# Requirements Evolution: Understanding the Type of Changes in the Requirement Document of Novice Designers

**Shraddha Joshi and Joshua D. Summers**

**Abstract** This paper presents the findings from a case study conducted to investigate the evolution in the requirements documents of novice designers (students). The case study was conducted with four student teams working in parallel on senior design project. At the beginning of the semester, all four teams were given the design problem and set of initial requirements by the sponsor. For the purpose of this case study, data was collected from all four teams in the form of weekly requirements update. These requirements were then investigated to explore the evolution of requirements and identify the changes in individual requirements from initial weeks to final week. The findings from this case study suggest that the requirements document of novice designers changes in multiple ways. Finally, a set of recommendations and guidelines are developed to help the novice designers to maintain the requirements document and manage the changes in the requirements document.

**Keywords** Requirements evolution · Case study · Novice designer · Capstone design

## 1 Case Study Motivation

The goal of this research is to investigate the evolution in the requirements document of novice designers. In studying how requirements evolve from the start to the completion of a project, engineering educators can identify the types of changes that occur throughout the design project. This knowledge can then be used for developing tools and methods to support the novice engineers for both engineering design process and project management aspects.

S. Joshi
Mechanical Engineering, Carnegie Mellon University, Pittsburgh, USA

J.D. Summers (✉)
Mechanical Engineering, Clemson University, Clemson, USA
e-mail: jsummer@clemson.edu

A. Chakrabarti (ed.), *ICoRD'15 – Research into Design Across Boundaries Volume 2*,
Smart Innovation, Systems and Technologies 35, DOI 10.1007/978-81-322-2229-3_40

In order to investigate the evolution of requirements, a case study is conducted with senior design students at Clemson University. Pattern matching is often used to study the analyzed data in case study research [1, 2] while counter-patterns are used to improve the qualitative objectivity [3]. Case study research has been used in the past to study novice engineers through capstone design projects at Clemson University [4–7].

The anticipated pattern for this study is that the requirements document of novice designers (students) will evolve in multiple ways; that there is no dominant mechanism for requirements change within a capstone project. This is based on the assumption that the students get feedback from the advisory committee throughout the project, thereby necessitating changes to the requirements. Moreover, these changes essentially illustrate the growth in problem understanding as the problem and solution co-evolve. Alternately, the counter pattern is that the requirements document will not change. This is based on the assumption that although the students receive weekly feedback, it is not specifically focused on the requirements document.

Capstone design class at Clemson University is a three credit, 15 week long class that falls at the end of the undergraduate curriculum and essentially serves as an exit exam for the mechanical engineering students. Student teams are created based on a balancing of expertise and work experience. Four teams of students are assigned to the same industrial sponsored project to work in parallel. The sponsored projects are funded to support prototyping, while serving as an incentive for the customer to be responsive to the team. The teams meet weekly with an advisory committee, consisting of graduate and faculty advisors. These meetings are design reviews where the teams get feedback on the project, but without direct management by the advisors. This course has been thus structured for roughly three decades, with many research studies derived from this course (e.g., [8]).

The case study to investigate the evolution of requirements in the requirements document of novice designers was conducted on one of the capstone projects from Spring 2011. The project was sponsored by Parker Hannifin and the problem given was:

*Design and build a system to automatically splice seals.*

For this project four teams of four students were given a presentation explaining the problem and sponsor requirements. Each team worked independently to further define and elicit requirements while developing solutions. The project is representative of a typical capstone design project at Clemson University, considering factors such as the complexity of the problem, the formation of the teams, the composition of the advisory committee, and the execution of the process.

As the goal of this study is to investigate the evolution of the requirements, the data collection entailed collecting weekly requirement documents from all four teams. In order to facilitate the data collection, a requirements update sheet was created as shown in Table 1.

This sheet captures details such as requirement, whether it was a constraint or criteria, the source of the requirement, justification for having the requirement, the

**Table 1** Snap shot of requirements update sheet

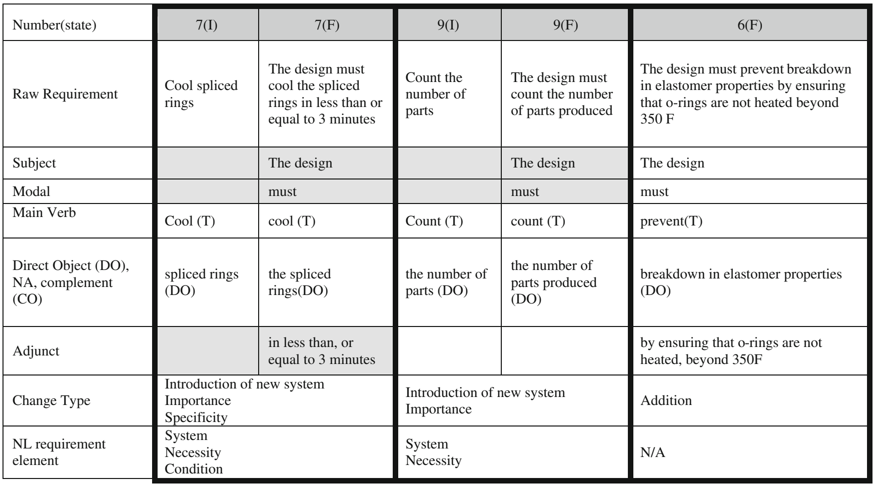
| | |
|---|---|
| Requirement number | 1 |
| Requirement | Grasp extruded circular cross-sections that range in thickness from 0.070″ to 0.250″ and diameters from 2″ to 2′ |
| Type | Constraint |
| Source | Sponsor |
| Justification for requirement | Requested by sponsor |
| Date of elicitation | 20-Jan |
| Target value | Described in requirement |
| Verification method | Calculations or actual results |
| Update | n/a |
| Reason for update | n/a |
| Description of change | n/a |
| Comments | |

target value, verification method, and any updates. All four teams were asked to update the requirements and submit it every week. This requirements documentation tool is similar to that which is taught in the preceding course and is common in design textbooks [9].

However, submitting the weekly documents was not a course requirement, thus most teams did not submit the requirements every week. The frequency of submission varied between the teams and, thus, the comparisons presented in this paper are made between initial weeks and final weeks. As with best practice in case study research, the context of study was intentionally not altered. It may be noted that initial week may refer to either week one, two or three. While the self-reporting of the requirements documents were not sufficient to ensure weekly change evolution, the study presented here is sufficient to provide preliminary evidence of the types of changes that occur within capstone projects. To this point, no other studies have been found to have explored this aspect of requirements in engineering design education.

## 2 Requirements Tracing Protocol

Three evolution aspects are investigated: (1) the number requirement additions, deletions, modifications, and no changes, (2) the changes in the requirement elements (system, necessity, behavior, object and condition), and (3) the types of change per the taxonomy of [10]. Protocols were established to study each of these change types. It may be noted that since this is a preliminary study investigating change types in requirements of novice designers, the goal here is to identify whether or not the requirements documents of novice designers change. Therefore, the robustness of these protocols is not tested for this preliminary study. In order to



| Number(state) | 7(I) | 7(F) | 9(I) | 9(F) | 6(F) |
|---|---|---|---|---|---|
| Raw Requirement | Cool spliced rings | The design must cool the spliced rings in less than or equal to 3 minutes | Count the number of parts | The design must count the number of parts produced | The design must prevent breakdown in elastomer properties by ensuring that o-rings are not heated beyond 350 F |
| Subject | | The design | | The design | The design |
| Modal | | must | | must | must |
| Main Verb | Cool (T) | cool (T) | Count (T) | count (T) | prevent(T) |
| Direct Object (DO), NA, complement (CO) | spliced rings (DO) | the spliced rings(DO) | the number of parts (DO) | the number of parts produced (DO) | breakdown in elastomer properties (DO) |
| Adjunct | | in less than, or equal to 3 minutes | | | by ensuring that o-rings are not heated, beyond 350F |
| Change Type | Introduction of new system<br>Importance<br>Specificity | | Introduction of new system<br>Importance | | Addition |
| NL requirement element | System<br>Necessity<br>Condition | | System<br>Necessity | | N/A |

**Fig. 1** Example requirements tracking sheet for team A

study this evolution, a requirement tracking sheet was created for each team and each requirement was traced as it evolved.

Figure 1 illustrates a snap-shot of requirements tracking sheet for Team A. The first row represents the requirement code and includes two elements, the requirement number and the state (I = initial weeks; F = final week). The mapping of the requirements from the initial document to the final document was done manually based on semantic content of the requirement. Some requirements were deleted and new requirements added. Thus the change type for these requirements was identified as 'addition' or 'deletion'. For example, in Fig. 1, there was no 6(I) requirement to correspond to the 6(F) requirement as this was only found in the final document. This requirement is designated as "addition" for change type. Similarly, if a requirement in the initial requirement document is not found in the final, then the change type is "deletion". If there was a change to the requirement, then the aspects of change are recorded. These change types are based on the work of [10].

The next step is to identify specific changes in requirement in terms of the elemental changes. The requirements are parsed into the elements: system (subject), necessity (modal), behavior/characteristic (main verb), and object and condition (complement or adjunct) [7, 11]. The elements that are different between the requirements are highlighted in Fig. 1. For instance, consider requirement 7 for Team A:

*7(I)*: *Cool the spliced ring.*
*7(F)*: *The design must cool the spliced ring in less than or equal to 3 min.*

In 7(F), the subject "design", modal "must", and adjuncts "in less than or equal to 3 min" are added to 7(I). These changes are highlighted (system, modality,

**Table 2** Description of change types, adapted from [10]

| Change type | Description (Identified as a change …) |
|---|---|
| Introduction of new system | In the system of the requirement. This could be addition of new component or system |
| Consistency | In vocabulary, units or terminology |
| Importance | In the necessity of requirement |
| Specificity | In the level of detail of a requirement |
| Application | In the application of part or system |
| Measurability/testing | In the measurability of a requirement |
| Withdrawal of system | In the system of the requirement. This could be a removal of system or component |
| Merging | When two or more requirements are merged into single requirement |
| Associated user | When the individuals associated with the requirement change |
| Splitting | When one compound requirement is split into two or more requirements |
| Scope change | Identified as change when the scope or focus of a requirement changes |

adjunct). Identifying the requirement elements is based on the protocol of [11]. After identifying the change in the requirement elements, the next step is to identify the change types based on the taxonomy of change types established in [10]. The definitions and examples of each change type provided in the taxonomy are used to identify the change type for each requirement pair. Table 2 provides explanation of change types identified by examining the requirements for this study based on the types proposed by [12–14].

A requirement pair can have multiple change types, as seen in the changes for requirement 7(I)–7(F) with changes in the introduction of a new system, the definition of an importance for the requirement, and a change in the specificity of the requirement (Fig. 1). The change types for all requirement pairs for all teams are identified in this manner and a summary table created by counting the number of each change type for each team.

## 3 Findings from Requirements Tracing

### 3.1 Number of Additions, Deletions, Change and No Change

The first evolution type investigated in this case study is the high level modification of the requirement document. This includes changes such as addition of, deletion of, change to a requirement (Fig. 2). It may be noted here that, this is a preliminary study to investigate the change types. While only four teams were investigated, the goal was to identify whether or not any change types can be identified in the requirements documents of novice designers. Once, these patterns are identified,

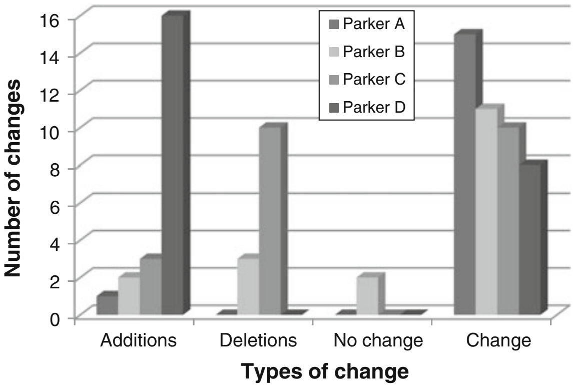


**Fig. 2** Types of change for project teams

extensive studies with multiple teams and different projects will be conducted to further support the findings. However, these four teams are representative of the other senior design teams in terms of the complexity of project, execution of the design process and final deliverables. So the findings can be extended and the recommendations are valid for other senior design teams.

From Fig. 2, it is clear that each team had different types of changes to the requirement document, with the most common change being a detail change to an individual requirement. Team D had eight requirements in initial week and 24 requirements in final week resulting to the most number of additions. Team C, on the other hand, had 20 requirements in initial week and 14 requirements in final week resulting to the most number of deletions. Of the four teams, only team B has requirements with “no change”. These findings suggest that requirement document of novice designers’ change significantly from initial weeks to final weeks. Further, these changes are not just limited to addition of new requirements as the understanding of the problem grows but also include deletions and modifications of existing requirements.

## *3.2 Change in Natural Language Requirement Element*

The second evolution type investigated in this case study is tracing the change in natural language requirement elements [10]. This accounts for changes in the system, necessity, behavior/characteristics, object or condition of a requirement. Figure 3 illustrates the overview of the changes in the natural language requirements elements.

From Fig. 3, it can be observed that all four teams had multiple changes in the natural language requirement elements. Most number of changes was observed in the ‘system’ element. The least number of changes were found in the ‘object’ element of the requirement sentences.

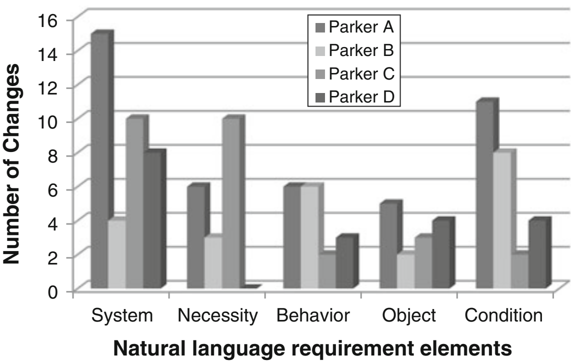


**Fig. 3** Changes in requirement elements

Syntactically, 'system' is represented by the subject within a sentence. Thus, the system changes also represent the change in the subject of the requirement sentence. There were several types of system changes observed across the four teams such as introduction of new system, splitting, with drawl of a system and consistency.

The element of 'necessity' within a requirement statement is syntactically represented by a modal and it shows the importance of the requirement [10]. Necessity element of requirement changed for all teams except team D. Investigating the requirements tracking sheet for all four teams, it was found that for most teams, the change in 'necessity' stemmed from addition of a modal such as 'must' or 'should' in the final week requirements document.

Next, the element of 'behavior' within a requirement represents the function of the system or component being designed and is syntactically represented by a verb [10]. Most changes in the 'behavior' stemmed from either change in the function of the system (change type—application—for example—must "remove" excess adhesive and must "minimize" excess adhesive) or inconsistency in the vocabulary used to describe the function of the system (change type-consistency—for example —must "provide" economic advantage and must "present" economic advantage). Apart from these two, the behavior changes also resulted from merging of two requirements, splitting of a requirement into multiple requirements and change in the scope or measurability of requirement.

The next element is 'object' and it represents what the system is affecting [10]. The changes in the 'object' of a requirement stemmed from change in the vocabulary within requirements. So for instance, one of the requirement for team D changed from "*the system must count completed spliced parts*" to "*The final design solution must count spliced O-rings*". Here the object changed from "spliced parts" to "spliced O-rings", which is essentially inconsistency in the vocabulary as the completed spliced parts are the O-rings.

After object, the next natural language element of a requirement is 'condition'. The changes in the 'condition' stemmed from change in the vocabulary or change in the specific details of the requirement. For example, condition for requirement-2 for team B changed from "ranging from 2 inch to 2 feet in ring diameter" in initial week to “ranging from 2 inch to 24 inch in ring diameter” in final week. Here the unit changed from 2 feet to 24 in. Other type of condition changes resulted from change in measurability, application, change in associated user, change of scope or splitting of a requirement.

Thus, this section describes the changes in the natural language elements of a requirement. Each of these elements is critical within a requirement and any changes must be documented with the justification for the change. Next, Sect. 3.3 describes the changes in the requirements document as per the change taxonomy.

### 3.3 Change Type as Per Change Taxonomy

The next evolution type investigated within the requirement documents was to trace the ‘type’ of changes as per the change taxonomy established in [10]. It may be noted that while the taxonomy captured all the change types occurring in student requirement document, not all the change types described in the taxonomy were found in the requirement documents. Example of some of these change types include replacement of system, updating, incorrect raw data interpretation and correcting among others. Figure 4 illustrates the summary of change types for all four teams.

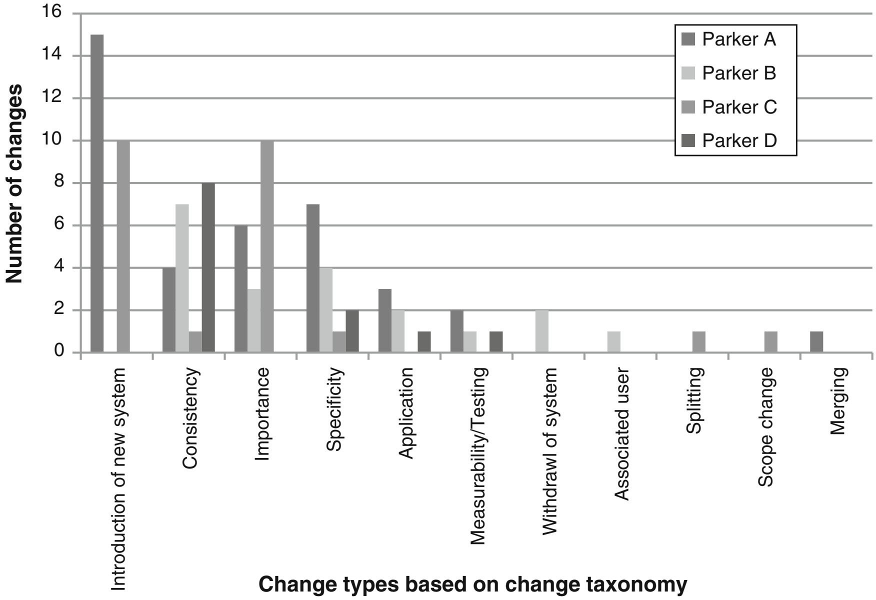


**Fig. 4** Change type occurrences

It can be observed from Fig. 4 that all teams had multiple types of changes in the requirements, with the introduction of a new system being the most frequent change type. This change was observed in teams A and C as a result of addition of the subject in the requirements for final week. Other change types that are more frequently observed across all four teams include consistency, importance and specificity. These changes stemmed from change in vocabulary or units, addition or deletion of a modal and change in the specific details of requirement respectively.

The change types that are less frequent include application, measurability/testing and withdrawal of system, while the change types associated user, splitting, scope change and merging only occurred once. Further, the changes that are less frequent were mostly accompanied by other change types that are more frequent. So for instance, splitting and merging occurred once, but in each case they were accompanied by a new system.

Some of these changes can be more critical than others and if not properly document can lead to information loss. For instance the change types such as splitting or merging involve either separating a compound requirement or uniting two requirements into one. This may not always lead to major change in information as it is essentially re-writing requirements in different form. For example the requirements "*(the system) Must handle organic and silicone adhesives*" and "*(the system must) apply adhesive to ends of extrusion*" were merged into one requirement "*The design must have the capability to apply either organic or silicone adhesive to an extrusion end*" which is essentially conveying the same information.

On the other hand the change types such as introduction of new system, importance, specificity and consistency which can alter the purpose of a requirement can prove to be more critical making the documentation of these changes more important. For example for team C, the requirement in initial document changed from "*Apply* ***controlled amount of*** *organic or silicone adhesive to extrusion ends*" to "*The system must* ***apply 1 drop per square inch*** *of organic or silicone adhesive to extrusion ends*" in the final document. Here the team added specific detail about the "controlled amount" and thus this change is specificity change. Adding information about how much adhesive enhanced the purpose of this requirement making this change more critical compared to just merging or splitting a requirement. It is interesting to note that the changes which are more critical are also more frequently observed in requirement documents of the teams.

## 4 Conclusion and Recommendations

It was observed that requirements documents have multiple additions, deletions, and changes from initial weeks to final week, though the novice designers rarely states the justifications for the changes. It is interesting to note that other documents such as weekly summary, presentations, mid-term and final reports that the students are required to submit as a part of project deliverable do not mandate providing updates on requirements. None of these documents require the students to provide

**Table 3** Recommendations for faculty

| | Recommendation for faculty | Benefit | Implementation |
|---|---|---|---|
| 1. | Mandate novice designers to maintain and submit weekly requirements update | This will help novice designers to manage and track the changes in the requirements document | Faculty may require the students to make weekly submission of requirements update sheet illustrated in Table 1 as a starting point |
| 2. | Educate novice designers to identify high level document changes such as additions, deletions and change and be able to justify them | This will help novice designers to prevent accidental additions or deletions of requirements | Faculty may use in protocol 1 discussed in this paper to educate students about high level requirement document changes |
| 3. | Educate novice designers to identify changes in requirement elements and be able to justify the changes | This will help prevention of potential information loss that could jeopardize the successful completion of the project | Faculty may use protocols 2 and 3 discussed in this paper to educate novice designers to identify changes in requirement elements and changes based on taxonomy |

justification for the changes in requirements document. If there is no justification for these changes, valuable information could be lost. Further, the novice designers are not able to trace if a requirement was accidently deleted or added if they do have the documented justification for additions and deletions of requirements. Next, investigating the evolution in the requirement elements, it is shown that each type of element can change throughout the project, yet again the students rarely documented the justification for the change. A survey of design textbooks was conducted to investigate the tools pertaining to requirements [9] and none of the textbooks describe tools for managing the requirement changes. With the lack of tools or methods to track requirement changes, the novice designer cannot track down if a change was accidently introduced in a requirement. Finally, the evolution in the requirements document in terms of changes as per change taxonomy was investigated. Again, if the students are not taught to identify these change types, the criticalities associated with the change types and appropriate tools or methods to document and track these changes; it can lead to potential loss of valuable information and jeopardize the successful completion of the projects.

Several recommendations are made based on the study presented above (Table 3). These recommendations may serve as guidelines for faculty helping novice designers manage changes in the requirements documents.

# References

1. Yin, R.: Case Study Research: Design and Methods. Sage, Thousand Oaks, CA (2003)
2. Tellis, W.: Application of a case study methodology. Qual. Rep. **3**(3), 1–17 (1997)
3. Teegavarapu, S., Summers, J.D., Mocko, G.M.: Case study method for design research: a justification. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, p. DTM–49980. ASME, Brooklyn (2008)
4. Joshi, S., Summers, J.D.: Investigating information loss in collaborative design: a case study with capstone design project. In: National Capstone Conference 2010, p. 51. Boulder, (2010)
5. Joshi, S., Morkos, B., Summers, J.D.: Requirements analysis: case study with capstone design project. In: National Capstone Conference 2012, p. 67. Champaign-Urbana, IL (2012)
6. Maier, J.R.A., Troy, T., Johnston, P.J., Bobba, V., Summers, J.D.: Case study research using senior design projects: an example application. J. Mech. Des. **132**(11), 111011 (2010)
7. Joshi, S., Summers, J.D., Morkos, B.W.: Requirements evolution: understanding the type of changes in requirement documents of novice designers. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, pp. DETC2014–35023. ASME, Buffalo (submitted) (2014)
8. Morkos, B., Summers, J.D.: Implementing design tools in capstone design projects: requirements elicitation through use of personas. In: National Capstone Conference 2010, p. 50. Boulder (2010)
9. Joshi, S., Morkos, B., Shankar, P., Summers, J.D.: Requirements in engineering design: what are we teaching? In: Horvath, I. (ed.) Tools and Methods for Competitive Engineering, p. 38. Karlsruhe (2012)
10. Morkos, B., Joshi, S., Summers, J. D.: Representation: formal development and computational recognition of localized requirement change types. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, pp. DETC2012–71417. Chicago (2012)
11. Lamar, C., Mocko, G.M.: Linguistic analysis of natural language engineering requirement statements. In: Horvath, I. (ed.) Tools and Methods for Competitive Engineering, pp. 97–111. Ancona (2010)
12. Rolland, C., Salinesi, C., Etien, A.: Eliciting gaps in requirements change. Requir. Eng. **9**(1), 1–15 (2004)
13. Stark, G.E., Oman, P., Skillicorn, A., Ameele, A.: An examination of the effects of requirements changes on software maintenance releases. J. Softw. Maint. **11**(5), 293–309 (1999)
14. Lin, L., Poore, J.H.: Pushing requirements changes through to changes in specifications. Front. Comput. Sci. China **2**(4), 331–343 (2008)