Formalization of a Software Systems Research Methodology that Uses Prototypes for Evaluation

Dr. Vladimir Tosic
NICTA, Australia
vladat (at) server: computer.org

Presented
2011/06/02,
Uni. of Nis

NICTA Copyright 2010
From imagination to impact

Motivation

• Computing/software researchers often do not formally identify their research methodology
• They simply follow some “customary” research procedures (“as the others did”)
• Our universities ask Ph.D. students to formally identify their research methodologies for Ph.D. confirmation
• This presentation attempts to formalize 1 common “customary” research procedure
• Software is also researched in other ways

The Problem with Naming

• Since this “customary” research procedure is usually not formalized, it is also not named
• The closest name used in the literature is “(computer) systems research”
• However, there are very different opinions about what “systems research” is and is not
• I will use this name, but limit my formalization to software systems and use of prototypes as the main (not the only) evaluation mechanism
• Research: analysis–synthesis–evaluation loop

1. Analysis: Decide What to Research

• Based on literature review (including publications about characteristics of existing IT systems and new technologies) and realistic usage examples or insights into possible uses of new technologies,
• determine significant and realistic problems where the existing IT solutions are not sufficient or could be improved and
• decide for which metrics (e.g. speed) these solutions should be improved

2. Synthesis: Design Better Solutions

• Design novel models, formulas, processes, algorithms, data structures, software architectures, patterns, protocols, specification languages, methods, techniques, and/or other generic means or principles that
• provide better (in the sense of the chosen metrics) solutions than the past work and
• are general so that they can be applied (possibly with modifications by others) to a class of problems and not only 1 particular problem
3. Evaluation: Evaluate Suitability

- Using a prototype implementation and experiments with it as the main evaluation mechanism (and possibly other evaluation mechanisms),
- check and critically and objectively discuss feasibility, functional correctness, optimality (in the sense of the chosen metrics), and/or generality (e.g. assumptions validity and use limitations) of the proposed solutions [=> Analysis]

4. Evaluation: Check Side-Effects

- Using the prototype implementation and experiments with it (and possibly other evaluation mechanisms),
- examine whether there are side-effects of the proposed solutions, such as resource overhead, performance overhead, impact on scalability, and/or other aspects and
- critically and objectively discuss all strengths and weaknesses of the proposed solutions [=> Analysis]

Research vs. Pure Development

- Amount of novelty and uncertainty that has to be dealt with
- Amount of generality and identified principles
- The extent and quality of critical and objective discussion of both strengths and weaknesses of the proposed solutions
- Answering not only ‘What?’ and ‘How?’, but also ‘Why?’ questions (less likely also some of ‘Where?’, ‘When?’ and ‘Who / Which software modules?’ questions)

Concluding Remarks

- It is important to formally identify/define the used research methodology
- However, this is often not done in research literature on computer/software systems
- I tried to formalize 1 (of several) commonly used “customary” research procedures
- It uses software prototypes for evaluation (which is the critical part of research)
- This presentation is an input into the discussion, not the final conclusion