Process Patterns for Requirement Consistency Analysis

PADMALATA V NISTALA, TCS Research, Tata Consultancy Services
KESAV V NORI, International Institute of Information Technology- Hyderabad
SWAMINATHAN NATARAJAN, TCS Research, Tata Consultancy Services

In the requirement space, patterns are gaining prominence to capture the requirement knowledge for reuse and help identify requirements. The quality of requirement specification is critical for effective understanding and implementation of requirements. This paper presents a set of process patterns that use the concept of compositional traceability to analyze how well the requirement specifications or user stories have been formulated and to identify inconsistencies among its encompassing elements. We present two patterns that have been successfully applied and found useful to carry out consistency analysis on requirements and detect inconsistencies in requirements: Requirements coverage analysis and Requirements traceability analysis. These patterns can be applied to projects during requirements phase for review and analysis of stated requirements. The paper describes the patterns, discusses its implementation and results from a project case study.

CCS Concepts: • **Software and its engineering → Software creation and management → Designing software → Requirements analysis**

Additional Key Words and Phrases: Process pattern, Requirements pattern, Requirements analysis, Requirements coverage, Requirements traceability, Requirements quality, Requirements consistency

ACM Reference Format:

Padmalata V Nistala, Kesav V Nori and Swaminathan Natarajan. 2016. Process Patterns for Requirements Consistency Analysis. *EuroPLoP'16*, Article (July 2016), 11 pages.
DOI:http://dx.doi.org/10.1145/3011784.3011809

1. INTRODUCTION

Patterns are popular in software engineering community especially for object oriented design patterns and architectural patterns [Gamma] [POSA]. The potential of software patterns as a unified body of knowledge for wide spread usage has been recognized and requirement patterns have been identified as a hotspot research area in Requirements Engineering to improve productivity of requirements analyst and quality of requirement artefacts, and creating repositories to store requirement patterns for reuse is recommended [Henninger] [Cheng].

A requirement pattern is a reusable, experience based framework that aids a requirement engineer to write or model better quality requirements in as less time as possible [Mahindra]. Recent approaches using patterns for writing software requirement specifications can be found in the work of Withall and in Pattern-based Requirements Elicitation (PABRE). Withall pronounces a requirement pattern as a guide to write a particular type of requirement and enabler for writing high quality requirements quickly and with less effort. His catalog comprises 37 patterns organized into eight domains: fundamental, information, data entity, user function, access control, performance, flexibility and commercial [Withall]. PABRE pattern based requirements elicitation framework uses patterns as a means to capture and reuse requirements, specifically for non-functional requirement schemas [Pabre].

Author's address: Padmalata V Nistala; email: nistala.padma@tcs.com; Kesav V Nori; email: kesav.nori@gmail.com; Swaminathan Natarajan; email: swami.n@tcs.com;

EuroPLoP '16, July 06-10, 2016, Kaufbeuren, Germany

DOI:http://dx.doi.org/10.1145/3011784.3011809

Requirements engineering process comprises elicitation, analysis, management and documentation [Sommervile]. Patterns are applicable to each of these stages. Amber introduced the concept of process patterns stating 'Not only do the same type of problems occur across domains, problems whose solutions are addressed by design patterns and analysis patterns, but the strategies that software professionals employ to solve problems recur across organizations, strategies that can be described by *process patterns'* [Amber]. Amber introduces three types of process patterns: phase, stage and task patterns. Hagge proposed few patterns for requirements process, formulating tactics that have been successfully applied in projects [Hagge]. These patterns offer guidance for organizing the specification procedure and for eliciting, specifying, and verifying requirements.

By and large it can be seen that the focus of requirement patterns has predominantly been on 'requirements elicitation' and not many patterns can be found for other stages: requirements analysis, management and documentation. Requirements analysis focuses on strategies or techniques for evaluating quality of the recorded requirements and identifying errors in requirements. Requirement specifications form the basis for discussion among key stakeholders, for project estimations, development planning and execution. The quality of requirement specification is critical for effective understanding and implementation of requirements. Given limitations of time and domain knowledge, during requirement workshops and review discussions with subject matter experts, only a few of the errors in requirements are detected and rectified. Statistics on industrial systems [Jones] show a high percentage of defects due to requirements in range of 30-40% resulting in high rework and cost. Also requirements related to quality aspects such as security, performance targets and so on are often missed. Subsequently, many clarifications on requirements are sought during design and development phases, leading to loss of productive development time in obtaining clarifications, rework effort and slippage of milestones. In this paper, we propose two patterns that can help to effectively carry out requirements analysis and detect errors or inconsistencies in requirements.

The conceptual foundation for these patterns is based on principle of 'Compositional traceability' [SQA]. A software requirements specification is traceable [IEEE830] if (i) the origin of each of its requirements is clear and if (ii) it facilitates the referencing of each requirement in future development or enhancement documentation. Software development can be viewed as involving decisions relating to various system layers such as business, requirements, specifications, design etc. Traceability involves tracking the relationships between decisions across these layers. The aim is to be able to assert that the decisions in lower layers are necessary and sufficient to implement the higher layer decisions, however this rationale generally remains implicit in the minds of engineers, typical traceability practices only capture that a relationship exists. Compositional traceability includes the aspect of compositional equivalence between specification and collection of elements that address the specification. It necessitates verifying the compositional logic involved. A recent ISO technical specification on product quality, ISO TS 30103 articulates the need for managing content consistency relationships among the information items as they are concurrently elaborated during project enactment [ISO 30103]. The patterns have been developed based on industry experience of applying these concepts and techniques in multiple projects and through abstracting the methodology and approach adopted in those case studies as generic patterns [Nistala1] [Nistala2] [Nistala3].

Two patterns are proposed for requirements analysis: **Requirements coverage analysis and Requirements traceability analysis**. Requirement coverage analysis pattern addresses the problem of missing requirements related to implicit Non Functional Requirements (NFRs) or generic problem analysis dimensions and helps to strengthen requirements by identifying additional requirements to ensure coverage with respect to those dimensions. Requirements traceability pattern addresses the problem of inadequate implementation of traceability in projects and helps to create a well formulated requirement traceability matrix with backward and forward traceability.

The paper is structured as per following sections. Section 2 describes the patterns - 2.1 and 2.2 detail the two patterns, Section 3 presents case implementation of patterns and cost benefit analysis, and section 4 concludes the paper with summary and future work.

2. PATTERNS FOR REQUIREMENTS ANALYSIS

2.1 **Requirements Coverage Analysis**

Pattern Name: Requirements Coverage Analysis

Context: This pattern is applicable during requirements phase of projects, for review activity of stated requirements provided by customer. It is particularly useful when teams have to review requirements, ask clarifications and work out project estimates based on requirements understanding.

Problem: Implicit or Non-Functional Requirements (NFRs) such as security, performance targets, compatibility and so on are often missed or inadequately covered in elicited requirements. Many NFR issues come up quite late in lifecycle during acceptance testing or even in production, causing schedule delays and outages to production systems. Also, some important logic and rules are not covered in requirement specifications and clarification are often sought during design and development phases, at which point customer subject matter experts may not readily be available for clarifications, resulting in loss of productive time and delays. Project teams underutilize requirements sessions with customers as they cannot identify what all need to be asked as clarifications to requirements.

Forces: Project factors that favor usage of this pattern are: need to review requirements and provide estimations in a short span of time, project team's lack of awareness on NFR dimensions resulting in missing requirements, lack of experience to analyze coverage of requirements resulting in late identification of requirement gaps, lack of any structured method adoption for analyzing requirements and limited availability of project subject matter experts,

Solution: Multi-dimensional analysis is the principle applied for requirements coverage analysis. The pattern ensures that relevant requirement dimensions are explicitly considered and requirement gaps are detected. The key dimensions proposed are NFR, reasoning and domain dimensions. Usage of industry standard product quality dimensions and standard reasoning dimensions provide comprehensive dimensional coverage. Coverage of requirements is analyzed from multiple dimensions such as NFRs, reasoning and domain.

Requirement coverage analysis in this pattern is carried out in following steps:

a) *Coverage for NFR dimensions:* In this step, requirements are analyzed for coverage of NFRs. ISO/IEC 25010 software product quality model has a comprehensive set of NFR definitions and is taken as a reference model for NFR coverage analysis in this pattern [ISO 25010]. Requirements are analyzed for coverage with respect to quality characteristics (QCs) and sub characteristics as defined in ISO standard. NFR dimension classification as per ISO model is depicted in Figure 1.

Stated requirements are checked for relevance with respect to each NFR classification of QC/ Sub QC and gaps found in requirements with respect to these dimensions are logged into Table 1 Requirement Coverage Tracker (RCT) as inconsistencies. If requirements for a QC are missing, it is marked as a coverage gap. For example security-confidentiality requirements are not stated, then 'NFR security dimension' is marked as a gap and entered into Table 1 under head 'NFR dimension' as 'security confidentiality requirements not stated for the system'.

b) Coverage for reasoning dimensions: In this step, requirements are analyzed for coverage with respect to reasoning dimensions. 5W1H (who, when, where, what, why and how) is a method of reasoning that is used in multiple contexts to analyze a problem and classify information [Kim][Ikeda]. The mapping of each reasoning dimension to requirement perspectives is depicted in Figure 2. If the requirement perspectives for 5W1H dimensions can be traced for a requirement, its coverage is considered complete from reasoning dimensions.

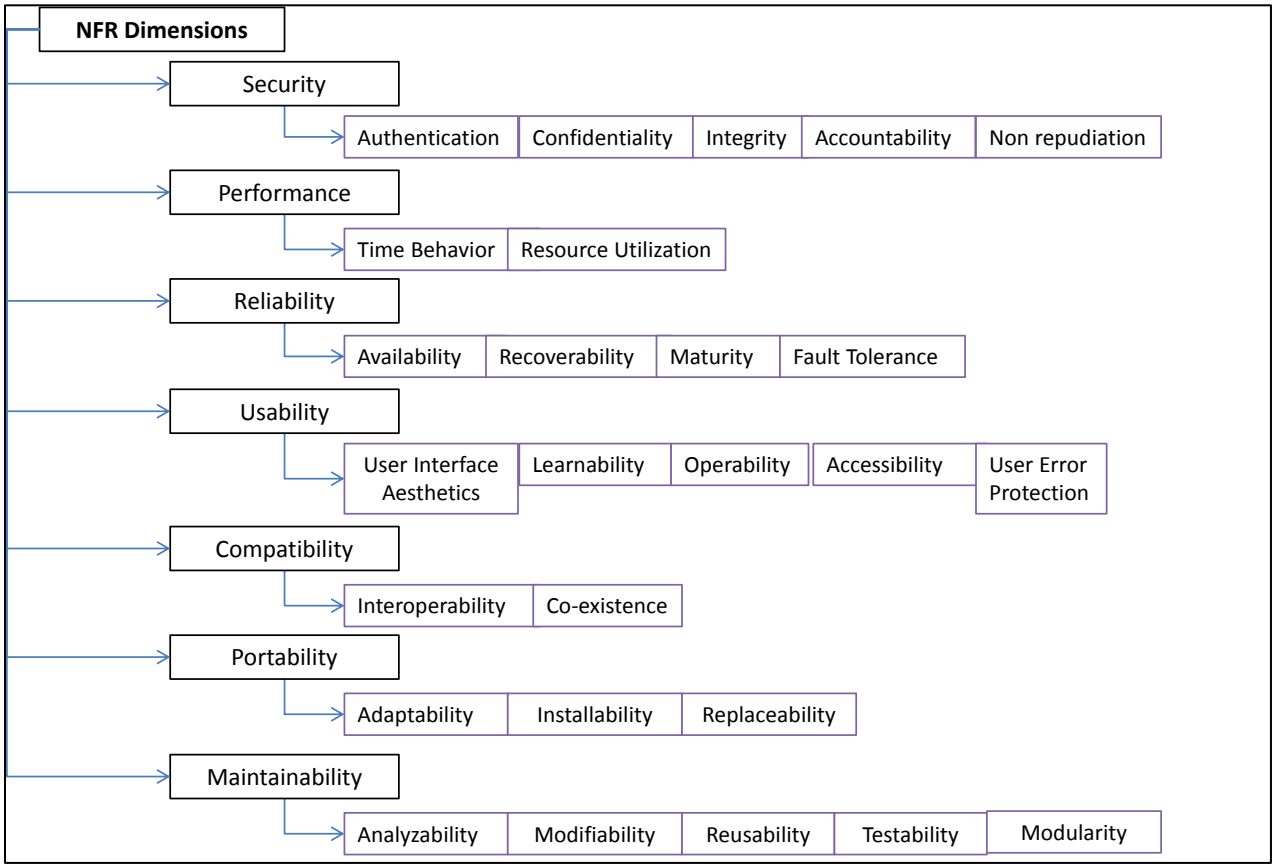

Figure 1: NFR Dimension classification based on ISO 25010 Product Quality Model

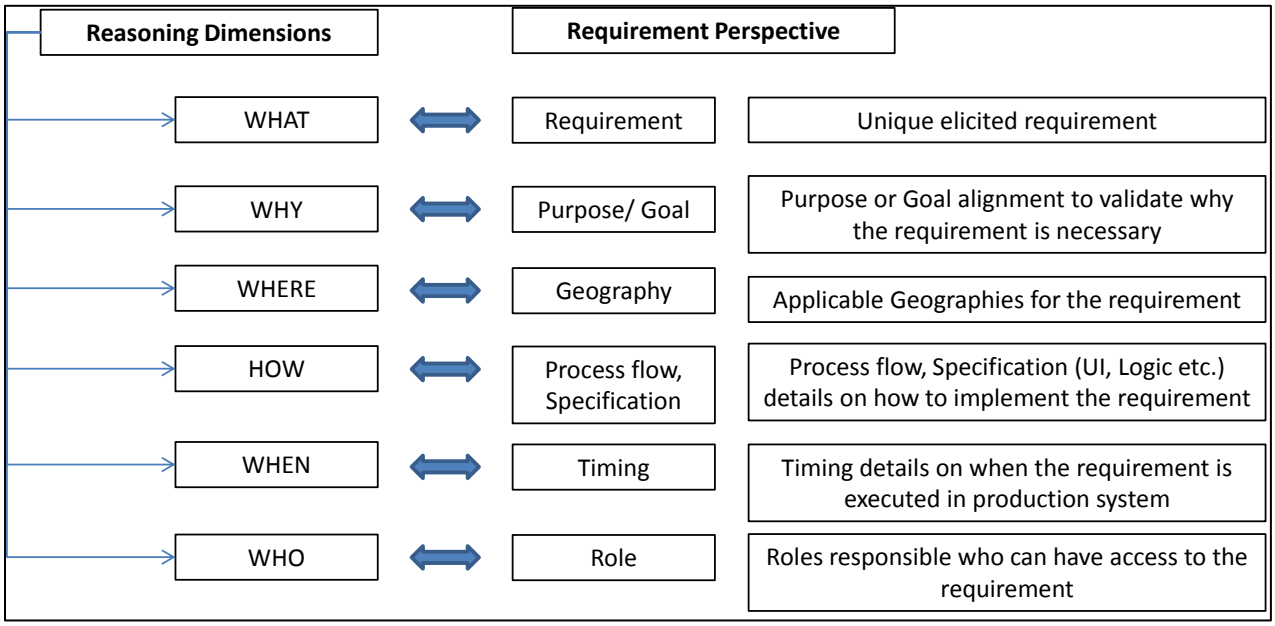

Figure 2: Reasoning Dimension classification based on 5W1H

Stated requirements are analyzed for each dimension of 5W1H classification and gaps found in requirements with respect to these dimensions are logged into Table 1 Requirement Coverage Tracker (RCT) as inconsistencies. If requirement information cannot be found for a reasoning dimension, it is marked as a coverage gap. For example: 'where dimension - applicable geographies' information is not specified for a requirement, then 'geography dimension' is marked as a gap and entered into Table 1 under head ' Geography dimension' as 'applicable geography information not specified'.

c) *Coverage for Domain specific dimensions:* If project has any specific domain or regulatory compliance standards, coverage analysis may have to be performed with respect to those dimensions and standards. Gaps found in requirements with respect to any of the compliance regulations are logged into Table 1 as inconsistencies. For example, in healthcare domain, compliance to HIPAA (Health Insurance Portability and Accountability Act) is a requirement and gaps found with respect to coverage of such regulations are logged under 'Domain dimension'.

d) Consolidate Requirement coverage inconsistencies: Table 1 outlines structure of Requirement Coverage Tracker (RCT). The gaps in requirements coverage with respect to the three dimensions are logged into Table 1. Each row in this table captures a requirement inconsistency along with meta-data of requirement and coverage dimension. The columns in Table 1 are requirement source (can be business function/ process/ user story), requirement id, description of gap/ inconsistency under an appropriate requirement dimension, new requirement flag (yes/no) and count of inconsistencies for a requirement. It is required to discuss the inconsistencies with subject matter experts and ensure resolution of all identified inconsistencies. The last column is 'resolution', intended for outlining the agreed resolution for the inconsistency: a complete requirement or reason for cancelling out.

Table 1: Requirement Coverage Tracker, RCT

Req. Source	Req. Id	Requirement Inconsistency Description			New Req.?	Count of Gaps	Resolution
		NFR. Dimension	Reasoning. Dimension	Domain Dimension			

Consequences: This pattern helps to strengthen requirements by identifying additional requirements that have to be elicited to ensure coverage of requirements with respect to analyzed dimensions. Application of an international standard for product quality ensures standardization of NFR requirements elicitation. It could become difficult to manually maintain the table for large number of requirements and across many projects, so it is recommended to integrate these patterns into the overall requirement process and tooling platforms.

Related Patterns: Requirement traceability analysis pattern. These two patterns complement.

Known Uses: The pattern has effectively been applied in multiple projects especially for NFR coverage analysis. Approach and case results of implementing NFR analysis using NFR classification scheme for an agile project is detailed in a paper [Nistala3].

2.2 Requirements Traceability Analysis

Pattern Name: Requirement Traceability Analysis

Context: This pattern is applicable to projects during requirements phase for review and analysis activities of stated requirements provided by customer. It is particularly useful when teams have to validate the requirements to ensure consistency of specification across multiple requirements artefacts, and to verify that there is enough detail in specification to proceed for next phase.

Problem: It is not clear to requirements engineer if the requirements stated across multiple requirement artefacts are aligned and what are the gaps in requirement specifications in order to proceed for design phase. Not being able to clearly see the traceability path of requirements: source of requirements and corresponding implementation specification, makes it difficult to identify consequences of changing a requirement and identify dependencies, Many clarifications are sought during later phases leading to delays and loss of productive development time.

Forces: Project factors that favor usage of this pattern are: need to review requirements and provide estimations in a short span of time, project team's lack of experience to analyze inconsistencies among various requirement artefacts and ask relevant questions during requirement workshops and reviews, late identification of requirement gaps, unavailability of client subject matter experts post requirements phase, and lack of any structured method adoption for analyzing requirements.

Solution: A software requirements specification is traceable [IEEE830] if (i) the origin of each of its requirements is clear and if (ii) it facilitates the referencing of each requirement in future development or enhancement documentation. In this pattern, we apply traceability check for requirement composition across project requirement layers in both forward and backward directions. Backward traceability check enforces the requirements are validated for alignment to overall purpose and goals of the system, and forward traceability check to field/ logic specification verifies adequacy of implementation guidance to proceed for design.

Requirement traceability analysis in this pattern is carried out in following steps:

a) *Identify project requirement traceability elements:* In projects, requirements information is generally specified as multiple layers of information across artifacts such as project charter, User Requirement Specification (URS) and Software Requirement Specification (SRS). Each artefact consists a set of requirement elements, for example project charter outlines a set of business goals to be achieved, URS states list of business requirements, SRS specifies implementation elements

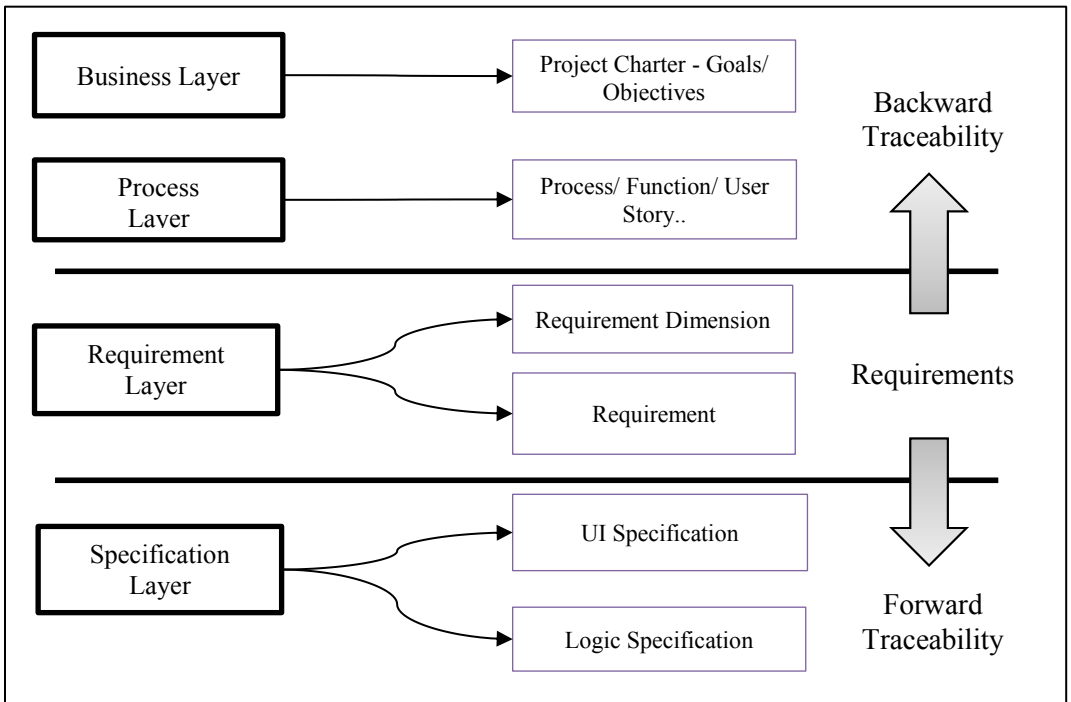

Figure 3: View of Requirement Traceability Layers and Elements

such as user interfaces, business rules and so on. A generic requirement composition can be considered as encompassing elements across four key layers: i) business, ii) process, iii) requirement and iv) specification [Nistala1]. Figure-3 depicts this composition demarcating backward and forward traceability elements. They are

[1]. Business layer element: Business goals or objectives (ref: project charter or proposal)
[2]. Process layer elements: Business process or function or user story (ref: URS)
[3]. Stated requirements (ref: URS)
[4]. Specification elements: Implementation specifications related to user interface, business rules and so on (ref: SRS or Use cases)

Specific project artefact titles and requirement elements may vary from project to project but these four layers of abstraction are found to be necessary and applicable in multiple project contexts. In this first step, project specific requirement traceability elements are identified for the four layers and outline of Requirement Traceability Matrix (RTM) is created. Table 2 outlines a reference template.

b) *Populate Requirements into Traceability Matrix:* Stated requirements details (functional, non-functional, regulatory and so on) as identified from various project artefacts are populated into Requirement Traceability Matrix, RTM cells: Req. source, Req. Id, and Stated requirement.

c) *Establish Backward Traceability for Requirements:* For backwards traceability, every requirement needs to contribute to project charter in terms of partial or complete realization of business goals. To trace backwards, it is required to analyze and establish traceability for each requirement to one or more project goals and business processes. Primarily this analysis validates the requirements in terms of 'why' is it required. For each Req. Id, corresponding business (e.g. goal) and process layer (e.g. user story) traceability elements are updated in Table 2. RTM cells. Sometimes we may find requirements that align poorly to project goals and might only represent a pet idea or wish list of one of the creators. If no alignment exists with business and process layers for a requirement, gaps detected are updated in corresponding RTM cells for that Req. Id.

d) *Establish Forward Traceability for Requirements:* For forward traceability, specification elements need to describe system implementation guidance for requirements in terms of process steps, user interface, computation logic and so on. Primarily this analysis verifies that the 'how' part is outlined in specification and each requirement is implementable. For each Req. Id, corresponding specification elements are updated in Table 2. RTM cells. Gaps found in analysis are marked as inconsistencies for that layer in RTM.

e) *Consolidate Requirements Traceability Data:* RTM consolidates the complete trace of requirements from its origin to implementation. Table 2 outlines structure of Requirement Traceability Matrix (RTM). Each row in RTM corresponds to a requirement. The columns in Table 2 are requirement source (can be business function/ process/ user story), requirement id, requirement element(s) for business layer, requirement element(s) for process layer, stated requirement, requirement element(s) for specification layer, trace flag, and count of inconsistencies for a requirement.

Gaps in cells in RTM denote gaps in requirements and need to be raised as inconsistencies and acted upon. If a requirement row has all columns populated with no gap entries, then it is marked as 'traced' (Yes), else 'not traced' (No).

Table 2: Requirement Traceability Matrix, RTM

Req. Source	Req. Id	Business layer	Process layer	Requirement layer	Specification layer		Trace Yes/ No	Count
		Goal	User Story	Stated Requirement	Field Spec	Logic. Spec		

Consequences The pattern provides a structured way to create well-formed requirement traceability matrix with both backward and forward traceability. It helps to validate requirements alignment to business goals and detect many inconsistencies in requirement specification which may go undetected

until later phases. It could become difficult to manually maintain the tables for large number of requirements and across many projects, so it is recommended to integrate these patterns into overall requirement process and tooling platforms.

Related Pattern: Requirement coverage analysis pattern. These two patterns complement.

Known Uses: The pattern has effectively been applied in multiple projects for traceability analysis of requirements for business alignment and specification traceability. Approach and case results of implementing requirements consistency analysis using requirement layers and configuration elements for an eRegistration system is detailed in a paper [Nistala1].

3. CASE STUDY

The two patterns have been applied to a pilot project, enterprise data-warehousing reporting system which had a set of user stories with stated requirements. From the given set, six user stories are taken for pilot, covering 35 requirements. For illustration purpose, user story US1 'Revenue run out report' is considered. This report should be available to a set of managers to view the revenue run out of upcoming and past revenue for service contract business.

This user story has stated requirements on filtering criteria of report; displayed invoice amount; information to be displayed on revenue, billing days and so on; extraction from source database, multiple currencies required and ability to export 1000 records to excel.

3.1 Pattern 1: Requirements Coverage Analysis

On analyzing requirements with respect to reasoning dimensions, it is detected that role dimension (who), timing (when) and specification part (how) are not covered in requirements. Other reasoning dimensions could be traced. Hence requirement gaps exist with respect to 'who' needs to run the report and 'when' report needs to be run and 'how' to implement.

Similarly, on conducting NFR coverage analysis on requirements, it was found that some key NFRs are missed out. Gap in security requirements on roles is already discussed and performance targets for the report were not stated. These gap entries are logged into Requirement Coverage Tracker, RCT. For each row, new requirement flag is marked or counter is incremented with number of gaps for each row. Table 3 depicts a partial view of RCT.

Table 3: Partial View – Requirement Coverage Tracker (RCT)

Req. source	Req. Id	Requirement Coverage Analysis			New Req.?	Count of Gap	Resolution
		NFR Dimension	Reasoning Dimension	Domain Dimension			
US1	R36	Security requirements missing – access role information for the report is not specified.			Yes		New req. for Access controls of report: access only to Finance manager.
US1	R03	Performance: Target response time to export 1000 records is not provided				1	Update req.: Response time targets agreed for the report.
US1	Multiple		Geography: Specific rules for geography not provided			1	Update req. USA and Australia are applicable geographies.
General	R37			DPA: requirements related to mandatory data protection act compliance not specified.	Yes		Applicable clauses of DPA added as req.

In the specific case, the consequence of applying the pattern was requirements count increased from 35 to 63 requirements, resulting in 80% improvement in coverage which designates a good expansion of baselined requirements.

3.2 Pattern 2: Requirements Traceability Analysis

Partial view of traceability of project requirement elements consolidated into RTM is depicted in Table 4. The cells marked in red colour indicate inconsistencies in requirements either due to lack of alignment with business layer or partial specification due to missing field mapping or logic rules. The requirement traceability elements identified specific to the project are Req Id, Goal, User story, Requirement, Field spec and Logic spec. Filed and logic specs are specific to the reporting platform project. These elements are configured in RTM.

Table 4 Partial View - Requirement Traceability Matrix (RTM)

Req. Source	Req. Id	Business layer	Process layer	Requirement layer	Specification layer		Trace Yes/ No	Count
		Goal	User Story	Stated Requirement	Field Spec	Logic. Spec		
US1	R1	Goal1	US1 Revenue runout report	Filter criteria for report : Contract#, Customer, Invoice#, Billing Start, Contract End and Invoice Amount	Field mapping from source database is missing for Invoice amount		No	1
US1	R4	Goal1	US1 Revenue runout report	Currency amounts in USD, AUD, EUR		Calculation logic on currency conversion is not specified	No	1

Out of 35 requirements, only 7 requirements could be traced forward and 28 requirements had gaps in field or logic specification. RTM is updated with number of requirement gaps for each requirement.

3.3 Evaluation

To evaluate the pattern from cost benefit perspective, effort spent on applying the pattern is logged and effort to fix the inconsistencies is estimated. Based on these effort calculations, return on investment of applying the pattern is computed.

Table 5 summarizes the findings from implementing requirements coverage analysis and traceability analysis patterns. For the case in consideration, system had a total of 35 stated requirements: 32 functional requirements and 3 NFRs. On application of coverage analysis pattern, 13 new functional and 15 NFRs were identified. On application of traceability analysis pattern, only 4/ 32 functional and 3/ 3 NFR requirements could be traced and specification inconsistencies exist for 28 requirements. These inconsistencies are in terms of gaps in specs: 45 field and 39 logic specs. Based on average project productivity figures, unit efforts to implement a new requirement and fix a requirement inconsistency are taken.

Development Effort Saved: This is computed as difference between 'Effort spent in consistency analysis' (A) and 'Effort required to fix the inconsistencies' (B).
A is computed as sum of effort spent in analyzing the user stories as per pattern. For each inconsistency in RCT, fixing effort is identified and summed up for all entries (B).

Table 5 Efforts to Fix

	Stated Req.	New Req.	Traceable Req.	Req. Gaps	Unit Effort to fix (pd)	TotalEffort (pd)	Remarks
Effort to apply pattern [A]						**65**	For six user stories
Effort required to fix gaps							
Functional Requirements	32	13	4	28			
NFRs	3	15	3				
Sum [B1]	**35**	**28**			**7.5**	**210**	28*7.5, To implement new requirements
Field Specs			223	45			
Logic Specs			9	39			
Sum [B2]				**84**	**2.5**	**210**	84*2.5, To fix gaps in requirement specs
Sum Total: Fixing effort[B]						**420**	B1 + B2

Effort taken to apply the patterns is 65 hrs. Total effort to fix the inconsistencies is computed in Table 5 as 420 hrs. A = 65 hrs, B = 420 hrs. Hence, Development effort saved = (B–A)/A = 546% indicating significant effort savings due to application of pattern.

4. CONCLUSION

Requirements gaps and inconsistencies are one of the biggest challenges in software development. If development teams are unable to take proper advantage of requirement workshops with subject matter experts, many clarifications in requirements arise in later phases resulting in idle time during development cycle, and reflect poorly on project team's competence.

We have proposed two patterns utilizing concepts of requirements dimensionality and traceability to provide development teams with a systematic approach to carry out requirements analysis. *Requirements coverage pattern* applies multi dimensionality concept to the mapping between requirements and stakeholder concerns: it analyzes how well the requirements provided cover the set of relevant quality concerns from ISO/IEC 25010 product quality standard, and whether the information associated with each function covers the concern space defined by the classical interrogatives. As can be seen from the *requirements traceability pattern*, traceability concept extends to capturing mappings across layers within various requirement elements, and that can be extended to design and construction as well.

The patterns, implementation and benefits of application are illustrated with an industrial case study. It can be seen that significant inconsistencies in requirements can be found which would not have surfaced till later phases in project life cycle. Project teams can use the requirements analysis data logged in tables: Requirement coverage tracker, RCT and Requirement traceability matrix, RTM to ask relevant clarifications during requirement grooming sessions with client subject matter experts. The practice generates data and metrics that can be used by project management to track requirements quality and identify problem areas. Planned future work includes wider deployment to demonstrate improvements in organizational capability, and development of tool to support implementation and facilitate analysis.

5. ACKNOWLEDGEMENTS

We thank our shepherds Azadeh Alebrahim and Christopher Preschern for providing quality feedback and guidance in shaping the paper. We are also grateful for the constructive suggestions by the EuroPLoP 2016 writers workshop participants. We also thank our TCS research colleagues,

specifically Bibhudendu Kumar, who contributed to the pilot and project teams that participated in case studies.

REFERENCES

[Cheng] Cheng, B. & Atlee, J. 2007. Research Directions in Requirements Engineering. In *Proceedings of Future of Software Engineering (FOSE '07)*. pp. 285 – 303. IEEE Xplore.

[Mahindra] Mahendra, P. and Ghazaria, A. 2014. Patterns in the Requirements Engineering: A Survey and Analysis Study. *WSEAS Transactions on Information Science and Applications*, *11*, 214-230.

[Henninger] Henninger, S. and Victor, C. 2007. Software Pattern Communities: Current Practices and Challenges, CSE Technical reports. Paper 52, http://digitalcommons.unl.edu/csetechreports/52,

[Withall] Stephen Withall. 2008. Software Requirements Patterns, Microsoft Press.

[PABRE] Palomares, C, Quer, C. & Franch, X. 2013. PABRE-Proj: Applying patterns in requirements elicitation. In *Proceedings of 21st IEEE International Requirements Engineering Conference (RE '13)*. IEEE Xplore.

[Nistala1] P. Nistala and P. Kumari. 2013. An approach to carry out consistency analysis on requirements: Validating and tracking requirements through a configuration structure, In *Proceedings of 21st IEEE International Requirements Engineering Conference (RE'13)*. IEEE.

[Hagge] Hagge, L. Lappe, K. 2004. Patterns for the RE Process, In *Proceedings of 12th IEEE International Requirements Engineering Conference (RE'04)*. IEEE.

[Jones] Jones, C. 2012. .Software Quality in 2012: A Survey of the State of the ART. http://sqgne.org/presentations/2012-13/Jones-Sep-2012.pdf

[Sommerville] Kotonya, G. and Sommerville, I. 1997. Requirements Engineering. John Wiley & Sons.

[Amber] Ambler, S.W. 1998. 'An introduction to process patterns', *Ambsoft WP http://www. ambysoft. com/processPatterns. pdf*

[Gamma] Gamma, E., Vlissides, J., Johnson R., and Helm, R. 1995. Design Patterns: Elements of Reusable Object-Oriented Software. Pearson Education, USA.

[POSA] Buschmann, F., Meunier, R. Rohnert, H., Sommerlad, P. and Stal, M. 1996. Pattern-Oriented Software Architecture Volume 1: A System of Patterns. Wiley series.

[Nistala2] Nistala, P. and Kumari, P. 2013. Establishing Content Traceability for Software Applications: An Approach Based on Structuring and Tracking of Configuration Elements. In *Proceedings of 7th International Workshop on Traceability in Emerging Forms of Software Engineering (TEFSE)*. 68–71. IEEE.

[ISO25010] ISO/IEC 25010: 2011. Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models.

[Nistala3] Nistala, P.V., Bharadwaj, A., and Kumari, P. 2013. An Approach to manage NFRs in Agile methodology: Expanding Product Roadmap to include NFR Features based on holistic view of product quality. In *Proceedings of ISSEC 2013 - Improving Systems and Software Engineering Conference.* Australia.

[IEEE830] ANSI/IEEE Standard 830-1984. Guide to Software Requirements Specifications.

[ISO24765] ISO/IEC/IEEE 24765:2010. Systems and software engineering--Vocabulary

[Kim] Kim, J., Son, J., Baik, D. 2012. CA 5W1H onto: ontological context-aware model based on 5W1H. International Journal of Distributed Sensor Networks.

[SQA] Nistala, P. V., Nori, K. V., Natarajan, S., Zope, N. R., & Kumar, A. 2015. Chapter 6: Quality management and Software Product Quality Engineering. in *Software Quality Assurance: In Large Scale and Complex Software-intensive Systems*, 133-150, Morgan Kaufmann

[Ikeda] T. Ikeda, A. Okumura and K. Muraki. Information Classification and Navigation Based on 5WIH of the Target Information", Proceedings of the 17th International Conference on Computational Linguistics, vol. I, pp. 571-577, ACM.