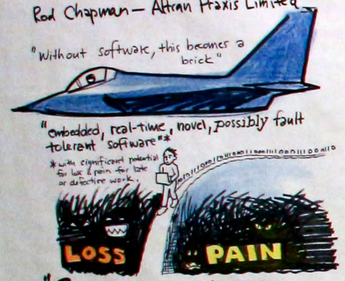
TSP Symposium 2012

Day Two

Delivering Agility & Discipline:

Experiences with High-Assurance Software Engineering

Rod Chapman — Altran Praxis Limited



"Correctness by Construction"

- Strong Process & "zero defect tolerance" culture
- Evidence-based Assurance
- Static Verification
- USE OF FORMAL METHODS
  - Maths
  - Tools that yield 100% for well-defined defect classes

Every Software Project uses Formal Methods,

the big question is not IF to use Formal Methods, but WHEN to start.

We must complement human brains with tools, b/c some problems are just too hard for our brains.

$$\pounds = \beta_0 + \beta_1 * \boxed{\text{XX}_1}\boxed{\text{XX}_2}$$

Discipline eventually becomes habitual

Strong Static Verification

Unpopular because of:

- False Positives
- False Negatives
- Slow

"PSP by stealth"

We need Sound tools that give us trustworthy results

- unambiguous notations early in the lifecycle to analyse

Deploying Static Verification in PSP

If a tool is sound for a defect class & so fast that defect removal is hardly worth noting — then deploy that tool as early & often as possible, & adjust later process steps to ignore the defect class completely.

Integrating Model-Driven Engineering Techniques in the Personal Software Process

João Faria — University of Porto

PSP & UML

- UML is enriched with documentation notes & model element properties
- construct specs in OCL
- Algorithms descriptions in a UML-compliant language
- UML diagrams are a convenient means to record data that TSP templates deal with.

UML for documentation only

- is time consuming
- is often wrong
- produces results that quickly become outdated & not maintained

Model-Driven Engineering

aims to avoid those problems by generating code from models

Our approach is a form of lightweight MDE

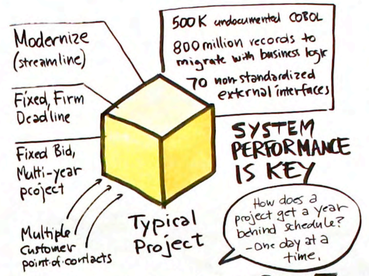
- develop structural models
- develop partial behavioral models
- generate executable tests from parameterized sequence diagrams

Excellence:

Methodolgy Assists, Discipline delivers

David Ratnaraj — AIS

Excellence: making success repeatable



> START THE PROJECT RIGHT

- Teams are fully committed to all aspects of the project and have the skills to execute & manage it.

> PLAN IN DETAIL

- Use historic data, plan the unknowns, keep the plan current, tailor processes

> EXECUTE WITH DISCIPLINE

granular Plan — team, individual; component
post-mortem & feedback — team & individual
with quality above all.

Team Morale + Employee confidence and growth was the MOST VALUABLE ASSET

> CUSTOMER COMMUNICATION

- Periodic status updates
- Frequent issue resolutions, technical interchanges, & product demonstration
- Frequent deliveries for acceptance
- NO SURPRISES; GET BUY-IN

Sucess is understanding and adapting from experiences

Results

- 680 total KLOC developed over 3 years
- Met required Federal Standards
- Zero system downtime due to functionality
- 0.097 defects/KLOC
- Final product 1 month ahead of schedule
- End user productivity increased 300%
- Saved customer $2 million
- Exceeded customer needs for Quality, Timeliness, & Value

Analysis of Code (and design) Defect Injection & Removal in PSP

Diego Vallespir — University of Uruguay

William Nichols — SEI

The Cost of removing code defects in Unit Test are **7 TIMES** higher than the ones removed in Code Review

To prevent or remove defects efficiently, we have to understand them.

Where and when are they injected and removed? Which type is most frequent? Which type is most expensive to remove? How many and which types hide until testing?

Syntax, Assignment and Function defects account for 80% of code defects

62% of defects are found early in the CR phase. **21%** of defects escape detection prior to UT.

The estimated yield of the PSP (during the course) is 81%