Instrumentation-Based Verification for Medical-Device Software

Rance Cleaveland
Professor of Computer Science, University of Maryland
and
Executive & Scientific Director, Fraunhofer USA Center for Experimental Software Engineering (CESE)
Fraunhofer CESE

- Applied-research institute in software engineering
- Founded 1998; in University of Maryland (UMD) technology park
- Staff: 30 – 16 technical (10 PhDs), 12 students / visitors
- Annual budget: US $4.5m
- Part of Fraunhofer USA / Gesellschaft; affiliated with UMD
Fraunhofer?

- German-based non-profit network of applied-research institutes
  - University-affiliated
  - All areas of applied science and engineering
  - 2011 revenues: $2bn
  - Institutes in Austria, US, other countries

- Inventor of MP3

Joseph von Fraunhofer (1787-1826)
The Fraunhofer Model

- Institutes act as conduit between universities, industry and government
  - Institutes affiliated with universities
    - Professors
    - Students
    - Professional staff
  - Single institutes focus on one discipline
- Funding: 30% base, 70% projects
CESE Overview

• Mission

Better software-development technologies, practices and processes

• Technical expertise

Software design, verification and validation, project management

• Target sectors

Aerospace / defense, automotive, medical

• Biggest customer

NASA
Types of CESE Projects

- Basic research
  \textit{Multi-year, externally funded, often collaborative}
- Technology development and evaluation
  \textit{Multi-year, internally and externally funded}
- Technical consulting
  \textit{Varying durations, externally funded}
- Software development
  \textit{Varying, externally funded}
- Training
This Talk

• Applying model-based V&V to medical-device software

• Work is part of a Model-Based Design (MBD) of Infusion Pumps effort at FDA OSEL
  – Investigate innovative methods for software development, safety, verification
  – Involves collaborations between FDA, universities, Fraunhofer (lead: Arnab Ray)
  – Showcase: Generic Infusion Pump (GIP) project
Talk Agenda

- Model-Based Development and the FDA
- The GIP project
- Model-based testing and Instrumentation-Based Verification (IBV)
- Using IBV on GIP: from safety to V&V
- Conclusions
Model-Based Development for Infusion Pumps

From www.fda.gov/MedicalDevices/ProductsandMedicalProcedures/GeneralHospitalDevicesandSupplies/InfusionPumps/ucm202511.htm#mdlBasedDesign:

“The FDA has recognized that if product developers had tools that enable them to examine and evaluate software earlier in the development cycle, then there would be a greater likelihood that the resulting software would be more robust. …[T]he software engineering community has been developing tools for modeling software and its interactions with the system it controls. The safety properties of the model can be systematically examined, and once the model has been verified, the software derived from it can be proven to conform to the model. The result is software designs that are far more robust than those developed using traditional methods.

Over a period of years, the Software Engineering Laboratory\(^2\) at FDA (within the Center for Devices and Radiological Health) has been working … to develop and refine model-based engineering methods and associated verification techniques. Characteristically, these methods have been applied first in the aerospace and automotive industries, where the cost of failure is enormous in both social and economic terms and the incentive to make the necessary investment is correspondingly high.
Additional Motivations

• Efficiencies in production of medical-device software
  – Faster development
  – Faster, more thorough V&V

• Efficiencies in regulatory approval activities
Software: An Automotive Perspective

- **Driver of innovation**
  
  *90% of new feature content based on software [GM]*

- **Rising cost**
  
  *20% of vehicle cost [Conti], 50% for hybrids [Toyota]*

- **Warranty, liability, quality**
  
  *High-profile recalls in Germany, Japan, US*
A Grand Challenge

• Ensure high quality of automotive software
  – ... preserving time to market
  – ... at reasonable cost
• Key approach: Model-Based Development (MBD)
Traditional Software Development

Requirements / specs / designs / test plans / etc.  

Source code
Model-Based Development

Use models (MATLAB® / Simulink®) as designs / specs

Requirements / test plans / etc.  Design / spec  Source code
Model-Based Development (cont.)

- Models support V&V, testing, inter-team communication
- Models can be simulated, managed electronically
What Is the “Auto-Motivation”? 

• What is motivation for model-based development?
  – Formal specifications? 
    \textit{No}
  – Clearer designs? 
    \textit{No}
  – Executable documentation? 
    \textit{No}

• Answer: \textbf{automatic code generation (aka autocoding)}
  – Coding part of existing processes
  – Coding is expensive, “non-core”
Intuition: A programming language is an executable notation for specifying system behavior.

- **Compilation**: Machine/Assembly language (load, stores, moves)
- **Auto-coding**: Procedural/OO languages ("high-level" constructs: guarded loops)
- **Executable modeling notations**: captures “high-level” design information, portable and re-usable
So What Are Models?

Executable designs

Can be simulated like “conventional” code, used for code verification

More abstract than code

Free from implementation details, can be constructed faster than “conventional” code
Simulink®

- Block-diagram modeling language / simulator of The MathWorks, Inc.
- Hierarchical modeling
- Continuous-time and discrete-time simulation
- Used in MBD of control software
Stateflow®
MBD Is *De Rigueur* in Auto

• Why? Efficiency!
  – Efficiencies mandatory in software development
  – MBD+autocoding has produced concrete, measurable advances (6x improvements at Nissan, for example)

• Other uses
  – Requirements specs for suppliers, offshore partners
  – Model-based testing
Model-Based Testing

- Compare specs, design, implementation
- Compare specs, design with system
Two Roles for Models in MBT

- Oracle: what are the right answers?
- Data: what are the right tests?
Reactis®

A model-based V&V tool from Reactive Systems, Inc.

Tester
Generate tests from models (also C)

Simulator
Run, fine-tune tests

Validator
Validate models / C

Simulink / Stateflow / C

Model / code

Reactis / Reactis for C
Reactis systematically generates inputs to drive simulation runs to cover model, produce test suites.
MBD and Medical Devices

• Some of same trends as automotive industry
  – Explosive growth in importance of software
  – Software development is “non-core”
  – Need for efficiencies

• Difference: regulatory environment

• Hence: FDA research on MBD
The Generic Infusion Pump Project

From:
www.fda.gov/MedicalDevices/ProductsandMedicalProcedures/GeneralHospitalDevicesandSupplies/InfusionPumps/ucm202511.htm#gipProject

“[An] ongoing research project related to model-based software development is the Generic Infusion Pump Project. The goal of this project is to develop a set of infusion pump safety models and reference specifications that can be used/adapted by manufacturers to verify safety properties of their own infusion pumps.... Collaborators include researchers at the University of Pennsylvania in Philadelphia and the Fraunhofer Center for Experimental Software Engineering in College Park, MD.

“The Generic Infusion Pump project team has a website that provides a sample hazard analysis and reference safety specifications for a generic patient-controlled analgesic infusion pump. This kind of device, often known as a “pain pump,” is used to infuse medication to relieve chronic pain.”
The GIP Project

• Safety analysis
  – Hazards
  – Mitigations
• Software requirements based on mitigations
• Reference architecture / Simulink models
• Verification of requirements on models
Model Validation

Design-time modeling, requirements verification
Instrumentation-Based Verification

- Model-validation technique supported by Reactis
- Combines assertions in models, testing
Instrumentation-Based Verification: Requirements

- Automatic verification requires formalized requirements
- IBV: formalize requirements as monitor models
- Example
  “If speed is < 30, cruise control must remain inactive”
Instrumentation-Based Verification: Checking Requirements

- Instrument design model with monitors
- Use coverage testing to check for monitor violations
  - “Skeptical tester”!
- Reactis:
  - Separates instrumentation, design
  - Automates test generation
But ... Does IBV Work?

• Three-month case study with Tier-1 automotive supplier on production system

• Artifacts
  – 300-page requirements document
  – Some source code

• Goals
  – Formalize requirements as monitor models
  – Prepare candidate design models
  – Check requirements
  – Work must be done by intern!
“[This] is the complete description of the control of the CAN output signals can1 and can2 produced by Function A. Function A can be activated only with in = 1. The activation takes place when either the CAN bus messages a or b is present...."
Results

• 62 monitor, 10 design models created for one major subsystem (50 pages of requirements documentation)

• Enhancements to the monitor architecture

• Verification results
  – 11 inconsistencies in requirements
  – Why?
    • Evolving document
    • Multiple teams
    • “The implementors will know what to do”
Monitor Model Architecture Change

Needed for *conditional* requirements

- Behavior only specified for certain situations
- “If timeout occurs do something”
Effort (Person-hours)

- Monitors, 53
- Design, 26
- Verification, 25
- Reqs. comp., 40
- Diagnosis, 10
Discussion

• Requirements modeling
  – 53 hours (34.4% of total) 62 reqts. (50 min. / reqt.)

• Design model development
  – 26 hours (16.9%) 10 models (2.6 hrs. / model)

• Verification
  – 25 hours (16.2%) 62 reqts. (25 min. / reqt.)

• Fault diagnosis
  – 10 hours (6.5% of total) 11 errors (55 min. / error)
Applying IBV to the GIP

• Goal: ensure safety strategies implemented correctly in GIP model

• Overview of approach
  1. Identify safety requirements (hazard + mitigation) in safety analysis
  2. Devise monitor model
  3. Instrument GIP model with monitor model
  4. Compute tests to determine if there is a violation
Instrumentation Based Verification

Step 1

Encode a safety requirement as a monitor model

When infusion is in progress and the infusion is paused for more than 10 minutes, an alarm should be sounded
When is the requirement valid?

Obtained from model under verification

Are ideal and expected equal?

Ideal output according to req.

Is this always value 1 for all simulation time instants?
Step 2

Instrument the design model with monitor
Instrumentation Based Verification

Step 3
Specify a model coverage criterion.

Step 4
Automatically generate a test-suite
Objective: Try to make output of monitor model “0”.

© 2012 University of Maryland and Fraunhofer USA
Instrumentation Based Verification

If assertion violated, identify test which led to violation and debug

Tests may be applied to code/system. Are the outputs consistent with the outputs of TS?
Benefits For Medical Software Certification

• IBV provides traceability between safety requirements and test cases for design/code
  – Vital part of 510(k) submissions

• The test suites can serve as evidence for assurance cases
Case Study Results

- Models of a GIP constructed
- Safety requirements created from hazards analysis
- 26+ safety requirements converted into monitor models
- Verified against GIP model (bugs fixed)
- Time frame / effort: 6 months, 3 person-months
Ongoing Medical-Device Work at Fraunhofer

- Testing and V&V consulting
- Software architecture analysis
- Security reviews / recommendations
- Software-process audits
- …
Conclusion

• MBD: future of high-confidence software systems like medical devices?
• Model-based V&V techniques have the potential to produce safer software more efficiently
• Instrumentation-based V&V shows a lot of promise in this respect
Acknowledgements

Raoul Jetley, Yi Zhang, Paul Jones
Division of Electrical and Software Engineering, Office of Science and Engineering Laboratories, Center for Devices and Radiological Health, Food and Drug Administration

Project funding: National Science Foundation
Thank You!

Rance Cleaveland
University of Maryland / Fraunhofer USA CESE
rcleaveland@fc-md.umd.edu
+1 240-497-2905
www.fc-md.umd.edu