SAFETY & RESILIENCE ISSUES IN AUTOMOTIVE SOFTWARE DEVELOPMENT

PANEL
✓ when 26262 will be issued, enforced? What about 61508?

✓ by whom, which authorities?

✓ who and how will verify conformance? (statement of conformance)

✓ how costs and schedule are impacted (ASIL A vs. ASIL D) (es. 1X10)

✓ how to handle pre-developed software, validated tools

✓ software partitioning of mixed critical software


✓ And security?

✓ Relationship with AUTOSAR (Safety WP)
Open Discussion

Experience and drawbacks in standards application

“Agility” for Small Projects
1968 : the software crisis is declared in Garmisch: we cannot continue to develop software this way!

Software Engineering is born...

Waterfall, V model, iterative, evolutive, spiral, etc.

DOD 2167A, DO 178B, PSS-05, CENELEC, etc.

ISO 90003, TickIT, ISO 12207, SPICE, CMM, CMMI

Late 1990s: a revolt against “heavy processes” with the Agile Challenge: Extreme Programming, Scrum, Crystal, Lean, ...
We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

**Individuals and interactions** over processes and tools

**Working software** over comprehensive documentation

**Customer collaboration** over contract negotiation

**Responding to change** over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler

James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick

Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

www.agilemanifesto.org
Some principles of XP

- **Pair programming.** All production code is written by two people at one screen/keyboard/mouse.

- **Continuous integration.** New code is integrated with the current system after no more than a few hours. When integrating, the system is built from scratch and all tests must pass or the changes are discarded.

- **On-site customer.** A customer sits with the team full-time.

- **Collective ownership.** Nobody “owns his own code.” Anybody can change anybody else’s code.

- **Planning game.** After each iteration, the requirements and scope of the project may be changed.
From waterfall to daily/weekly delivery
The new buzzwords

- Planning game
- Product burndown
- Sprints
- Scrum master
- Team velocity
- Standup meeting
- Project wall-board
- Architecture slices
- Celebration

- Toolsmith
- Story points
- Wireframe
- Backlog
- Product roadmap
- ...

Toolsmith
Lessons Learned From Agile

- Efficiency involves minimizing:
  - Minimize unnecessary scope / requirements
  - Minimize unnecessary documentation
  - Minimize unnecessary coding

- Iterative development appears to be central

- Although these are well-known best practices in *theory*, the agile movement is helping discover what is necessary to make them operational in *practice*
SERVE UN PO' DI AGILITY NEI NOSTRI PROCESSI?
Agile versus CMMI-like processes

- The big question: are they incompatible?
- Many are working hard to make them compatible (www.agilecmmi.com)
- The SEI and others (e.g. Boehm) have a strong interest in reconciling them
- Can a “CMMI team” be agile? The SEI Team Software Process tries to be
<table>
<thead>
<tr>
<th>Agile Manifesto</th>
<th>How the SEI Team Software Process (TSP) Relates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individuals and interactions over processes and tools</td>
<td>TSP holds that the individual is key to product quality and effective member interactions are necessary to the team's success.</td>
</tr>
<tr>
<td></td>
<td>Project launches strive to create gelled teams. Weekly meetings and communication are essential to sustain them. Teams define their own processes in the launch.</td>
</tr>
<tr>
<td>Working software over comprehensive documentation</td>
<td>TSP teams can choose evolutionary or iterative lifecycle models to deliver early functionality—the focus is on high quality from the start.</td>
</tr>
<tr>
<td></td>
<td>TSP does not require heavy documentation.</td>
</tr>
<tr>
<td></td>
<td>Documentation should merely be sufficient to facilitate effective reviews and information sharing.</td>
</tr>
<tr>
<td>Customer collaboration over contract negotiation</td>
<td>Learning what the customer wants is a key focus of the launch.</td>
</tr>
<tr>
<td></td>
<td>Sustaining customer contact is one reason for having a customer interface manager on the team.</td>
</tr>
<tr>
<td></td>
<td>Focus on negotiation of a contract is more a factor of the organization than of whether TSP is used.</td>
</tr>
<tr>
<td>Responding to change over following a plan</td>
<td>TSP teams expect and plan for change by:</td>
</tr>
<tr>
<td></td>
<td>Adjusting the team's process through process improvement proposals and weekly meetings. Periodically relaunching and replanning whenever the plan is no longer a useful guide. Adding new tasks as they are discovered; removing tasks that are no longer needed. Dynamically rebalancing the team workload as required to finish faster. Actively identifying and managing risks.</td>
</tr>
</tbody>
</table>
Agile Versus Certification and Standards

- Where could the issues lie?
- Emergent architecture – architecture “emerges” from the continuous design process
- Continuous changing (refactoring) of code
- Scope of projects (such as requirements) can expand and contract over the life of an agile project
- Where to standards fit into this paradigm?