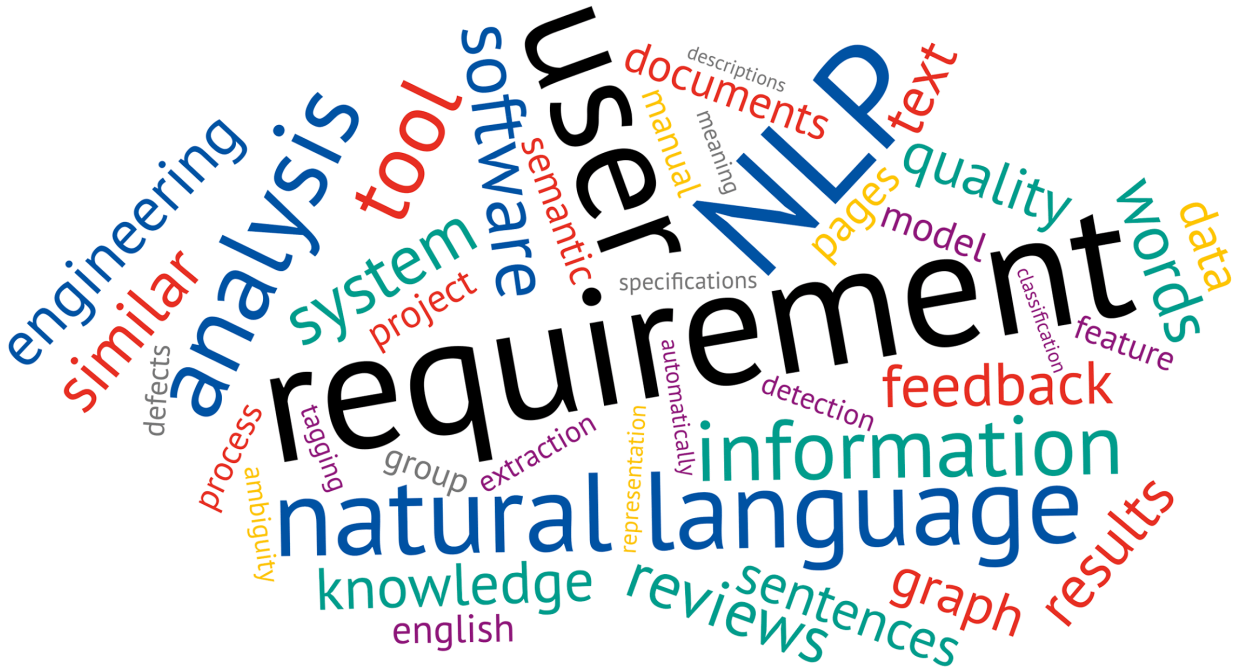
Software Engineering

IEEE Software Assignment Report



Group 16

Pragya Agrawal (2019BCS-040)

Richa Gupta (2019BCS-047)

Sriya Chettebhaktula (2019BCS-063)

Krishna Gandhi (2019BCS-079)

How Common Is Common Enough in Requirements Engineering Practices

Abstract**—** *The key is to figure out the requirements-engineering activities are "common enough" without being too rigid or too loose in a complex organization where different groups can have different needs.*

I. Introduction

The process of collecting the software requirements from the client then understanding, evaluating and documenting it is called as requirement engineering. In this report we are going to take a closer look at these requirement engineering methods and their 'Commonality '.

II. Summary of paper

The two threats faced by RE are Dogma-set of principles, and Corruption-breaking of the set of authorized rules.THe solution to these threats is given by the Layers of Product Development Process.
At the program stage, a standard structure helps everyone to understand goals and deadlines. The phrase "common enough" appears in the following layers of the product, alluding to the various memberships that are often at work in product creation. To achieve a common understanding of the system in progress, these barriers must be overcome.

III. Methodology Of Paper

Requirements engineering is the basis on which managers estimate cost, developers build system and system designers design the architecture in the system. Any defects in RE is not encouraged as it makes an impact in the future development of software.

A. Problem Stated

There are many methods to practice RE. Different types of companies and different types of departments need different RE practices. They are influenced by factors such as:

- corporate culture
- product criticality
- potential risks of missed requirements

These factors outline the 'range of rigour' through which the company's needs are met.

The companies face different dilemmas:

- Companies which develop security related products need high maintenance RE practice.
- Whereas the companies which develop products which have high error tolerance can opt for some lightweight RE practice.

The requirements practice must clear the two dangerous threats:

- *Dogma:* Dogma is the set of principles laid down by an authority as incontrovertibly true. Declaring such principles might cause the risk to the complete requirement process if the product's basic orientation is not always the same.
- *Corruption:* Corruption directly points to breaking of rules and when any of the process or practice are corrupted, the outcome is undesirable.The hazard of corruption—workarounds, missed steps, and bad practices—increases when the hazard of dogma is discovered, adherence is required blindly, without concern for usefulness or the impact on the program or team.

Thus it raises the question "how common is common enough?"

B. Solution Stated

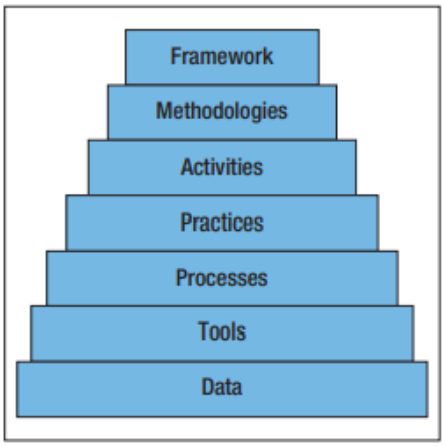


Figure depicts : Layers of Product Development Process.

1. **Framework:**
   Software Process Framework is a foundation of complete software engineering processes Software.

   The process framework includes all sets of umbrella activities. It also includes a number

of framework activities that are applicable to all software projects.

Demanding absolute consistency in all areas might lose important domain and involve unnecessary effort. Not enough commonality will produce chaos in which each group.

2. **Methodologies:**

   Software development methodology is a process or series of processes used in software development.

   Dogmatic insistence on outlining all the requirements before beginning any design and development or treating those parts as a necessarily linear activity is an indication of too much commonality. No assessment of the activities at all to determine what teams will do to satisfy their objectives suggests insufficient commonality.

3. **Activities:**

   RE is broken down into its constituent operations using a variety of schemas.

   Elicitation, review and validation, specification, verification, and management are the operations that are distributed. There was no evaluation of the tasks to decide what the teams would do to achieve their goals, implying a lack of commonality

4. **Practices:**

   Practices are the measures we take in order to complete the activities. They are the skills and strategies that are applied to a project or program to achieve a specific goal.

   When someone asserts that because a particular practice worked before on a qualitatively and quantitatively different program, it will certainly work on the present one, it is a sign of too much commonality. On the other hand, when each team member is free to determine how to do his or her work, inconsistent practices can lead to corruption of the work in the activity.

5. **Processes:**

   Processes ensure that work is completed in an efficient manner and that contradictory procedures do not contribute to work misconduct in the operation.

6. **Tools and Data:**

   The tools and data—the practices' repositories and outputs, respectively—show that the RE practices were completed successfully.

C. Result

"It Depends". In an enterprise with autonomy among the different organizations, finding a common-enough solution in any layer can be elusive. We still cannot define 'common enough'. The specific factors on which the answer will depend can vary but we are often found at the points where the different boundaries can be crossed.

IV. ADVANTAGES AND SHORTCOMINGS

The article "How common is common enough?" deals with various questions related to Requirement Engineering (RE) and helps in understanding the layers of the product development process. It answers the following questions -

- ❏ How much RE should one teach?
- ❏ What template(s) should be recommended?
- ❏ Where can one demand more rigor and precision
- ❏ when might teams be counseled to adopt lightweight RE practices?

Below, we discuss the various pros and cons which have been addressed in the paper.

A. Advantages

This article has been found to be beneficial in several different contexts. It explains the significance of RE as one of the most important phases of the Software Development Life Cycle (SDLC).

- A common framework allows everyone to understand milestones and deadlines at a program level. Overall results improve.
- Having a common enough process has an advantage of uniformity.
- The importance of RE is enormous to develop effective software and in reducing software errors at the early stage of the development of software. Since RE has a great role in different stages of the SDLC, its consideration in software development is crucial.
- It is used to translate the imprecise, incomplete needs and wishes of the potential users of software into complete, precise and formal specifications. The specifications act as the contract between the software users and the developers.
- It helps in focusing on where / how to address commonality or lack of the requirements engineering process and practices.

- RE plays a fundamental role in all sorts of software development processes.
- In a product development the common enough RE should have boundaries to reach a common understanding of the team under development. This understanding improves the efficiency of the project.
- In the layers of product development process, the highest being the framework which establishes the essential terminology used to communicate among team members across group boundaries. This communication enhances the project cooperation and speed.
- The intent of methodologies is to meet the intent of the frameworks objectives. Overall it simplifies the work.

B. Shortcomings

This model is advantageous over many reasons though it still has some disadvantages.

- The work of finding common enough requirements practice must steer clear of two potential threats: dogma and corruption.It is a difficult task to accomplish.
- The main problem for an organization is to find a practice that is common enough to apply across the whole enterprise.
- Importance of requirements management -
  - ❖ Requirements traceability to design and code unaccomplished
  - ❖ Modules don't integrate
  - ❖ Code is hard to maintain
  - ❖ Poor performance under load
  - ❖ Build-and-release issues
  - ❖ Ultimately, business needs are not met
- The challenge that the company/ enterprise faces is having a standard process or no process.
- Differences in common enough process will lead to variance. Due to the vast directions to follow, the projects turn out very similar. The 'unique' factor is lost.
- The key responsibility of an organization is to deliver products not having common requirements.
- Whenever the requirements engineers lack the knowledge of the performance and characteristics of the different elicitation methods, the activities related to requirements will fail, thus leading to wrong gathering of requirements.
- Not enough commonality will produce silos in which each group adopts / invents a locally optimized solution with little attempt to communicate across boundaries.

.

V. CRITICAL ANALYSIS

The paper addresses the issue of finding common grounds for requirements practice within the diverse teams having uncommon practices in a particular company to maximize product's success. When different groups gather for requirements elicitation and planning, usually there is no consistency and agreement as each team uses their own applications and tools which aren't similar to other teams although each one is correct. This suggests that in addition to having experienced requirements engineer and right RE techniques, there is a need for a common RE practice to avoid confusion and maintain consistency among wildly diverse teams, however having a commonality isn't always optimal for all the different teams so we aim to have a "common enough" strategy.

This concept allows new requirements practitioners, whether new to the discipline itself or experienced authors just joining a project team to understand the demands of the project and answer the most important question "What process must I follow here to create my requirements for this project?".

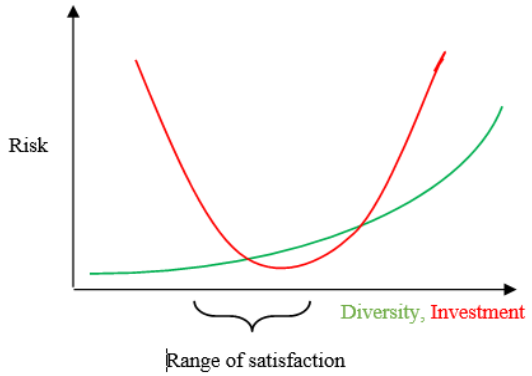
a) Basing decisions of requirement process on factors such as corporate culture, product criticality, including the potential risks of missed requirements, and the implications of failing to meet users' needs as described by Al Davis, gives an idea of the right level of process rigor and formality required for that particular project and therefore eliminates the overhead of mandating a heavy process which is beyond the needs of the project.

b) The commonality of any practice among adopting teams exists along a spectrum, with two relatively well-defined endpoints.

A proposed range for a common enough process thus spans the distance between

uniformity and diversity. In addition, too much investment in driving commonality may increase, rather than decrease risk.

Taken on their own, a specific set of right homogenous practices implemented uniformly will generally entail little risk. In addition, too much investment in driving commonality may increase, rather than decrease risk.



"Just enough" investment and common practice provides a satisfactory range of "enough-ness" that allows work to move forward at an acceptable level of risk.

c) As indicated by the author, even the best techniques are subject to dogma – the belief that rigid application of this approach will be the one that will save us from all of the broken promises of past practices – as well as corruption – the belief that we can simply go faster by shaving off bits and pieces of disciplined work. The author also answers how common practice must be to be "common enough" to move forward safely between dogma and careless practices using the metaphor of layer cake which is appreciable.

d) In RE process, requirements statements will likely never reach the point where "too common" is an issue. Requirements can be "not common enough" – insufficient commonality is a sign of false consensus, especially in a very heterogeneous environment.

VI. Suggestions and Improvements

The paper brings about the problem of finding a common enough solution in an enterprise with autonomy among different teams - "Common enough" invokes the many memberships that are often silently at work in product development.

Boundaries occur between organizations, groups, individuals, disciplines, phases, methodologies, conceptual models, datasets, activities, languages, and many others. Thus, what is common within a category may not be common when one crosses across a boundary into a different category. Therefore, we need to find ways to work across all these boundaries.

One very good way to work across boundaries is through use of protocols

- Establishing "code of correct conduct", or "how unrelated objects communicate with each other"
- Protocols should be relatively simple, yet work in very complex settings

It would be better if we view commitment as a negotiated, cross-boundary communication device rather than a strict contract for enforcement in RE practices.

In addition, the discipline and the areas in which the work takes place, as well as the ways they work themselves, are subject to ever more complexity. More complexity often leads to demands for more commonality. Within any given group, innovation and creative means of sensing and responding to complexity can yield good results.

VII. REFERENCE

1. T. Gilb and D. Graham, “Software Inspection”, Addison-Wesley, 1994.
2. A.M. Davis, “Just Enough Requirements Management: Where Software Development Meets Marketing”, Dorset House, 2005.
3. https://en.wikipedia.org/wiki/Requirements_engineering

VIII. INDIVIDUAL CONTRIBUTION

1. The research paper was read and a summary which includes a problem statement, Approachto solve the problem and result of the approach was made by **Krishna Gandhi** and **Pragya Agrawal.**
2. After studying the research paper, the advantages and shortcomings of the model were listed and noted down by **Sriya Chettebhaktula.**
3. After Going Through The research paper, a thorough analysis was done and the possible improvements were suggested by **Richa Gupta.**