A demonstration of Respecify: a requirements authoring tool harnessing CNL

Michael Ledger
*mike.ledger@transportengineering.com.au
Transport Engineering Pty Ltd

Abstract—Respecify is a new web-based requirements authoring tool using constrained natural language (CNL) to guide the authoring process and drive the creation of alternative views that elucidate the complexity of specifications, thus reducing the occurrence of certain types of requirements quality defects. This tool demo motivates Respecify’s existence, its key features, and evaluates its current use in industry.

I. INTRODUCTION

Requirements engineering is difficult. Intended needs are often neither fully nor accurately captured in requirements specifications. Complexity begets verbosity, and verbosity begets unwieldiness; the better a specification becomes in accuracy, the worse it becomes in understanding. Respecify is a software tool developed in order to ease the requirements authoring process, while also delivering specifications with fewer defects[1].

We partially achieve this through use of a constrained natural language (CNL) powered word processor, definitions glossary, and alternative views of relationships. Use of CNL ensures a basic level of quality throughout a specification[2][3], and may reduce the likelihood of requirements defects from occurring[1].

II. MOTIVATION

Authoring requirements in a word processor in plain English prose has real merits: namely it’s easy and intuitive, though quality is entirely up to the requirements engineer, and analysis is difficult. Tools like IBM’s DOORS sit at the other end of the spectrum: infamously complex, full of specific and powerful sub-tools, but highly arcane.

III. RESPECIFY DESCRIPTION

Respecify aims to be as intuitive and easy to pick up as a word processor, while still providing value through its use of a CNL.

A. The requirements editor

The most important feature of Respecify is its requirements editor. This is simple word processor that guides the user in conforming to the CNL as they type. If a requirement fails to parse the CNL, the tool uses the partial parse result to prompt the user with the possible syntactic tokens that would cause the parser to continue parsing.

For instance, upon typing “The”, the user is prompted with completions that are valid following a determiner (e.g., nouns). The user can then either select from the list or simply continue typing, which has the effect refining the completions by choosing completions that are valid suffixes of what the user typed.

For instance, Figure 1 shows an auto-completion box within a Respecify document that forms the Transport Engineering employee travel policy. "Business Administrator" is not a defined term, but it is similar enough to “Business Administration”, “business card” and “business cards” which are defined nouns within the specification) that it is suggested.

B. Definitions glossary

The CNL is extensible by a user-defined, per-document definition glossary. This glossary defines the nouns, verbs, reference documents, and special states/modes available within

3Respecify currently has no special handling of plurals – this is future work
the requirements editor\(^2\). On the panel on the right of Figure 1, since *Business Administrator* was not defined in the definitions glossary for the “Travel v1” document, it is offered as a definition candidate. A mode, or a reference, or a noun are valid at this point in the grammar, so these are the suggestions given.

In 1, the user has selected “Business Administrator” as a noun, and is given a list of suggestions for definitions (which, since the tool could not find a definition for “Business Administrator”, are instead for “Business”), or the option to add a custom definition for “Business Administrator”.

### C. Relationships view

The design of the CNL is such that relationships can be generated from requirements that conform to it, without any further user intervention. Relationships in Respecify roughly mean connections between entities defined within requirements. Relationships that we have defined so far include component hierarchies, states and modes, and interface relationships. This is an area of very active development.

![Fig. 3. Relationships generated by Respecify given a short specification for children’s cubby houses.](image)

Relationship types have associated algorithms for diagramming. We use a bespoke diagram notation similar to URML\(^4\) or data-flow diagrams. Component hierarchies can be visualised with UML dependency arrows, or attributes. Figure 3 shows a simple tabulation of component relationships. Figure 4 is an example relationships diagram (verbatim from our RE@Next submission\([1]\)) demonstrating how diagrams elucidate possible requirements defects.

### IV. Future Work

Our own use of Respecify in industry guides development. Current goals include: to validate the hypotheses\([1]\) for RE’17; to improve relationships graphing and extraction; to integrate change tracking à la Google Docs; to use literal generation from relationships for consistency checking, or to otherwise plug it into a SAT solver; to make publicly usable Respecify; and explore options for sharing definitions, and requirements between specifications (similar to Requirements Bazaar\([5]\)).

\(^2\)While reference documents and states/modes are usually nouns, the distinction allows for more granular relationships generation.

### V. Evaluation and Conclusion

*Transport Engineering* has so far respecified thousands of individual requirements, and around 20 complex specifications. This process of using Respecify – even in its infancy – has driven the direction of development so far, and lead to the development of a robust and flexible requirements CNL within a highly-performant, stable and user friendly tool. The CNL itself is still in development in order to better support relationship extraction (anecdotally, we have not struggled to express ourselves within the CNL so far); so the details are omitted here.

### REFERENCES


