SP6: Verification of Software Requirements in Dynamic, Complex and Regulated Markets

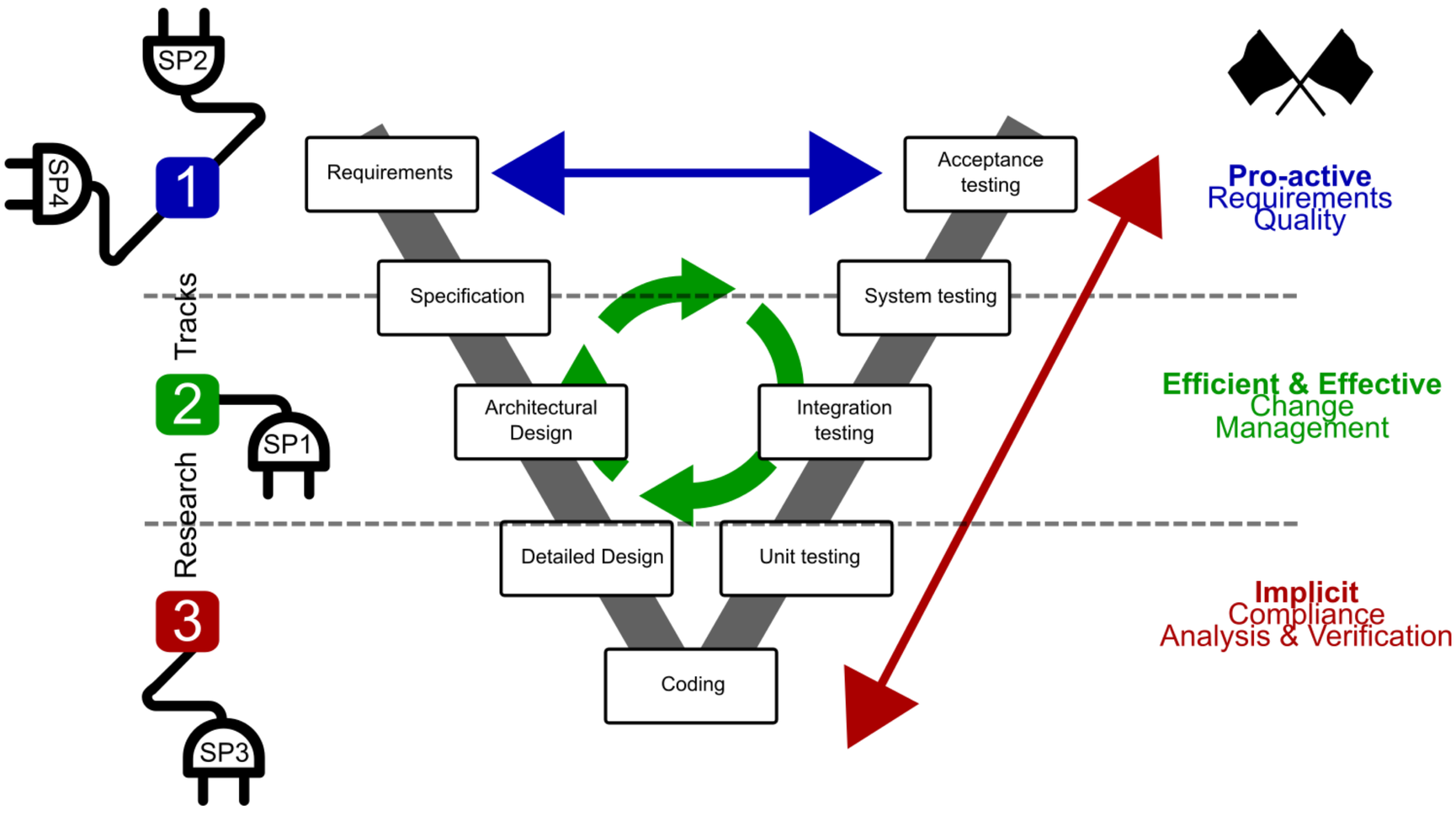
Did your software just break the law?

Evolving legislation and regulations are growing concerns in companies developing software intensive systems due to the costs associated with making their products compliant. Regulatory requirements as well as most requirements specifications in industry are written, to a large extent, in natural language (NL). While this facilitates communication among diverse stakeholders, NL is also inherently imprecise and ambiguous, which makes software compliance verification challenging.

In regulated markets, the ability to adapt to changes in a flexible manner is key to remain competitive. However, compliance analysis is still primarily performed with practices that lack scalability and little automated support exists to acquire, analyze and prioritize information required to achieve compliance with legislations and regulations.

Michael Unterkalmsteiner
(Sub-project Leader)
Assistant Professor
michael.unterkalmsteiner@bth.se

Three levers for effective compliance management



Research Team

Emil Alégroth
Assistant Professor
emil.alegroth@bth.se

Fabian Fagerholm
Associate Professor
fabian.fagerholm@bth.se

Previous experience

Femmer, Unterkalmsteiner, Gorschek (2017). Which requirements artifact quality defects are automatically detectable? A case study. In *4th International Workshop on Artificial Intelligence for Requirements (AIRE), pp. 400-406, Lisbon, Portugal*

Unterkalmsteiner, Gorschek (2017). Requirements quality assurance in industry: why, what and how?. In *23rd International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ), pp. 77-84, Essen, Germany.*

Alégroth, Karlsson, Radway (2018). Continuous Integration and Visual GUI Testing, In *11th International Conference on Software Testing, Verification and Validation (ICST 2018)*

Computational meets human intelligence

- **WHAT:** Let computers do what they do best: process large amounts of data, filter and reduce. Let humans do what they do best: weigh information, abstract, decide.
- **HOW:** Support legislators and engineers to focus on difficult problems, increasing their effectiveness. Identify tasks that are cognitively demanding, but replaceable with computational intelligence.
- **STAKEHOLDERS:** Legislators, requirements engineers, test engineers, project management.

Planned Outcomes

- Processes and support systems that align requirements quality standards and that reduce faults early in the development cycle: **Pro-active requirements quality**.
- Support systems that point to potential inconsistencies and omissions in work products, caused by regulatory changes : **Efficient & Effective Change Management**.
- Reuse and repurpose of low-level compliance checks (tests) and automated collection of evidence for high-level compliance to regulations: **Implicit Compliance Analysis & Verification**.

SERL Sweden
LEADING SOFTWARE ENGINEERING
SOFTWARE ENGINEERING RETHOUGHT
Knowledge Foundation

Data-focused Software Engineering

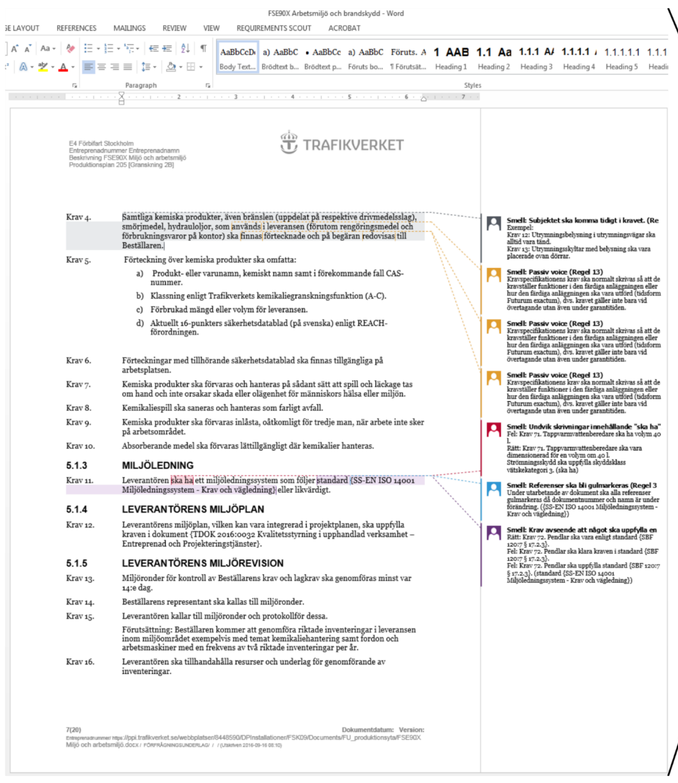
Smarter allocation of cognitive effort

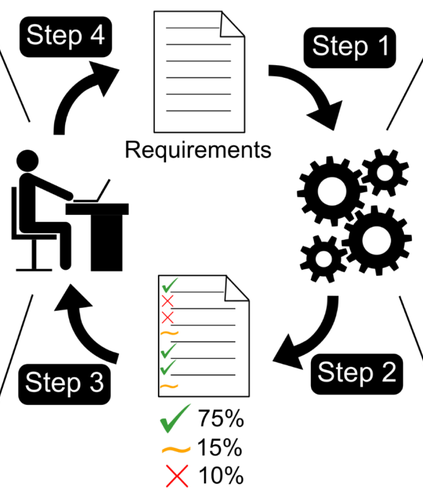
Many tasks in Software Engineering require that the engineer analyses large amounts of information. CASE tools support engineers in analysis tasks were information is well structured, such as source code or design models. However, little support exists for software engineers that need to work with natural language text, e.g. when expressing requirements or modeling domain knowledge. It is often challenging to precisely describe the task for which support would be needed, hence general purpose tools or processes are used while dedicated support would be more efficient and effective.

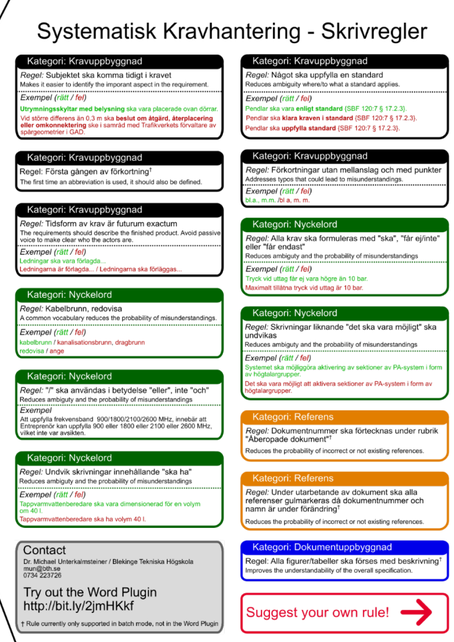
The examples illustrate very specific tasks that were extracted from a larger engineering context and analyzed in order to design support systems that help engineers. These particular tasks are related to requirements engineering.

Michael Unterkalmsteiner
Assistant Professor
michael.unterkalmsteiner@bth.se

Example 1: Real-time requirements quality assessment

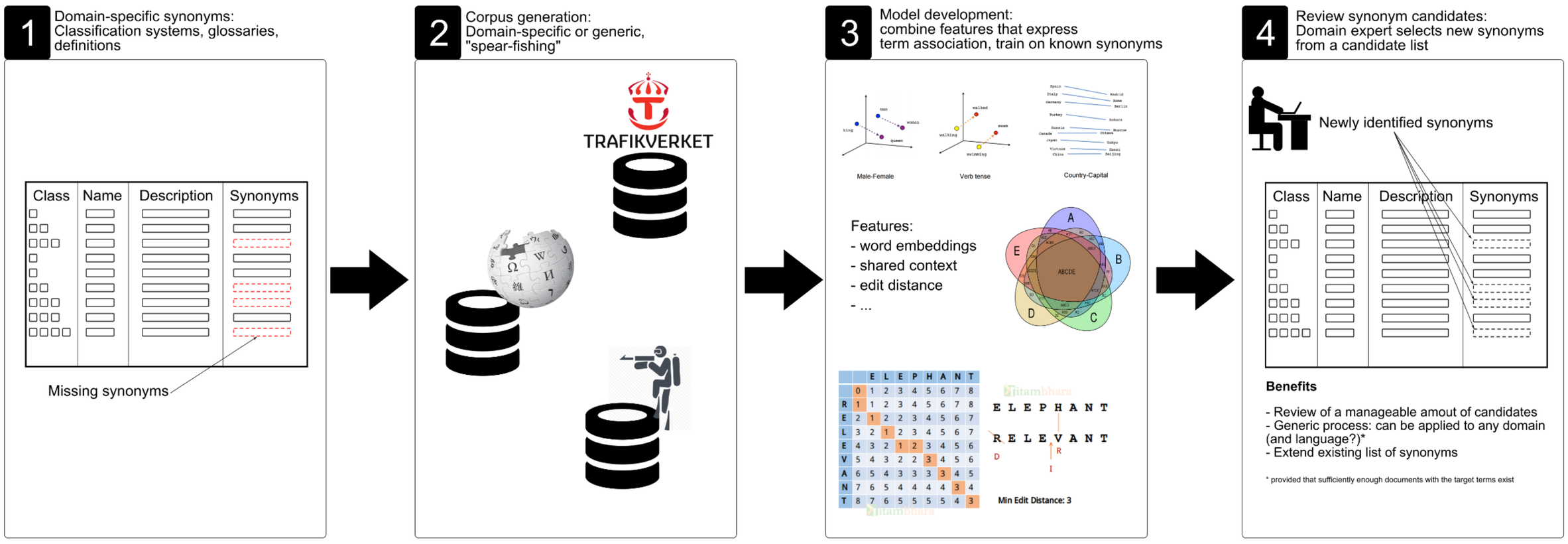






- Fosters a common understanding of requirements quality among engineers
- Allows to focus on “difficult” quality issues
- Allows to estimate overall quality of requirements specifications (over time)
- Pedagogical conveying of requirements quality aspects
- “Spellcheck”-type, immediate feedback
- Integrated into existing infrastructure
- Future work: feedback loops, learning of rules

Example 2: Identification of domain-specific synonyms to improve communication



SERL Sweden
LEADING SOFTWARE ENGINEERING
SOFTWARE ENGINEERING RETHOUGHT
Knowledge Foundation