SReYantra: Automated Software Requirement Inter-dependencies Elicitation, Analysis and Learning

Gouri Deshpande
Dept of Computer Science
University of Calgary, Canada
Email: gouri.deshpande@ucalgary.ca

***Abstract*—Requirements elicitation is a cognitively difficult task. Rich semantics in natural language based requirements impose challenges in elicitation, analysis and maintenance of requirement inter-dependencies. The challenges intensify further when dependency types and strengths are considered. Ignoring inter-dependencies can adversely impact the design, development and testing of software products. This PhD research proposal addresses three main challenges. First, Natural Language Processing (NLP) is studied to automatically extract dependencies from textual documents. Further verb classifiers are utilized to automate elicitation and analysis of different types of dependencies (e.g: requires, coupling etc). Second, representation and maintenance of changing requirement dependencies from designing graph theoretic algorithms will be explored. Third, the process of providing recommendations of dependencies will be studied. The results are aimed at assisting project managers to evaluate the impact of inter-dependencies and make effective decisions in software development life cycle.**

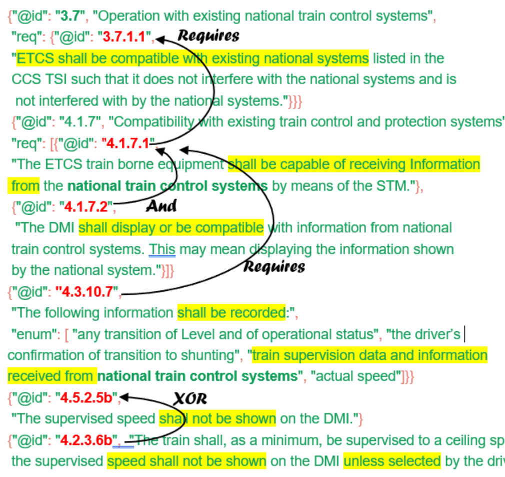
***Keywords*-Requirements Engineering, Requirements Inter-dependency Management, NLP, Machine Learning.**

I. Introduction

Existing research has extensively explored the effects and importance of requirements inter-dependencies on the design, development, and testing of software products. The term dependency can have a different connotation in implementation, feature efforts, feature value and feature usage space [1]. A few of the widely discussed dependencies types are Requires, Coupling, Either/OR, And and XOR. Although identifying these dependency types is crucial, due to rich semantics and inherent ambiguity of natural language (NL) based requirements, large collections of requirements involve considerable efforts and frequent involvement of domain experts. Additionally, dependencies management and re-analysis of the dependency network imposes challenges in the advent of the changed requests or new requirements.

Figure 1 shows the implicit requirement dependencies at various levels in a Software Requirements Specification (SRS) document (consisting 45 functional requirements, represented hierarchically) for the European Rail Traffic Management System (European Train Control System)[2].

Requirement inter-dependencies are addressed from the requirements traceability perspective, however, nature and type of dependencies are fairly unexplored [3]. Current traceability tools provide means to store a relationship between requirements but they provide very little guidance regarding the semantics, inherent meaning and consequences of a relationship [3]. **Our research** focuses more on the



```
{"@id": "3.7", "Operation with existing national train control systems",
 "req": {"@id": "3.7.1.1",
 "ETCS shall be compatible with existing national systems listed in the
 CCS TSI such that it does not interfere with the national systems and is
  not interfered with by the national systems."}}}
 {"@id": "4.1.7", "Compatibility with existing train control and protection systems",
 "req": [{"@id": "4.1.7.1",
 "The ETCS train borne equipment shall be capable of receiving Information
from the national train control systems by means of the STM."},
 {"@id": "4.1.7.2",
  "The DMI shall display or be compatible with information from national
 train control systems. This may mean displaying the information shown
 by the national system."}]}
{"@id": "4.3.10.7",
 "The following information shall be recorded:",
 "enum": [ "any transition of Level and of operational status", "the driver's
confirmation of transition to shunting", "train supervision data and information
received from national train control systems", "actual speed"]}}
{"@id": "4.5.2.5b",
 "The supervised speed shall not be shown on the DMI."}
{"@id": "4.2.3.6b", "The train shall, as a minimum, be supervised to a ceiling speed;
 the supervised speed shall not be shown on the DMI unless selected by the driver."}
```

Figure 1. Representative requirements inter-dependencies for European Rail Traffic Management System [2]

semantics, i.e. *how* requirements effect each and *not only that* they affect each other.

Unlike requirement elicitation, requirement inter-dependency elicitation lacks a substantial body of knowledge. Research on the subject has been mostly limited to classification schemes for inter-dependencies [4][5] and identification of these dependency types through NLP methodologies [6] on pair-wise analysis of the requirements on an unknown data set. Now that we have access to a public data set of SRS documents [2], it would be worthwhile to explore the requirement dependencies in terms of multiple requirements with varying degree of dependency in the course of their dynamic changes.

In this research, we will utilize NLP in the initial stage to generate annotated data set to identify the semantic inter-dependencies and then further train machine learning algorithms to identify all the inter-dependencies to provide decision support to domain experts in this task. Thus generated Machine Learning (ML) can be utilized as a prediction system for processing change requests as well as new requirements. Further, we will store this dependency information in a graph database for dependency network analysis. This approach will amount to reduced time for re-analysis as the re-computation will go through an automated process during consecutive iteration, as explained before.

It is imperative that we will begin with the supervised learning algorithm first in the solution approach which eventually will be supported by a substantial amount of data annotation by human experts. Hence, we argue that this solution framework can be potentially used as a plugin to elicit requirement inter-dependencies for a similar project.

II. Research Questions (RQs)

This research is organized around three main research questions.

A. *How effective and accurate NLP is in detecting semantic requirements dependencies? (RQ1)*

Although the use of NLP has been considered to detect the inter-dependency types in [6] and [7], their complete research is theoretic in nature and focuses from an aspect-oriented requirement engineering perspective which is based on a single verb classification [8].

We will thoroughly evaluate the verb types and semantic roles identified by multiple verb classifications [8] [9], whether they are sufficient and, if not, what new semantics need to be introduced. We will develop a library of verb semantics, including all potential variants of usage, such that it relates to the appropriate semantics of software engineering specific research. Eventually, we will apply an ML algorithm (such as Random Forest and Naive Bayes) to predict the dependency type for new and changed requests.

Approach: Soon after initial text pre-processing to eliminate special characters, the textual data will be parsed with the Stanford Parser[1] to record all verbs with the subject, object, or prepositional typed dependencies. Further, OpenNLP4[2] will be used to resolve the entity mentions. Finally, the verb classification, adapted to Software Engineering, will be looked up to determine the dependency type.

B. *How to represent and maintain the requirement inter-dependencies in case of change requests? What are the implications for the evolving inter-dependency network in structure and dependency strength? (RQ2)*

New requirements and change requests can lead to changes in inter-dependencies, which could lead to conflicts within requirements and increase or decrease the importance of other dependent requirements. We are interested in designing a near-optimal method for identifying such occurrences and providing suggestions for resolution. Working further on the output generated from RQ1, we will store the dependency information in a graph database and utilize graph theory based algorithms to detect negative cycles to identify the conflicting requirements.

Approach: Since graph database helps leverage data relationships and stores relationship information as a first-class entity, we will store the complete requirements data into a graph database, such as Neo4j[3] to facilitate continued dependency detection and analysis. Additionally, to provide a powerful model execution environment for capturing changes in the dependency graph and its implications on the rest of the requirements, in this work, we will utilize graph theory based algorithms coupled with requirement priority information. By representing the requirements as nodes and dependency types as arcs, we can detect negative cycles in this graph representation using the *Depth First Search* algorithm. Also, additional node information, such as requirement priority will be utilized to propose the resolution for the conflicting requirements.

C. *Can we utilize the prediction system (RQ1) and analysis platform (RQ2) as a plugin to jump-start a similar project? (RQ3)*

To further automate elicitation of inter-dependencies, we will utilize already developed intelligent system across (similar) projects. Based on the results from RQ1 and RQ2 results, we will evaluate the effectiveness of this system on a requirements document of a project from a similar domain.

Approach: Building upon the research work from RQ1 and RQ2, a *ML algorithms engine* will be designed to utilize already harnessed data in the graph database. Thus, a classification model will be developed using the existing data for training algorithms such as Random Forest and Naive Bayes. Well-known NLP tasks including part-of-speech tagging, chunking, named-entity recognition, learning a language model and the task of semantic role-labelling will be employed before training the algorithm.

SReYantra will be evaluated on public data sets [2] first. Subsequently, the evaluations will be carried out on the real world data from three industry partners[4].

References

[1] G. Ruhe, *Product release planning: methods, tools and applications.* Auerbach Publications, 2010.

[2] A. Ferrari, G. O. Spagnolo, and S. Gnesi, "Pure: A dataset of public requirements documents," in *Requirements Engineering Conference (RE), 2017 IEEE 25th International*, pp. 502–505, IEEE, 2017.

[3] Å. G. Dahlstedt and A. Persson, "Requirements interdependencies: state of the art and future challenges," in *Engineering and managing software requirements*, pp. 95–116, Springer, 2005.

[4] P. Carlshamre, K. Sandahl, *et al.*, "An industrial survey of requirements interdependencies in software product release planning," in *Proceedings Fifth IEEE International Symposium on Requirements Engineering*, pp. 84–91, Aug 2001.

[5] P. Carlshamre, "Release planning in market-driven software product development: Provoking an understanding," *Requirements engineering*, vol. 7, no. 3, pp. 139–151, 2002.

[6] R. Chitchyan and A. Rashid, "Tracing requirements interdependency semantics," in *Early Aspects*, 2006.

[7] N. Weston, R. Chitchyan, and A. Rashid, "Formal semantic conflict detection in aspect-oriented requirements," *Requirements engineering*, vol. 14, no. 4, p. 247, 2009.

[8] R. M. Dixon, *A semantic approach to English grammar.* Oxford University Press, 2005.

[9] B. Levin, *English verb classes and alternations: A preliminary investigation.* University of Chicago press, 1993.

[1]http://nlp.stanford.edu/software/lex-parser.shtml

[2]http://opennlp.sourceforge.net

[3]https://neo4j.com/why-graph-databases/?ref=footer

[4]Three industry partners: BrightSquid, Daimler and BellMTS