

Formal Representation of Product Design Specifications

Students: Alexander Weissman and Kenan Özdemir

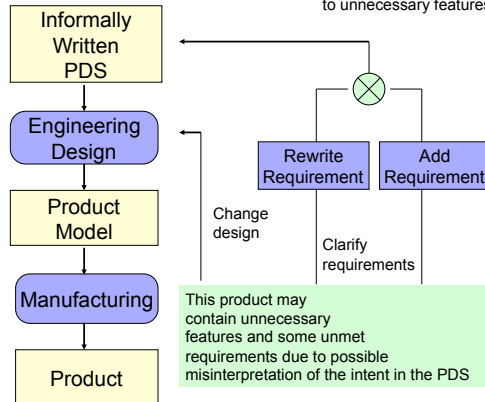
Advisor: S.K. Gupta; Collaborators: Xenia Fiorentini, Rachuri Sudarsan, and Ram Sriram Sponsor: NIST

Motivation

- Product Design Specification (PDS) is currently written as a Microsoft Word document.
- Common problems with this type of informal PDS include
 - Ambiguity as to what each term refers
 - Inconsistency among requirements
 - Redundancy of requirements
- This leads to:
 - Overdesign
 - Underdesign
 - Design bottlenecks



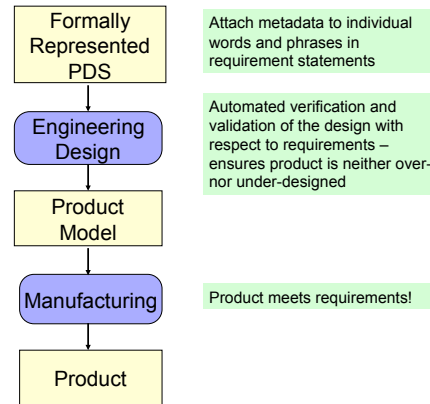
Informally written PDS can lead to unnecessary features.



Current Practice

Goals

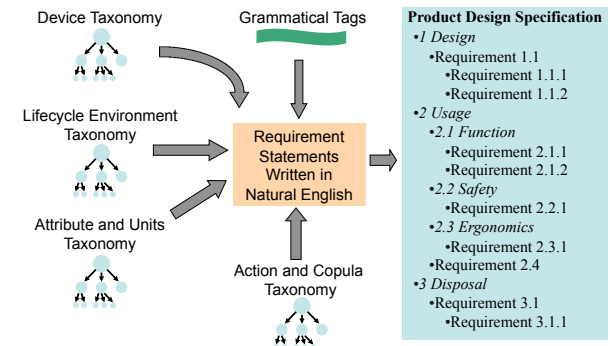
- Provide a formal representation for PDS that eliminates common sources of error
- Allow author to write statements *in their own words*, and then attach *metadata*
- Create a tool that provides automatic validation of design solution against formally represented requirements
 - Overdesign analysis (are there any unnecessary features?)
 - Coverage analysis (are all the requirements met?)
 - Bottleneck analysis (does one design feature affect many requirements?)



Practice Enabled by Proposed Work

Approach

- Capture most salient elements of the PDS
- Allow author to tag statements with terms from standardized, comprehensive taxonomies
 - Device and lifecycle environment taxonomy
 - Attribute (mass, viscosity, etc) and unit taxonomy
 - Action and copula taxonomy
- Create categories which allow author to classify statements according to role in the product lifecycle
- Allow author to tag grammatical units according to their semantic role
- Allow author to make *controlled* customizations and extensions to the taxonomies
- Preserve human readability of requirement statements



Representation Summary

- Taxonomies of terms
 - Action and Copula taxonomy
 - 33 primary verbs
 - 837 synonyms/hyponyms
 - Attribute Taxonomy
 - 12 primary categories
 - 196 unique attributes drawn from technical language across disciplines
 - Device Taxonomy – unique to each development group
 - Lifecycle Environment Taxonomy
 - People/objects classified in 16 unique lifecycle categories
 - Some people/objects may appear in multiple categories
- Requirement categories
 - Based on product lifecycle
 - Some requirement statements repeated in multiple categories
- Rich linking of requirements and design elements
 - Allow for linking groups of requirements or design elements

Analysis of Representation

- 5 Product Design Specifications containing a total of 365 requirement statements were analyzed. Each statement was classified based on how easily it could be encoded with our model.
- Statements that could be directly encoded (42%)**
 - Written as a requirement: "The product shall..."
 - All significant terms can be found in our taxonomies
 - Follows Subject-Verb-Object-Modifier structure
 - Statements that could be encoded with some rewriting (42%)**
 - All significant terms can be found in our taxonomies
 - Verbs may be missing from statements
 - Phrases might need to be reorganized
 - Statements might need to be decomposed
 - Statements that required significant rewriting (16%)**
 - Represents requirement data, but not written as such
 - Many verbs, nouns, and attributes missing from taxonomies or statements

Anticipated Benefits

- Automated validation against requirements will ensure good design of large, critical systems
- Creating formalized traces from design elements to requirements will help maintain justification of design decisions and avoid recurring errors
- Searching across Product Design Specifications using metadata from taxonomies will help designers to efficiently discover solutions based on past projects with similar requirements

