Summary of
"An Empirical Study on the Potential Usefulness of Domain Models for Completeness Checking of Requirements"

Introduction and Motivation

Completeness in software requirements is critical for avoiding costly oversights in system design and implementation. This paper addresses external completeness, which refers to ensuring all relevant information from external knowledge sources is included within the requirements. Unlike internal completeness – which checks for undefined references within the specification itself – external completeness involves cross-checking against external domain knowledge.

A common method proposed for ensuring external completeness is using domain models. These models, such as UML class diagrams, explicitly capture domain concepts and relationships. The underlying hypothesis is that every important domain concept or relation should appear in the requirements specification. Thus, the absence of a domain model element in the requirements might indicate missing information. This paper empirically investigates the practical usefulness of domain models for completeness checking in real-world scenarios.

Key Concepts

Completeness

- **Internal Completeness:** Specification contains no gaps or undefined internal references.
- **External Completeness:** Specification includes all necessary information relative to external sources.

Requirements ("Shall" Requirements)

- Functional requirements are structured in "shall-style" statements (e.g., "The system shall…").
- Each shall-statement specifies a single function or feature.

Domain Model

- Abstract, structured representations of domain knowledge using UML class diagrams.
- Models typically include domain concepts (classes), their attributes, and relationships (associations, generalizations).

Omission Types

- **Unspecified Requirements:** Entire missing functional requirements.
- **Under-specified Requirements:** Requirements missing specific details (conditions, constraints, objects).

Research Methodology

The study involved three industrial case studies:

1. Aerospace Simulator (163 requirements)
2. Sensor Platform (212 requirements)
3. Safety Assurance CMS (110 requirements)

Each case analyzed 35 "shall-style" functional requirements. Domain experts constructed feasibly complete UML class diagrams representing domain concepts derived from these requirements, adding any critical tacit concepts not explicitly stated in the documents.

Traceability and Simulation

Researchers established traceability mappings between domain model elements and requirement texts. Using Monte Carlo simulation, they systematically injected realistic omissions into the requirements (both entire requirements and specific segments like conditions and constraints) to measure the sensitivity of domain models—defined as the rate at which domain model elements lost coverage due to omissions.

Findings and Analysis

Sensitivity of Domain Models

The experiments showed domain models effectively highlight omissions:

- Domain models demonstrated near-linear sensitivity to omissions.
- Omitting entire requirements had a 4.4 times higher impact than omitting individual conditions or constraints. Removing whole requirements consistently left more domain elements uncovered, indicating higher sensitivity.
- Across all cases, domain models were approximately four times more sensitive to missing entire requirements than to missing details within existing requirements.

Variation Among Cases

Sensitivity varied based on the redundancy of domain concepts in requirements:

- Cases A (Aerospace) and B (Sensors) showed high sensitivity due to low redundancy of concepts. Removing even one requirement often uncovered domain concepts.
- Case C (Safety CMS) showed lower sensitivity, primarily because many concepts appeared multiple times across different requirements, providing redundancy that cushioned the impact of individual omissions.

Surrogate Metric for Domain Model Sensitivity

The study proposed a practical surrogate measure for estimating sensitivity based on the redundancy of keyphrases (nouns and verbs) within the requirements themselves. High redundancy of keyphrases correlates strongly with reduced domain model sensitivity to individual omissions. Thus, organizations can gauge the expected value of domain modeling for completeness checking by assessing keyphrase redundancy in their requirements documents.

Implications for Practice

- **Domain models as completeness checklists:** Domain models effectively identify omitted requirements or significant gaps. Elements of the domain model without corresponding requirements indicate potential omissions needing review.
- **Prioritizing completeness checking:** The strong sensitivity to completely missing requirements makes domain modeling particularly valuable for detecting overlooked functionalities.
- **Estimating value of domain modeling:** Organizations can assess redundancy in their requirements documents to estimate whether domain modeling will provide sufficient returns. Low redundancy indicates higher sensitivity and value in domain modeling.

Effort and Feasibility Considerations

Constructing domain models involves significant domain expertise and investment (approximately 6-8 hours for the small subsets examined). However, benefits extend beyond completeness checking to improved stakeholder communication, consistent terminology, and clearer understanding.

Limitations and Future Work

While domain models can clearly highlight missing requirements, this study did not evaluate whether analysts effectively identify and correct those omissions in practice. Future research should include user studies to explore the practical effectiveness of domain models in supporting requirements validation.

Conclusion

This empirical study provides concrete evidence supporting domain models' value as effective tools for completeness checking in natural-language requirements specifications. By systematically highlighting omissions, particularly entirely missing requirements, domain models can significantly improve requirements quality, making them valuable components of requirements engineering practice.