability. Pattern A is applicable to E-SOA	Evaluate the impact of the pattern sequence on designing available and scalable multi-tier systems. Evaluate the Scale out Evaluate the enhancement in terms of reliability with respect to the solution provided in pattern B. Evaluate the architect assumptions leading to the suggestion of the particular sequence	Results of the PoC1 in E-SOA can be used as baseline to evaluate the improvement in terms of reliability
PoC3	Pattern E →	+		+	+		Alternative Sequence for	Same as PoC2 with the	Alternative to

	Pattern F						pattern A	exception of the last point	PoC2.
PoC4	Pattern G → Pattern D	+		+	+	+	Refines the sequence C→D for pattern A Pattern G is a refinement of pattern C. G can be used whenever C can be used but makes different architectural choices	Same as PoC2	Validate a refinement of the sequence validated by PoC2. Positively affect performance

Table 2: List of related PoCs to validate alternatives for a problem domain

The procedure leading to the identification of the related PoCs for a particular problem domain has been guided by the first three principles presented at the beginning of section 4 and a complete validation would require the set-up and execution of all the related PoCs.

If this is not possible, due to time and resource constraints, a baseline and meaningful set of related PoCs may be identified on the basis of priorities and selection criteria.

4.2.1 Priorities and selection criteria

In NEXOF-RA, priority is given to PoCs that satisfy the following criteria:

- presenting wide objectives, e.g. PoCs aiming to validate the effect of the application of patterns to design solutions under different contexts or using different approaches. In this way, the results of a PoC can be used to compare among different situations and provide additional information either to the end-user (e.g. software architect willing to use the pattern) or to the pattern producer (e.g. to improve the pattern description).
- validating patterns applicable across more domains or patterns applicable to emerging domains such as Cloud and IoS. This criterion is aimed at supporting the possibility of comparing results across domains and a well balanced coverage of the three domains investigated in the project (given the availability in the project of results on patterns applicable to E-SOA).
- leveraging existing results. In this case, for example, the validation team can select PoCs whose results can be discussed and compared with existing ones to inform debate on alternatives and/or cross-domain comparisons. Another case is to leverage an existing PoC and define additional tests. This criterion is mainly motivated by the fact that software architecting is an iterative process and, thus, previous validation results / artefacts may exist in the organisation[11] (or in literature).

[11] This is the case of the NEXOF-RA project where a set of results on patterns applicable to E-SOA are already available for some problem domains such as High Availability and Security.

The criteria presented above respect the general principles identified for the PoC process and can help the validation team in reducing the number of PoCs to be set-up and executed without losing the possibility of providing some relevant and useful considerations from several perspectives on the results of validation, leveraging already available results.

4.3 PoC definition

This activity is devoted to formalising each of the selected PoCs. Each PoC must be clearly defined in terms of:

- its objective
- its architectural diagram showing the pattern or the patterns sequence validated by the PoC, and presenting the alternatives
- its scenario for validation
- the assessment criteria and metrics
- a description of the tests to be executed
- its related PoCs (e.g. PoCs validating alternatives in the same problem domain, PoCs validating the same patterns in different contexts or domains).

A template to define a PoC is presented in Annex A of this document.

The most critical activities during the definition of a PoC are *i)* the identification of assessment criteria for validation and metrics, and *ii)* the planning and documentation of the tests to be executed during the PoC execution. These activities are described in the following subsections.

4.3.1 Assessment Criteria & metrics

Assessment criteria and metrics are identified for each PoC.

The assessment criteria for the patterns are based on the quality attributes of the NEXOF-RA Quality Model [35], that is solidly grounded in widely adopted software engineering standards [17] [18] [24] [25].

The emergence of new trends and domains (such as Cloud computing, Internet of the Services, Web 2.0) in the software and services area has led to the identification of non-functional qualities not explicitly included in the current standards but related to them. It is the case, for instance, that the elasticity quality in Cloud Computing, (according to [27]), relates to (horizontal and vertical) scalability and reliability.

To identify assessment criteria, the validation team will focus on cross-domain qualities, such as security, scalability, availability, reliability, modifiability and performance taking into consideration, if necessary, current results and initiatives on the Future Internet [28] to understand how these quality attributes relate to non-functional aspects such as elasticity and adaptability.

The identification of assessment criteria is supported by scenario generation techniques as proposed in ATAM's step 5 [2]: for each scenario a quality attribute utility tree is generated and annotated with stimulus-response. The utility tree is used to prioritize and refine the quality goals and will serve as a basis for the validation of the patterns and to reason about identification of sensitivity and/or trade-off points.

To each quality attribute a metric is associated. The NEXOF-RA Quality Model [35] already provides a set of metrics associated to the quality attributes (i.e. Mean Time To Failure[12] is a metric for the Reliability attribute) and a PoC leverages these metrics.

If additional metrics have to be defined, they have to be validated following the principle and the framework defined by the IEEE in [17] and then added to the NEXOF-RA Quality Model.

To allow meaningful comparisons, all related PoCs have to share a common set or sub-set of assessment criteria and metrics.

4.3.2 Test planning and definition

Appropriate measurement of non functional requirements is a very critical step to evaluate risks, non-risks, tradeoffs, and metrics and how each business requirement could be affected.

This step is devoted to the definition of tests to be executed via a PoC.

According to [19], a test refers to: *(A) A set of one or more test cases, or (B) A set of one or more test procedures, or (C) A set of one or more test cases and procedures.* In the context of a PoC, "test item" and "testing" [19] refers respectively to the pattern (or the pattern sequence) implementation and to the process of analysing the pattern (or pattern sequence) to evaluate and measure each quality attribute, according to the utility tree defined in the previous steps.

Tests should be planned and defined according to the set of basic test documents defined in [19].

Regarding the specific techniques to be used in the test, in a PoC non functional software testing techniques (such as performance, reliability, scalability, security) should be properly selected and described depending on the specific quality attributes under validation.

4.4 PoC set up and execution

This step foresees the design and implementation of all the assets of a PoC (i.e. patterns, scenario, test cases or benchmark applications) and its execution in order to proof the architectural trade-off of the patterns to be validated. It is suggested to leverage concrete components listed in the RA Specification [38] for the pattern implementation, and / or to adopt specific techniques (e.g. Mock Objects[13]) used in Test Driven Development in order to implement the PoCs without having to implement all the parts of the system required by a pattern.

---

[12] Defined as *the time value indicating, on average, how much time it takes from one service failure (unavailable) to the next*

[13] http://www.mockobjects.com/

Test cases defined or benchmark applications are executed to accumulate evidence for the validation.

Thus PoC execution is useful to explore and collect evidences of several aspects such as failure situation, bottlenecks, resource starvation, database performance, etc.

This information can support the validation team in identification of risks, sensitivity and trade-off points.

4.5 PoC results reporting

A validation report is produced for each PoC. A report contains:

- architectural patterns validated
- scenarios and utility tree
- assessment criteria and metrics
- test cases
- risks, non-risks, sensitivity and trade-off points found
- comparison with the results of related PoC and references to the result reports of the related PoCs
- concrete components that can be used to implement the patterns
- graphs and tables
- lessons learnt and conclusions of the validation team on specific circumstances in which the validated pattern is preferable to the alternatives available. This may also include recommendations for future work to achieve further validation results (e.g. suggested related PoCs to set-up and execute and / or additional tests to define to validate further aspects) as well as recommendation on how to improve the patterns.

5 MOTIVATION OF THE SELECTED APPROACHES AND PRACTICES BEHIND PoC PROCESS

Figure 6 recaps the PoC process showing also the methods, approaches and practices on which the activities are based.

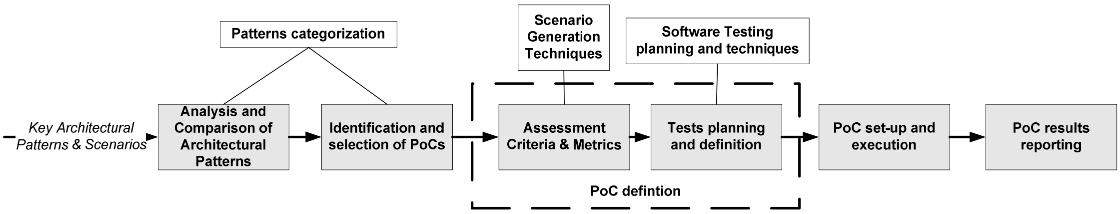

Figure 6: PoC process activities

As the picture shows, the most critical steps of our process leverages on *Patterns categorization and analysis of relationships*, *Scenario Generation techniques*, *Software Testing planning and techniques.* In general, as the next sub-sections will detail, this is motivated by the fact that they are widely adopted practices and approaches suitable for the specific steps in which they are exploited.

5.1 Motivation of the approach behind the pattern analysis & comparison, and identification and selection of PoCs

The first activity of the PoC team is the analysis and comparison of architectural patterns (coming from the Architecture team) in order to identify related PoCs. For each set of patterns belonging to a particular problem domain, the PoC team uses a pattern diagram to support the identification and selection of PoCs.

The patterns diagram can be created thanks to the constructional-pattern-based approach promoted in the NEXOF-RA framework and principles [37]. This approach allows categorising and relating patterns with respect to a problem domain.

In section 4.1.1 we already argued that the proposed approach can be used when the patterns to be analysed and compared share a common ground, such as the NEXOF-RA ones. In case of analysis and comparison also with other patterns, a practical approach that can be followed is provided in [10].

In the literature, there are approaches similar to the ones adopted for identification of related PoCs, that are based on Pattern Language [11] [12].

A similar process for instance is presented in [14] to identify hot-spot (i.e. framework parts that must be kept flexible, as they are specific to individual systems) in a pattern language. Different types of hot-spot are defined in [14] such as “optional pattern”, “optional participant”, “change relationship”, etc. and the guidelines proposed to identify hot-spots can be summarised in the following picture:

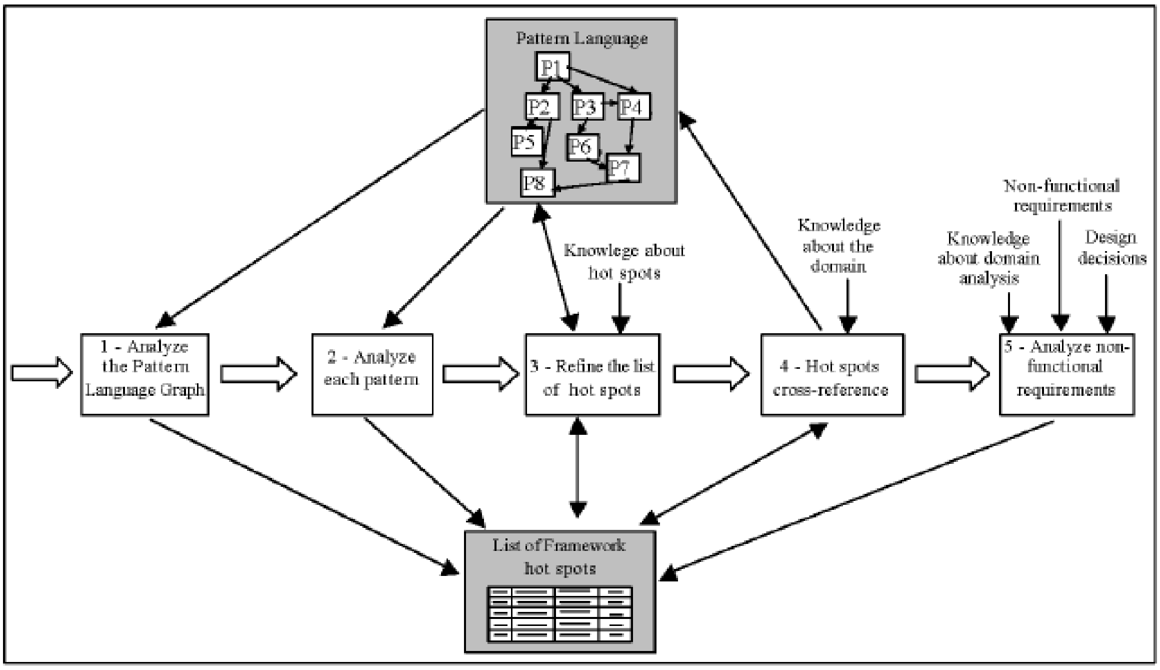

Figure 7: Process to identify hot-spot from pattern language (from [14])

The process to identity hot-spot presents similar steps to the one proposed in this document to identify and select PoCs: both start from the analysis of a pattern graph, analyse the pattern to identify a set of items and their relationships (cross-referenced hot-spots or related PoCs), consider the non-functional qualities (respectively to add new hot-spots or to support the selection of PoCs). Both the processes, in addition, aims to identify relevant elements from the pattern texts[14] and map in the definition of list of respectively hot-spot types or related PoCs.

However, within NEXOF-RA the goal is different. The process described in [14] is aimed at reducing the complexity of object-oriented framework development, and thus introduces hot spot types that can be found in object-oriented frameworks. Our approach is, instead, aimed at supporting identification and selection of a meaningful set of alternatives to be evaluated and validated and is based on general pattern relationships (rather than focusing on object-oriented frameworks).

Pattern Language is the most widely adopted approach to support pattern analysis and selection. Other approaches exist for the formal specification of software patterns, such as for example LePUS (LanguagE for Pattern Uniform Specification) [15], and also for formal specification and verification of patterns [16]. As evidenced also in [10], these approaches, however, have not gained much momentum mainly because of their complexity and the resulting limitations regarding their practical use.

[14] In the case of our approach, these elements are the ones annotated in the Architects Considerations column of Table 2

5.2 Motivation of the technique behind the identification of assessment criteria and metrics

The identification of assessment criteria and metrics is one of the two most critical steps in the PoC definition activity.

Scenario generation techniques are adopted in the PoC process to identify assessment criteria and metrics. NEXOF-RA assessment criteria and metrics are based on the NEXOF-RA Quality Model [35]. If additional metrics need to be defined, they should have a proven degree of association with the quality attributes and, thus, should be validated following the principle defined by the IEEE Standard for a Software Quality Metric Methodology (e.g. the validation criteria provided in [17]).

Scenario generation is a technique for capturing quality attributes and refining them into quality attribute scenarios. This technique is adopted in several industrial methods such as ATAM, the Quality Attribute Workshop (QAW) [31], the Active Review for Intermediate Design (ARID) [32] and the Software Architecture Comparison and Analysis Method (SACAM) [33].

Besides being widely supported by industrial methods and best practice approaches, this technique fits well with the overall project focus on quality attribute validation.

5.3 Motivation of the Software Testing planning and techniques

The definition of tests is the other most critical step in the PoC definition activity.

It is mandatory to define the tests in a proven way and this motivates the adoption of a widely adopted standard for test planning and identification of test artefacts.

In the PoC process the IEEE Standard for Software Test Documentation [19] is promoted to plan and define tests artefacts.

Depending on the specific quality attributes to be evaluated, appropriate non-functional software testing techniques (e.g. Software performance testing, Stress testing, Load testing, Volume testing, Scalability testing) have to be used for the tests.

6 KEY FEATURES AND ADDITIONAL CONSIDERATION ON THE POC PROCESS

The key features of the PoC process are the following:

- it grounds its most critical steps in widely adopted and recognised best practices, industrial methods, and standards
- it supports pattern comparison across domains or according to their similarity, via the validation of related PoCs
- it is generic and complete enough to be applicable beyond the NEXOF-RA project and as such is a key element for sustainability of work, particularly in committed projects (NSPs & NCPs). Key pillars of the NEXOF-RA project are the priorities and the selection rules presented in section 4.2.1, and the adoption of the NEXOF-RA Quality Model to derive assessment criteria and metrics. Adopting different priorities, rules and quality model does not affect the validity of the process
- it is extensible to consider additional analysis of the validated patterns (see section 6.1)
- it supports external contributions (see section 6.2).

6.1 Possible extension of the PoC process

The output of the PoC process is a report providing qualitative and quantitative information about the validated patterns. The main focus of the PoCs is to validate quality attributes of the patterns and provide the architect with information on trade-offs, risks and sensitivity points of systems that can be built on the basis of the validated patterns.

There are at least two possibilities to extend the PoC process.

The first one relates to extension of the PoC process to acquire important information such as cost-benefit analysis of implementing architectural patterns or decision. This can be done exploiting the Cost Benefits Analysis Method (CBAM) [29] [30] at the end of the PoC process. It is worth mentioning that the set-up of the PoC can provide information about the estimated cost of implementing a particular pattern more precisely than the rough estimation foreseen in the so-called CBAM *Triage Phase*.

The second possibility relates to extension of the PoC process to allow architecture validation via the systems skeleton techniques. This kind of validation can only be done by developing a part of the solution to demonstrate that the architecture is viable. The architectural skeleton is then incrementally fleshed out with details. The PoCs may be used thus as building blocks for a NEXOF-RA architecture skeleton to be further validated.

If a skeleton is proven, it can be considered as a building block of a NESSI Compliant Infrastructure.

6.2 External contributions

External contributions refer to:

- **Proof of Concept ideas,** i.e. scenarios, metrics and measurements for evaluating whether all the requirements and statements made in the patterns are correct.
- **Software Artefacts**, to extend the list of concrete components which can be used to set-up a PoC.

The purpose is to increase the knowledge base of NEXOF-RA by increasing the number of available alternatives for pattern implementation, and to demonstrate that the results provided are good by gathering ideas about scenarios which would support the validation of patterns (individually and in conjunction with other patterns).

Moreover, it is a way to increase the sustainability of NEXOF-RA results by supporting other projects and showing that the proposed patterns and their evaluation are useful for other projects

Reservoir and SLA@SOI strategic projects have already agreed to contribute as part of a joint initiative by providing PoC ideas on cloud computing in the PoC process.

7 CONCLUSION

Validation of NEXOF-RA specifications is a critical activity whose results are of clear value for several NESSI stakeholders: specification producers, software architects, other NESSI strategic projects.

This document has presented a process to validate patterns via identification, set-up and execution of PoCs, defining principles, rules and criteria to follow during the activities. Best practices, techniques and widely adopted approaches have been proposed for each step of the process.

The NEXOF-RA validation team will follow as much as possible this process for second phase activities but, given time and resources constrains, some limitations may be applied. These limitations, will not contradict the guiding principles and criteria and will not affect the quality of validation results.

The validation team strongly encourages others to follow the defined process for future activities devoted to the analysis of the NEXOF-RA architectural patterns and to the validation of some of them so that there is a common basis for comparison with alternatives.

The process is in fact generic and complete enough to be applicable besides the NEXOF-RA project and as such is key element for sustainability of work engaged through committed projects (NSPs & NCPs). Moreover, it is extensible to support additional considerations / validation activities.

In conclusion, it is worth mentioning that the principles and the process described in this document have to be considered as a part of the whole set of principles and processes of the NEXOF-RA project (such as the ones to define the architectural specification, the instantiation guidelines, etc.) that will survive after the end of the project, and will provide the common theoretical and methodological ground for external stakeholders interested in NEXOF-RA project results.

8 REFERENCES

[1] NEXOF-RA Project, http://www.nexof-ra.eu/

[2] Clements, Kazman, Klein. ATAM: Method for Architecture Evaluation. August 2000, available at http://www.sei.cmu.edu/reports/00tr004.pdf

[3] R. Kazman, G. Abowd, L. Bass, P. Clements, "Scenario-Based Analysis of Software Architecture", IEEE Software, Nov. 1996, 47-55

[4] R. Kazman, P. Clements, et. al, Recommended Best Industrial Practice for Software Architecture Evaluation, Technical Report CMU/SEI-96-TR025 ESC-TR-96-025, January 13, 1997

[5] Nick Rozanski and Eoin Woods, Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives. Addison-Wesley Professional (April 30, 2005), ISBN 978-0321112293

[6] Tutorial on architecture validation, University of Rome 3, Available at http://www.dia.uniroma3.it/~cabibbo/asw/pdf/asw950-validazione-architettura.pdf (in Italian)

[7] Herman Postema, Evaluation of (Software) Architectures

[8] Boehm B. et. al, Software Requirements as Negotiated Win Conditions", 1st Int. Conf. on Req. Engineering (ICRE94). Colorado Springs: IEEE Computer Society Press, 1994

[9] M. Ionita, D. Hammer, H. Obbink: Scenario Based Software Architecture Evaluation Methods: An Overview. Workshop on Methods and Techniques for Software Architecture Review and Assessment at the International Conference on software Engineerign, Orlando, Florida, USA 2002

[10] Zdun U., Systematic pattern selection using pattern language grammars and design space analysis. Soft. Pract. Exper. 2007; 37:983-1016, Wiley InterScience, 19 Dec. 2006

[11] C. Alexander, S. Ishikawa, M. Silverstein, M. Jakobson, I. Fiksdahl-King, and S. Angel. A Pattern Language – Towns, Buildings, Construction. Oxford Univ. Press, 1977

[12] Avgeriou, Paris; Uwe Zdun (2005). "Architectural patterns revisited: a pattern language". 10th European Conference on Pattern Languages of Programs (EuroPlop 2005), Irsee, Germany, July

[13] Zdun U. Pattern language for the design of aspect languages and aspect composition frameworks. IEEE Proceedings Software 2004; 151(2):67–83

[14] Rosana T. Vaccare Braga, Paulo C. Masiero: "Finding Frameworks Hot Spots in Pattern Languages", in Journal of Object Technology, vol. 3, no. 1, January-February 2004, pp. 123-142. http://www.jot.fm/issues/issue_2004_01/article2

[15] Eden AH, Hirshfeld Y. LePUS—symbolic logic modeling of object oriented architectures: A case study. Proceedings of the 2nd Nordic Workshop on Software Architecture (NOSA'99), Ronneby, Sweden, April 1999

[16] J. Dong, P. Alencar, and D. Cowan, "Formal specification and verification of design patterns," Design Pattern Formalization Techniques, pp. 94–108, 2007

[17] IEEE Standard for a Software Quality Metrics Methodology, IEEE Std. 1061-1998

[18] IEEE Standard Glossary of Software Engineering Terminology, IEEE Std. 610.12-1990

[19] IEEE Standard for Software Test Documentation, IEEE Std. 829-1998

[20] Gamma E, Helm R, Johnson R, Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley: Reading, MA, 1994.

[21] Buschmann F, Meunier R, Rohnert H, Sommerlad P, Stal M. A System of Patterns (Pattern-Oriented Software Architecture, vol. 1). Wiley: New York, 1996

[22] Kircher M, Jain P. Patterns for Resource Management (Pattern-Oriented Software Architecture, vol. 3). Wiley: New York, 2004.

[23] Volter M, Schmid A, Wolff E. Server Component Patterns—Component Infrastructures Illustrated with EJB. Wiley: New York, 2002.

[24] ISO/IEC 9126 International Standard. Information technology - Software product evaluation - Quality characteristics and guidelines for their use, 1991

[25] ISO/IEC 25000:2005 Software Engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE

[26] Toward Deriving Software Architectures From Quality Attributes, Rick Kazman, Len Bass, Rick Kazman, Thomas R. Miller, Lt Col, 1994

[27] The Future of Cloud Computing, opportunities for European Cloud Computing beyond 2010. Expert Group Report, Public Version 1.0, available at http://cordis.europa.eu/fp7/ict/ssai/docs/cloud-report-final.pdf

[28] Future Internet initiative, http://www.future-internet.eu/

[29] Cost Benefit Analysis Method, http://www.sei.cmu.edu/architecture/cbam.html

[30] Robert L. Nord, Mario R. Barbacci, Paul Clements, Rick Kazman, Mark Klein, Liam O'Brien, James E. Tomayko, Integrating the Architecture Tradeoff Analysis Method (ATAM) with the Cost Benefit Analysis Method (CBAM) (CMU/SEI-2003-TN-038).

[31] Barbacci, M.; Ellison, R.; Lattanze, A.; Stafford, J.; Weinstock, C.; & Wood, W. Quality Attribute Workshops (QAWs), Third Edition (CMU/SEI-2003-TR-016). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2003

[32] Clements, P. Active Review for Intermediate Design. Technical Note CMU/SEI-2000-TN-009

[33] Stoermer, C. et. al. SACAM: The Software Architecture Comparison Analysis Method. Technical Report CMU/SEI-2003-TR-006 ESC-TR-2003-006. December 2003

[34] Proof of Concept Report V1.1, NEXOF-RA deliverable D8.1.b, available at http://www.nexof-ra.eu/sites/default/files/D8.1.b%20Proof-of-concept_V1.1.pdf

[35] Quality Model for NEXOF-RA Pattern Designing, NEXOF-RA deliverable

[36] Requirements Report, NEXOF-RA deliverable D10.1.b

[37] Definition of an architectural framework and principles, NEXOF-RA deliverable D7.2.c

[38] RA Specifications, NEXOF-RA deliverable D7.5.b

[39] Assessment Criteria, NEXOF-RA deliverable D10.2.c

[40] Instantiation Guidelines, NEXOF-RA deliverable D14.1

[41] NEXOF-RA Model, NEXOF-RA deliverable D6.3

9 ANNEX A – POC TEMPLATE

The following template is used to describe a PoC. The template requires the following details: *i)* What is the objective of the PoC, *ii)* Why the objective of the PoC is relevant, especially for the project, and *iii)* How the validation should be proven (methods, criteria, metric, way to document).

POC Name	
What	
Owners	Owners Short Name
Description of the PoC	A clear description of the PoC
Architectural Choices and Patterns (ACP) involved	Description of the ACPs involved.
Objective of the PoC	What is the objective of the validation?
Functionalities	Description of the functionalities of the PoC
Dependencies	Describe the relationships with other ACPs not included in this PoC.
Why	
Rationale	The motivation behind the identification of the PoC in NEXOF-RA.
Architecture Component(s) affected	Specify the architecture components (according the model provided by WP7) affected by this PoC.
Alternatives	Are there alternatives to the proposed PoC?
Relationship with the NEXOF-RA	Clarify the relationship with the NEXOF-RA
How	
Scenarios for validation	Specify scenarios. Scenarios can be linked to WP10.
Suggested Architecture	Provide a detailed design of the PoC
Environment	Specify the environment in which the validation will be performed. This section will report tools, frameworks, standards, etc. and requested integrations among them
Estimated Effort	Describe and detail the required effort for the proposed PoC.
Methods	Describe the methods adopted for validation of the ACPs. Presents the tests to be executed.
Assessment criteria & metrics & way to document	Give clear and measurable assessment criteria to validate the ACPs and metrics to measure the assessment. Presents also the foresaw modality to document the result.
Further Information	Further information considered useful for describing the specific aspects that should be implemented or stressed, any additional feedback required (i.e. for improving other aspects), significant warning related to specific topics, generic comments, etc.