

Proposal: Separating Requirements from Goals and Design Decisions

RYAN DARRAS

Too often developers have a list of "requirements" dropped on them that isn't actually a list of requirements. Developers have to go through the process of eliciting the requirements themselves (whether through assumptions or pestering stakeholders/project managers) in order to be working on the project. This is due to elicited requirements often being goals or design decisions that don't dictate exactly what the project must do in order to be considered complete. By intentionally separating requirements from goals and design decisions, project managers and stakeholders will be able to more accurately create requirements that can be confidently passed to developers so that they can begin development.

ACM Reference Format:

Ryan Darras. 2018. Proposal: Separating Requirements from Goals and Design Decisions. 1, 1 (November 2018), 3 pages. <https://doi.org/10.1145/nnnnnnn.nnnnnnn>

1 INTRODUCTION

In the related literature, "requirements" has been defined in ways. "*requirements specifications state the desired functional and performance characteristics of some component independent of any actual realization*" [6] emphasizes that feature requirements don't actually exist yet. "*requirements are usually understood as stating what a system is supposed to do, as opposed to how it should do it*" [8] differentiates requirements from design decisions. "*a statement that identifies a capability or function that is needed by a system in order to satisfy its customer's needs*" [1] focuses on a single feature's functionality.

These definitions of "requirements" can often be considered synonymous with "goals" or "design", which can cause confusion when in the requirements elicitation processes. Author Paul Ralph describes the difference between requirements, "*a structural or behavioral property that a design object must possess*" [4] and goals/design decisions, "*goals describe the desired impacts of a design object on its environment*" [4] which we will use to dictate the structure of our proposed solution in order to obtain effective requirements.

Our tested environments will assume an overarching *project* for which stakeholders and project managers will develop *goals*. Similarly, for each goal the project managers will elicit a collection of *features* from the stakeholders. *Features* will be broken down into two categories; *required features* are features that must be implemented for the project to be considered complete, and *design features* are features that determine the design aspects of the project such as the components of UI and UX. For example, assume the client hired a team of developers to create a space simulation (the project) that will allow researchers to study the effects of extraterrestrial objects on theoretical man-made shuttles (the goal). The client states that this application must use accurate physics and allow for the user to simulate extraterrestrial objects of various characteristics so that they can simulate a broad range environments in which to research (the required features). The client also states that, while being astrophysicists and astronomers, the users of this program are not incredibly tech savvy and the program must be easy to learn and operate (design features).

Author's address: Ryan Darras.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2018 Association for Computing Machinery.

XXXX-XXXX/2018/11-ART \$15.00

<https://doi.org/10.1145/nnnnnnn.nnnnnnn>

2 RELATED WORK

There is a ton of related work in the area of requirements elicitation; documented below are a few that focus on requirements vs goals vs design decisions.

Gonzalo Génova and his team focus on quantifying requirements so that they can be measured in a fully automatic way in order to improve the quality of requirements as well as improve the writing skills for the requirements engineers [3]. They claim that trying to obtain all of these measurements by human hand would be impossible to do accurately, which is why they suggest using automated tools to collect this data. However, they mention that forcing these metrics on the requirements engineers isn't the best option as it would feel like a policing tool to prosecute or penalize the writers. To bypass this problem, the authors suggest a gradual implementation of automated tools to measure the quality of requirements such that the requirements engineers see the potential benefit of these tools as opposed to the oppressive penalization they might see.

Author Axel van Lamsweerde takes the approach of requirements elicitation with emphasis on goal-oriented requirements [7]. As the first sentence in his abstract he defines goals as, "Goals capture, at different levels of abstraction, the various objectives the system under consideration should achieve." Throughout the paper, the author describes how to create and understand goals that are effective in the requirements engineering process. The author summarizes the benefits to goal-oriented requirements engineering as follows:

- object models and requirements can be derived systematically from goals;
- goals provide the rationale for requirements;
- a goal graph provides vertical traceability from high-level strategic concerns to low-level technical details; it allows evolving versions of the system under consideration to be integrated as alternatives into one single framework;
- goal AND/OR graphs provide the right abstraction level at which decision makers can be involved for important decisions;
- the goal refinement structure provides a comprehensible structure for the requirements document;
- alternative goal refinements and agent assignments allow alternative system proposals to be explored;
- goal formalization allows refinements to be proved correct and complete.

A. Terry Bahill researches methods to elicit requirements in effective ways [1]. This author explicitly compares requirements vs goals by stating that, "The Mission Statement, Concept of Operations, and Business Model contain goals, objectives, features, and capabilities. Formal requirements are contained in the Specific Requirements Sections of the Use Cases, Supplementary Requirements Specification, and Tradition Requirements Specification." Although requirements and goals are critical parts to a successful project, they must be treated as separate entities to get the job done effectively.

Author Paul Ralph focuses directly on preventing the creation of statements labeled requirements, when these statements are in fact goals or design decisions [5]. The author considers the idea that a developer or team member can't be effective when their requirements are stated as goals. Instead, the developers should be given explicit requirements that don't elicit additional questions. Similarly, in another paper the author proposes a formal definition for the design concept which incorporates seven elements: agent, object, environment, goals, primitives, requirements and constraints [4]. By linking these concepts together in an appropriate way, the authors proposed method should allow stakeholders and developers to classify design knowledge and classify design approaches.

Pierre Bourque authored a book that discusses, in detail, software engineering which encapsulates the processes of requirements elicitation [2]. The author defines what a software requirement is and gives examples of software requirements. This book is a collection of useful information that can be used to review definitions and opinions of various software engineering areas.

Eric Yu focuses on the early stages of requirements elicitation as opposed to existing requirements techniques that are intended for the later phase of requirements engineering, "which focuses on completeness, consistency, and automated verification of requirements." Instead, the author sights in on the organizational context and rationals (the "Whys") that lead to the formulation of system requirements. By practicing and perfecting the early stage of requirements engineering, you can insure greater success in the overall outcome of the requirements.

3 BACKGROUND WORK

Bourque discusses many different types of requirements elicitation techniques and breaks them down into core principals [2]. These principles consist of Interviews, Scenarios, Prototypes, Facilitated meetings, Observation, and User stories but he also mentions that many other techniques exist. This paper will consider these six core principles to build a framework similar to that of Ralph's formal definition for the design concept [4]. By building a graph of Project -> Goals -> Features -> Required Features and Design Features -> Dependencies (other features) we can create a framework to utilize the six core principles to populate this map in such a way that allows for automated verification of requirements and insure a fully complete list of requirements that will take the project from start to finish.

REFERENCES

- [1] A. Terry Bahill, *Discovering system requirements*, In: Sage AP, Rouse WB (eds) Handbook of Systems engineering and Management. 2nd edn. (2009), 205–266.
- [2] Pierre Bourque, *Guide to the software engineering body of knowledge (swebok)*, IEEE Computer Society Press (2004).
- [3] Gonzalo Génova, José M. Fuentes, Juan Llorens, Omar Hurtado, and Valentín Moreno, *A framework to measure and improve the quality of textual requirements*, Requirements Engineering **18** (2013), no. 1, 25–41.
- [4] Paul Ralph, *A proposal for a formal definition of the design concept*, In: Lyytinen K, Loucopoulos P, Mylopoulos J, Robinson W (eds) Design Requirements Engineering: A Ten-Year Perspective. Lecture Notes on Business Information processing **14** (2009), 103–136.
- [5] Paul Ralph, *The illusion of requirements in software development*, Requirements Engineering **18** (2013), no. 3, 293–296.
- [6] Gruia-Catalin Roman, *A taxonomy of current issues in requirements engineering*, Computer **18** (1985), 14–23.
- [7] Axel van Lamsweerde, *Goal-oriented requirements engineering: A guided tour*, In: Proceedings of the Fifth IEEE International Symposium on Requirements Engineering (2001), 249–262.
- [8] Eric Yu, *Towards modelling and reasoning support for early-phase requirements engineering*, In: Proceedings of the Third IEEE International Symposium on Requirements Engineering (1997), 226–235.