SoSE: National Graduate School on Software Systems and Engineering

- 7 fully funded positions for PhD students in SE
- Organizes shared courses, seminars, etc.
- Networks SE research community in Finland
- Non-funded PhD student positions, too: seminars, courses, travel funding, contacts etc.
- Fall seminar on Software architecture and agility
- Spring courses on SE research and paper writing
- Coordinated by TUT

Sulake project: Supporting software architecting in embedded control systems

Kai Koskimies
TUT

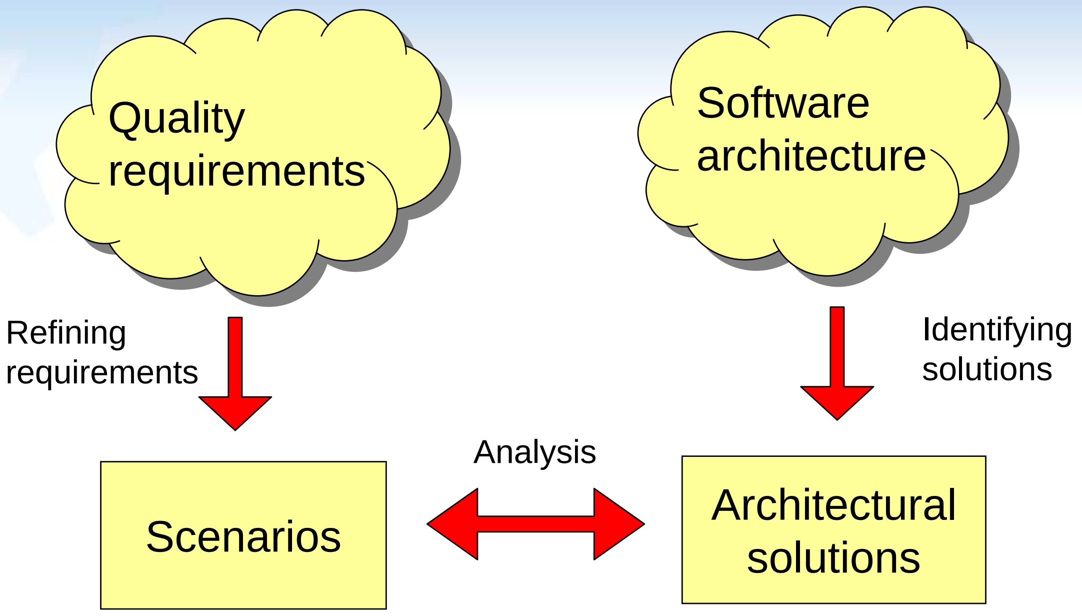
Aims of Sulake

- Study the applicability of existing architecture evaluation methods (ATAM) in embedded control systems (ECS)
- Develop architecture evaluation practices in ECS
- Identify and document successful architectural solutions in ECS (pattern mining)

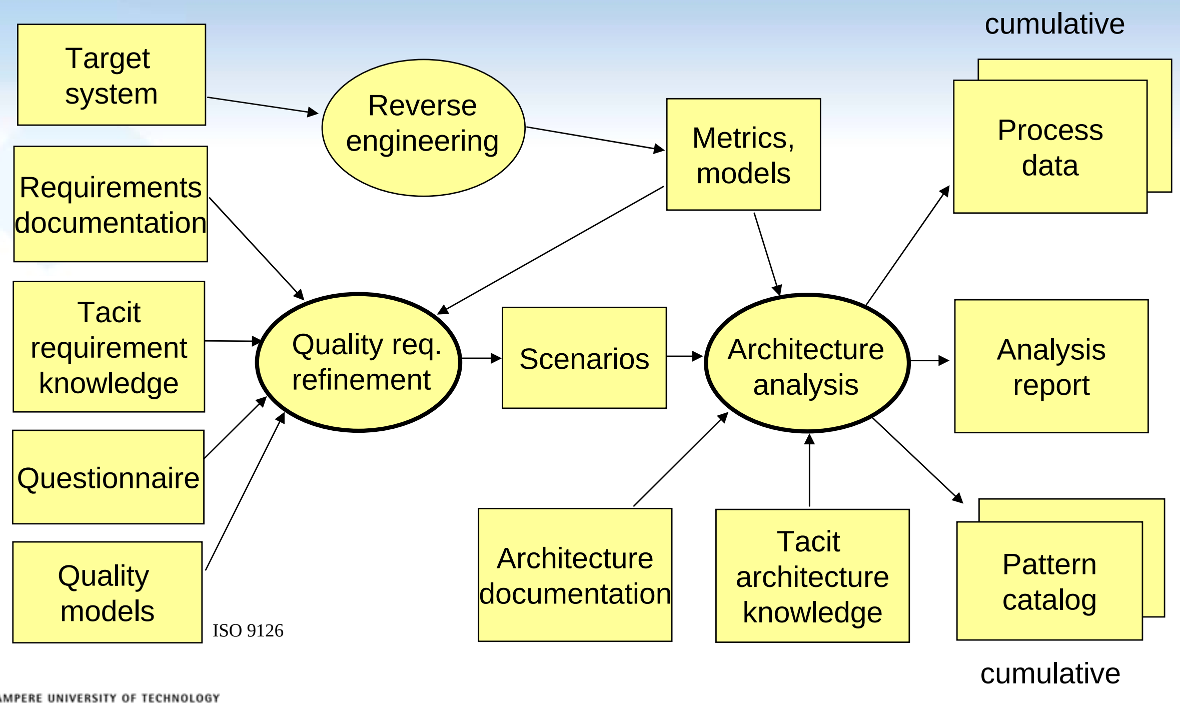
- Partners: Areva T&D, John Deere, Kone, Sandvik

- Tekes project Jan 2008 – July 2009

What is architectural evaluation?



Sulake data flow

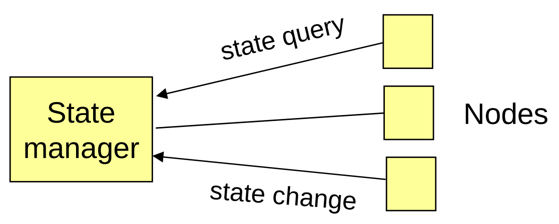


Pattern example: Common System State

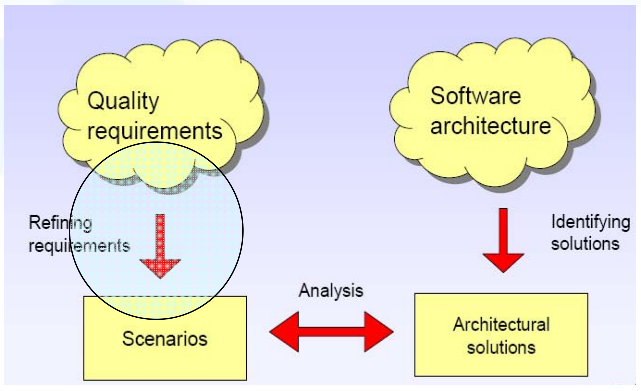
Context
An embedded control system, where several autonomous units in a system must share common state information about the system as a whole.

Problem
How can you efficiently share sufficiently accurate, consistent system state between different parts of the embedded system?

Solution
A common state manager module is implemented that contains the variables constituting the shared state. Nodes access the variables by static names. The values of the variables are updated using different strategies (by-request, periodically, as a side-effect). The values can have associated status or age.



Main problem in scenario-based architectural evaluation



How to find a useful set of scenarios efficiently?

Analogy: finding test cases

But scenario-based analysis differs from normal testing:

- requires human creativeness
- no obvious coverage principles
- running scenarios is expensive

Observations on architectural evaluation

- Stakeholders are ”forced” to communicate about architectural solutions, quality requirements, future needs etc. => increases common understanding
- Provides information for improved architecture documentation
- Allows and encourages fresh views on the system
- Reveals potential risks in the architecture (20-50% of scenarios)
- Embedded control system context brings additional flavour: systems resemble each others, same or similar basic problems and solutions, large potential for synergy