Distributed Software Requirement Specification- An Overview

Neha Bhateja
Department of Computer Science & Engineering,
Amity University Haryana,

Abstract: :- The requirement phase plays an important role in the software development process. This paper presents an overview of the distributed software requirement specification. It covers the various topics like characteristics, categories of requirement specification, requirement inconsistency.

Keyword: *Software Requirement Specification, DSRS, Inconsistency.*

*****

I. Introduction

In today's life, software became a vital component in business. To get the advantage of the competition in IT sector, each individual organizations depends on software. Software requirement specification plays an important role in the software development process. A SRS is a requirement definition document which is the first phase of any software development process. A Software Requirements Specification (SRS) presents the complete structure or behavior of the system which is to be developed. It describes all the ways by which a user can interact with the software. Or say, SRS is a two-way insurance policy that confirms that both the client and the organization understand each and every aspect of the requirements of the software.

The software project team is distributed geographically on a worldwide scale. This task is characterized as Distributed Software Development (DSD). Distributed Software Requirement Specification (DSRS) defined as a document that is prepared by collecting the requirements from different users with different viewpoints and from different geographical locations [1].A Software Requirement Specification is described in a natural language and is collected from the different stakeholders, which becomes a main reason that the requirement specification is inconsistent. In consistency refers as when more than one requirement or description do not follow the rules that are defined to hold them.

II. Characteristics of Software Requirements Specifications

To achieve the quality in Software Requirement Specification, the following characteristics must be in SRS [3].

- Complete: Completeness is the required feature of the requirement phase. No requirement should be missing because missing requirement is hard to locate. So the in the Software Requirement Specification, the requirement should be in organized manner so that it can help the reviewers to understand the functionality of the structure and easily locate the missing data.
- Consistent: Requirement phase should be consistent. Before proceeding to the development phase, the conflicts in the requirement must be fixed. Requirement document must be reviewed if some modification is done in the requirement.
- Modifiable: Keeping a record of the changes made in the requirement to achieve the unambiguity. This can be achieved by inserting the table of contents, indexes and cross reference links.
- Traceable: Every requirement should be linked with its origin (from its source), with the source code in the design phase and also linked with the test cases that are executed on it. This activity can be done by assign the labels to each requirement or by the apply bullet lists.
- Ranked: to make the design process more convenient, each individual requirement should be arranged according to the level of their security, level of implementation (easy or problematic), stability etc.
- Unambiguous: Each individual requirement that are defined in the Software Requirement Specification should represent unique significance. There is no conflict between them.
- Valid: Software Requirement Specification is valid in terms when all the individuals who are associated with it can understand, accept, analyze and approve it.

ISSN: 2321-8169
570 – 572

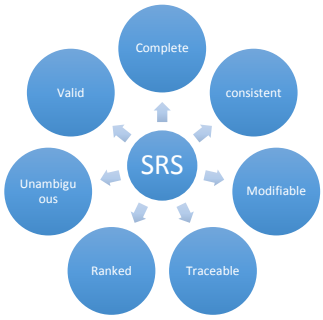

Fig: Characteristics of SRS

III. Categories of requirements engineering in Global Software Distribution

There are various factors that are involved in these categories [4]. These are as follow:

- Communication: The communication is the basic factor on which the requirements engineering process depends. Clear communication is must to avoid misunderstandings and conflicts. The components that are identified with communication are language, communication medium and time zone.
- Culture: the requirement engineering process can be effected by the culture of the organization and the individual team members. The primary variables identified with culture are context, state of mind and values.
- Knowledge Management: The requirements engineering process deals with a huge amount of data. The knowledge management skills are required to collect, process, store and use the data related to the requirements process. The elements related to knowledge management are awareness, expectations and management of cultural information.
- Technical aspects: In distributed environment, various technical features can affect requirements engineering process. The coordination and control mechanisms features are must in the requirement process. The identified variables are patterns, process and configuration management.

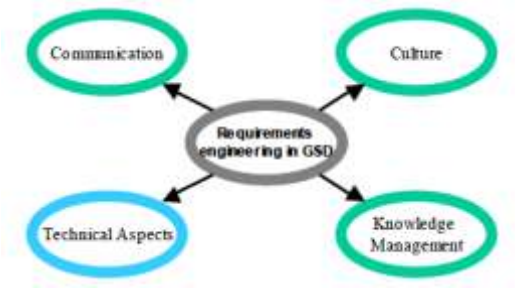

Fig: Categories in Global Software Distribution [4]

IV. Levels to Manage the Requirement Inconsistency

In requirement process, inconsistency can be managed at the following levels [5]:

- Inconsistency Detection: Inconsistency detection is picking up the violation of a consistency rule between two or more descriptions. Detection must be an automatic process, that does not need any support from users. Detection of inconsistency specified in policies or those arising due to addition, deletion or modification of descriptions.
- Inconsistency Diagnosis: The diagnosis process begins whenever an inconsistency is detected. Diagnosis help to determining what parts of a description have broken a consistency rule and Identifying the cause of an inconsistency.
- Inconsistency Handling: Inconsistency Handling involves actions taken in response to the inconsistency encountered. Handling actions can range from simple actions, such as modifications to descriptions, to complex handling actions like ignoring, defer etc.
- Inconsistency Tracking: Inconsistency Tracking is the recording of the events and actions, and information associated with inconsistency. Details to be recorded about inconsistency are (a) the reasoning for the detection of an inconsistency (b) the source and cause of inconsistency (c) the handling actions that were considered and (d) the arguments for the decision to select one of these options and reject the other.
- Inconsistency Measurement: Inconsistency Measurement is needed to ensure efficient and effective inconsistency management. Measures are needed such as benefit and risks related with the numerous handling actions of inconsistencies. Measures relating to make a decision on which choice is “more consistent”.

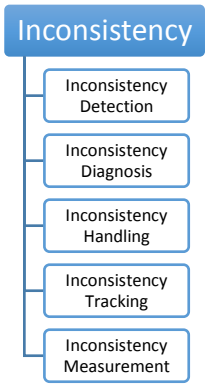

Fig: Management levels of Inconsistency

a. Classes of inconsistency

The classification of requirement inconsistencyis defined at three levels i.e. process, product and instance level [6]. Based on these three levels, classes of inconsistency are defined as follow:

- a process-level- It is a state evolution in the requirements engineering (RE) process that results in terms of inconsistency between the rules of process-level and the process state at the product level;
- an instance-level - Itis a state transition of a running system that results in terms of inconsistency between a product-level requirement and a specific state of the running system;
- a terminology clash- It occurs when a single real-world concept is given different syntactic names in the requirements specification;
- a designation clash- It occurs when a single syntactic name in the requirements specification designates different real-world concepts;
- a structure clash- It occurs when a single real-world concept is given different structures in the requirements specification;

Conclusion:

DSRS plays a vital role in the process of software development as project team members are distributed globally to collect the data from various stakeholders and from various locations. Requirement engineering is the first phase of software development process, so it should be consistent.

References:

[1] M.Kamalrudin and S. Sidek, " A Review on software Requirement Validation and Consistency Management," International Journal of Software Engineering and its Applications, Vol.9, No.10, pp:39-58 2015.

[2]Ivo Krka and Nenad Medvidovic, "Distributed Refinements of a system-level partial Behavior Model", IEEE Explore,2013

[3] Donn Levie, Jr. "Writing Software Requirement Specification", Whitepaper.

[4] Leandro Loper, Rafael Prinklannicki and A.Majdenbaum: "Requirement Specification in Distributed Software development": a Process Proposal (2008).

[5] K.Mu, Weiru Liu, Zhi Jin and David Bell:" Handling Inconsistency In Distributed Software Requirements Specifications Based On Prioritized Merging"; Fundamental Informaticae 91(2009) 1-40, DOI 10.3233/FI-2009-2008, IOS Press.

[6] K.Mu, Weiru Liu, Zhi Jin and David Bell:" A Merging-Based Approach to Handling Inconsistency in Locally Prioritized Software Requirements"; KSEM 2007, LNAI 4798, pp. 103–114, Springer-Verlag Berlin Heidelberg 2007.

[7] Irit Hadar and Anna Zamansky, " Cognitive factors in inconsistency Management", IEEE Explore 2015.

[8] M.Kamalrudin: "Automated Software Tool Support for Checking the Inconsistency of Requirements"; IEEE/ACM Int'l Conf. On Automated Software Engineering, 2009.