IFIP WG 10.4 Dependable Computing and Fault Tolerance

Challenge to Open Systems Dependability

January 22, 2010

Mario Tokoro
Sony Computer Science Laboratories, Inc.
Research Supervisor of Dependable Embedded OS Project, CREST, JST

CREST: Core Research for Evolutonal Science and Technology
JST: Japan Science and Technology Agency
© 2010 Mario Tokoro
Background

• Japan Science and Technology Agency selected Dependability as one of its strategic research areas in 2005 and launched Dependable Embedded OS Project in 2006.
• The budget is about $50M total over 7 years.
• Surprisingly, I was assigned as the Project Supervisor! (maybe because the theme is not pure research but for practical applications and I have background of both academia and industry)
• Then, the story begins....
What is Dependability?

• In the spring of 2006, we chose 5 research teams (Ishikawa, Tokuda, Nakajima, Sato, and Maeda) for 5 years and started discussion of *What is Dependability*. (We selected 4 additional teams in 2008.)
  – Dependability is Reliability? Or, Safety?
  – Dependability is Security? Or, combination of these?
  – How can we treat human factors?
  – How can we cope with networking?
  – The famous Avizienis paper covers all aspects of Dependability?

• A lot of discussions, but No answer!!!!!!!!!!!!!!!

January 22, 2010 Mario Tokoro

© 2010 Mario Tokoro
Then, what are Threats?

• Safety is freedom from injury or risk....
• Security is freedom from danger and harm....
• Security is to protect our daily life, property, privacy and life
  – From physical damage (natural disaster, accident, misuse,...) and aging
  – From malicious attack
  – From design, manufacturing, and operation errors
  – From unexpected ways of use, ...
• Can we really get rid of these threats?
What are the history of Dependability and Safety?

Dependable Computing
- Elemental Technology
  - Reliability (computers)

Functional Safety
- Elemental Technology
  - Safety (parts & devices)
  - Functional Safety (computer controlled system)

Architecture
- RAS/RASIS
  - (online systems)

Architecture & Verification
- MAPE Loop
  - Autonomic (networked services)

Software Processes
- Elemental Technology
What is the history of Software Engineering?

- 1967 SIMULA
- 1982 Smalltalk-80
- 1995 Java
- 1986 Booch’s OOD
- 1991 Rumbaugh’s OMT
- 1995 Jacobson’s OOSE
- 1997 UML/2004 UML2.0
- 1989 CMM
- 2000 CMMI
- 2006 ~
- ultra-large-scale system
- open source software
- system of systems engineering
- software process improvement
- modern project management
- software crisis
- structured programming
- 1972 E.W.Dijkstra
- 1960’s ~
- 1981 ~

January 22, 2010 Mario Tokoro
Standards and Guides

• Standards
  – IEC 61508: Functional Safety
  – IEC 60300-1: Dependability Management
  – IEC 60300-2: Dependability Program Elements and Tasks
  – etc.

• Guides
  – CMMI: Capability Maturity Model Integration
  – DO-178B: Software Considerations in Airborne Systems and Equipment Certification
  – MISRA-C: Guidelines for the Use of the C Language in Vehicle Based Software
  – IEC 61713: Software Dependability through the Software Life-Cycle Processes
    – Application Guide
  – IEC 62347: Guidance on System Dependability Specifications
  – etc.
What are the Demands?

• Demands for the dependability of huge, complex, integrated systems
  – which are connected through networks that may cause security and integrity problems
  – which include black box software resulted by legacy codes and off-the-shelf components
• Demands for coping with environmental and requirement changes in operation
  – functions, user interfaces, performance, etc
  – networks and services on networks
• Consciousness to performance/cost over lifecycle
• Increased accountability to service/system providers
Can We Satisfy Such Demands?

- Can we consider all the events that would happen in the system’s life cycle?
- Can incidents are really avoidable?
- How can we assure that our system is dependable?
- Can we really build a dependable system?
- Can we prove that a system is dependable?
Maybe Not....

• We need to shift our viewpoint from
  – *Designing a system to prove its dependability*
• to
  – *The way of Implementing and operating a system*
  – *and explaining satisfactorily in case of incidents.*

• That is, *Risk Management* and *Accountability* become the main issue.
Do We Need a New Approach?

• Previous approaches to huge, complex, ever-changing integrated systems were based on the *Closed Systems Hypothesis*
  – we supposed a system can be composed of *complete* components
  – we supposed we can know the *whole* system and the *behaviors* of the whole system

• However, the hypothesis *cannot* hold, due to
  – the *incompleteness* of specifications and implementations
  – the *uncertainty* of environment and requirement changes to systems in operation

• We may need to treat a system as *an Open System*. 
What are *Open Systems* and *Open Systems Dependability*?
Closed Systems vs. Open Systems
External View

Closed Systems

No interaction with outer world

Open Systems

Interaction with outer world by channels or membrane

\[ E(t+\Delta t) = f(E(t), O(t)) \]

\[ O(t+\Delta t) = g(O(t), E(t)) \]
Closed Systems vs. Open Systems
Internal View

Closed Systems

- Whole problems are solvable by dividing into elements and collecting answers from the elements.
- It consists of subsystems with simple structure.
- The structures, relations, boundary conditions, and functions of subsystems are statically defined.

Open Systems

- A system cannot simply be decomposed into subsystems. Entire behavior emerges from all interactions among subsystems.
- Time development and irreversible systems
- The structures, interactions, boundary conditions, and functions of each subsystems change dynamically.
## Closed Systems vs. Open Systems
### Summary

**Closed Systems**
- Simple closed system.
- (mainly) Equilibrium system.
- Reversible.
- Reproducible.
- Can be divided into elements.
- Can be halted.
- An n-body problem.
- Can take external observers’ view.

**Open Systems**
- Open complex system.
- Temporal developmental system.
- Irreversible.
- One-time only (non-reproducible).
- Cannot be divided into elements.
- Need to keep alive; cannot stop.
- An n-system problem.
- Can take only the internal observers view.
Open Systems Dependability (1)
Target and Objective

• A huge, complex, ever-changing, integrated system can be seen as an open system
• It has the potential for incidents due to
  – the *incompleteness* of specifications and implementations
  – the *uncertainty* of environment and requirement changes to systems in operation
• We need to secure dependability of a huge, complex, ever-changing integrated system over lifecycle in a practical way, based on the perspective of Open Systems
• The purpose of Open Systems Dependability is to remove incident factors before they cause an incident, to provide appropriate and quick actions when an incident occurs, to manage an incident in order to minimize the damage, and to safely and continuously provide the services expected by users as much as possible.
Open Systems Dependability (2)
Definition

• Dependability is the degree of *Accountability*

• Accountability is secured by showing *evidences* of having done and doing *Risk Management in best effort*
  – to provide *expected services continuously*,
  – to manage quickly and properly *to minimize damages* when an incident occurs
  – to take countermeasures *never to let the same incidents occur again*
Open Systems Dependability (3)
Technological Elements for Achievement

- Open Systems Dependability is achieved by
  1. elemental technology,
  2. architecture, and
  3. process and management
DEOS Project

Dependable OS for Embedded Systems Aiming at Practical Applications

- A project under Japan Science and Technology Agency (JST)
- Roughly $50M in total over 7 years started in 2006
- 5 teams selected in 2006 and 4 teams in 2008
- To develop Dependable Embedded OS based on the notion of Open Systems Dependability
- Based on Linux (for users’ adaptability)
- To make publicly available in 2014 through consortium
- R&D Center (DEOSC) was established in 2007 for supporting development, integrating technologies developed by the teams, and promoting the use.
- The first interim evaluation was made Sep. 2009.
Organization

Area Management Advisers

Area Advisers

Research Supervisor
Deputy Research Supervisor

Research Promotion Board

Consortium

DEOS R&D Center
- System Architecture
- Framework
- Reference System
- Management Process
- Development Environment
- Demonstration System
- Maintenance

Research Teams
FY 2006 Research Teams
- Elemental Technologies
Core Team
- System Architecture
- Metrics, Benchmark, Configuration
- D-Case
FY 2008 Research Teams
- Elemental Technologies
- Standards, Guidelines, Int'l Standardization

Public Relations
Information Exchange
Publication & Advertisement
Dissemination and Promotion

External Development Resources

Community

January 22, 2010

© 2010 Mario Tokoro
R&D Items

• Elemental Technologies
  – P-Bus as a standard interface for verified kernel modules,
  – VM for multi-core processors; for monitoring, logging, and countermeasures
  – Type/Model-based verification
  – Precise real-time scheduling and power management

• System Architecture
  – The framework (Systems design guideline in conjunction with Monitoring, Logging, Management Policy, etc.)
  – D-Core (Dependability Case for stakeholder agreements, configuration manager, DS-Benchmarking, etc.) which can be used also for system changes in operation

• Management Processes
  – Processes for PDCA (using available software processes, etc)
  – Processes for Risk Management
  – Standardization for Accountability (risk management with evidence) with Dependability Level
Realizing Open Systems Dependability

System Architecture

Framework
- Application Monitoring & Management
- Policy Management & Control
- Evidence Management & Logging
- System Monitoring & Management

D-Core
- Metrics / D-Case
- Configuration
- DS-Bench

Technologies for Design and Development
- Type/Model Verification
- WCET Analysis
- WCAT Analysis
- DS-Bench Runtime Environment
- Fault-Injection

Technologies for System Safety and Implementation
- Kernel Extension Base
- Virtual Monitor
- Real-time Processing
- Multi-core Support
- System Reconfiguration

Technologies for Maintenance and Operation
- Fault-tolerant of Network
- Fault-tolerant of Node
- Fault-tolerant of Software
- Security-attack Protection
- Predictive Detection

Process
- Design/Development Phase
- Maintenance/Operation Phase
- PDCA

Standardization
- Evaluation Criteria
- Int’l Standards
- Guidelines

Processes and Management

Elemental Technologies

Mario Tokoro January 22, 2010
Elemental Technologies

Main technologies:
- P-Bus
- VM for multi-core processors
- Type/Model-based verification

Technologies for Design and Development
- Type/Model Verification
- WCET Analysis
- WCAT Analysis
- DS-Bench Runtime Environment
- Fault-Injection

Technologies for System Safety and Implementation
- Kernel Extension Base
- Virtual Monitor
- Real-time Processing
- Multi-core Support
- System Reconfiguration

Technologies for Maintenance and Operation
- Fault-tolerant of Network
- Fault-tolerant of Node
- Fault-tolerant of Software
- Security-attack Protection
- Predictive Detection
Main Technologies
- Use available processes
- Process for PDCA
- Contribute to standardization

Processes and Management

Process
- Design/Development Phase
- Maintenance/Operation Phase
- PDCA

Standardization
- Evaluation Criteria
- Int’l Standards
- Guidelines
Schedule

• Sep. 2010: setting up users group
• Sep. 2011: first completion of *Framework* and *Reference System*; and drafting first *standard specification of PDCA process and management*
• Mar. 2014: second completion of *Framework, D-Case, and Reference System* taking feedbacks from consortium members; second drafting *standard specification of PDCA process and management*; all the activities will be transferred to the consortium
Summary

• We proposed a new perspective for a huge, complex, ever-changing integrated system as an open system.
• A huge and complex system inherently has incident factors due to incompleteness and uncertainty.
• We proposed a new approach called Open Systems Dependability.
• Open Systems Dependability is the degree of Accountability which is secured by showing evidences of having done and doing Risk Management in best effort.
• Open Systems Dependability is achieved by elemental technologies, architecture, and process and management.
• We are proving our perspective and method through the DEOS project.
Thank you

JST/DEOS Project

JST/DEOS Center
http://www.dependable-os.net/index-e.html

Sony Computer Science Laboratories, Inc.
http://www.sonycsl.co.jp/en/
Thank you