Altreonic

Systems Engineering for the masses with GoedelWorks and OpenComRTOS

Eric Verhulst, CEO/CTO

www.altreonic.com

Trustworthy Forever

Company profile

- History goes back to 1989 (Eonic Systems)
  - Specialised in parallel RTOS (T800, C40, C6x, 2016x, TS102, G4, …)
  - Used from 1 CPU to 1600 DSPs (sonar, radar) to 12000 nodes (heterogeneous (sensing + 3D deconvolution))
  - Acquired by Wind River systems in 2001
- Altreonic: created as new spin-off in 2008 after R&D
  - Unified systems engineering methodology
  - Formalised when possible => OpenComRTOS project
  - Covers from early requirements capturing till final hardware
  - Focus on trustworthy scalable embedded systems
    - Safety, Security, Usability, Privacy
    - Unique "Open Technology License" model

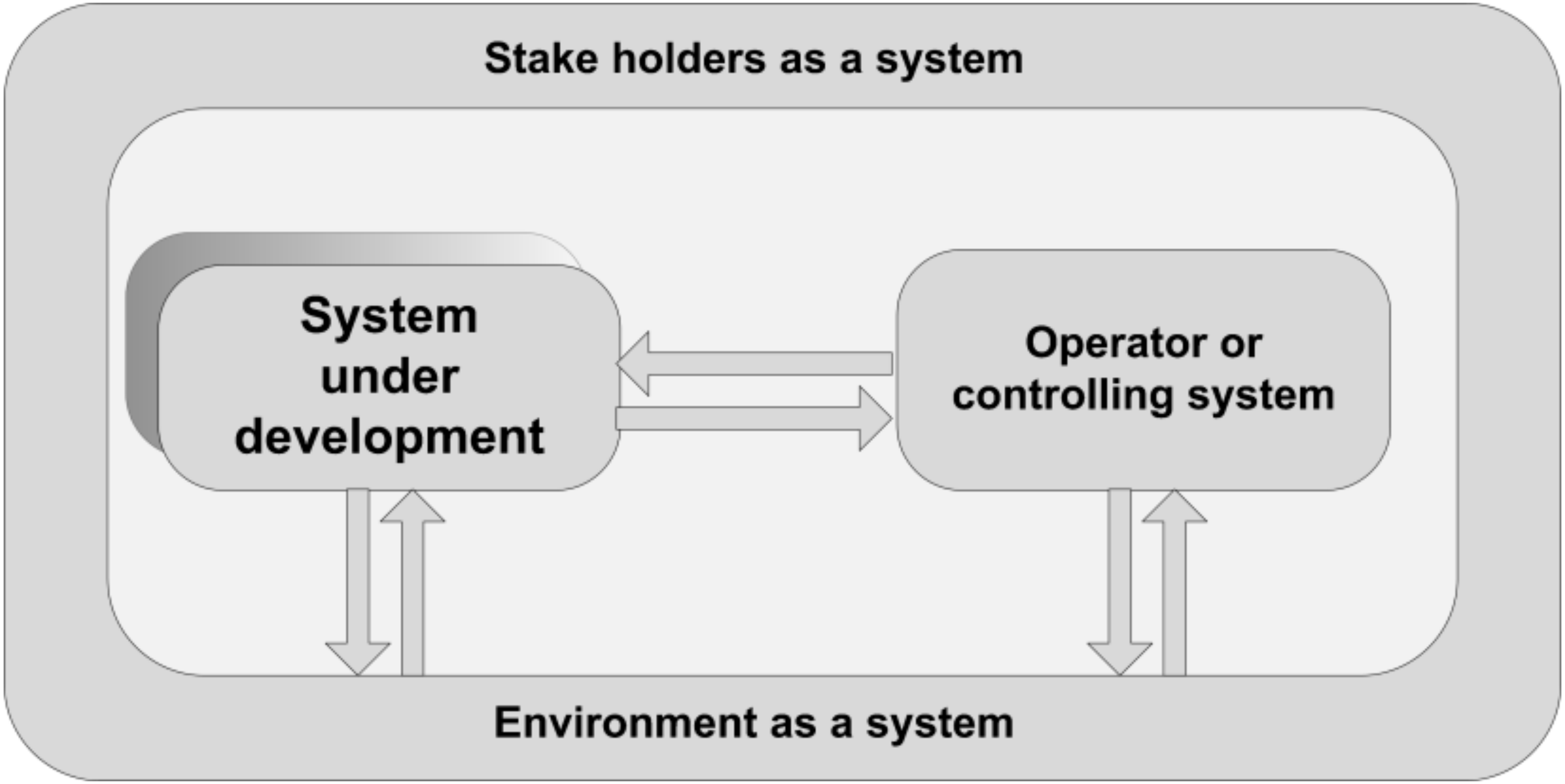
From R&D to products

- R&D projects
  - Metamodel for systems engineering: “systems grammar”
  - Formal development of network-centric OpenComRTOS
- **Other R&D projects:**
  - **EVOLVE** ITEA project
    - **Evol**utionary **V**alidation, **V**erification and C**e**rtification
  - FP7 OPENCOSS project:
    - Cross domain certification (automotive, railway, avionics)
  - **ASIL**: Flanders Drive project on developing a common safety engineering methodology for automotive
  - Artemis **Crafters**. – Dynamic resource scheduling in MPSoC
- Currently **GoedelWorks** and **OpenComRTOS Designer**

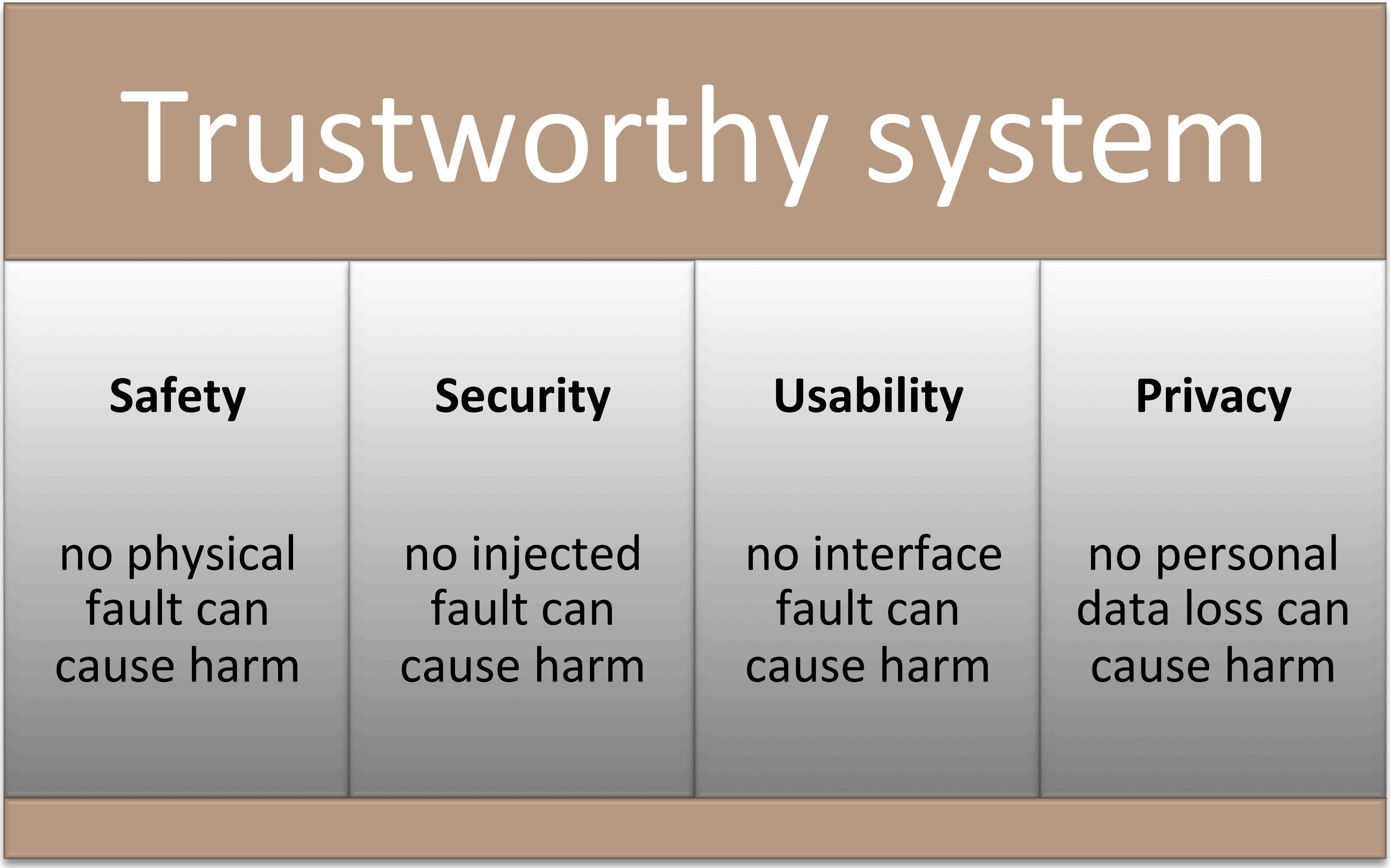
What is systems engineering?

- A system: from component to System-Of-Systems
- Engineering = a controlled/managed process
- Requires: skills, knowledge, systematic approach
- Good engineering = the system can be trusted + cost-efficiency
- Certification: engineering produces the evidence as well
- Important when safety is at stake

A system is never alone



Trustworthiness as a goal



Trustworthy system

Safety

no physical fault can cause harm

Security

no injected fault can cause harm

Usability

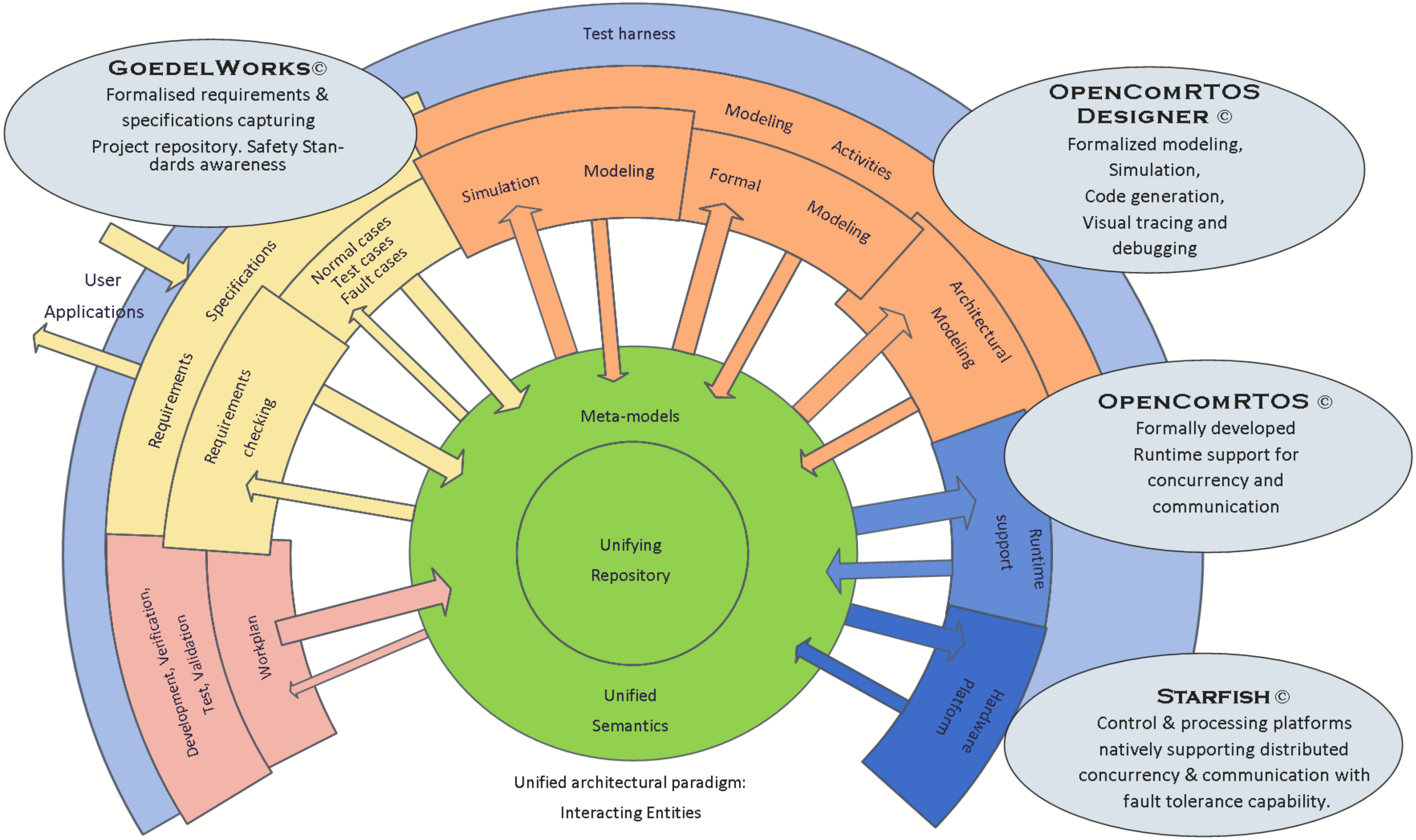
no interface fault can cause harm

Privacy

no personal data loss can cause harm

The methodology picture

A coherent approach to systems and safety engineering



GoedelWorks©
Formalised requirements & specifications capturing
Project repository. Safety Standards awareness

OpenComRTOS Designer ©
Formalized modeling,
Simulation,
Code generation,
Visual tracing and debugging

OpenComRTOS ©
Formally developed Runtime support for concurrency and communication

Starfish ©
Control & processing platforms natively supporting distributed concurrency & communication with fault tolerance capability.

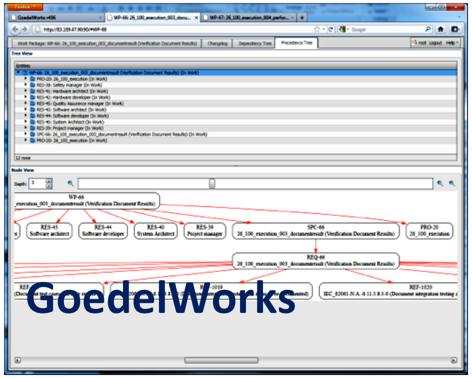
Unified architectural paradigm:
Interacting Entities

Two paradigms

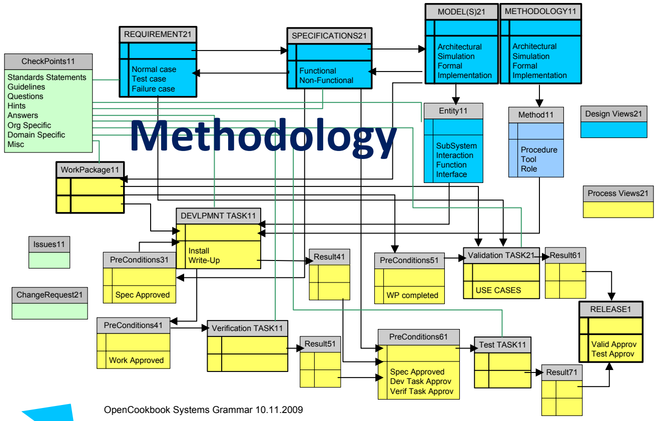
- Unified Semantics:
  - Use the same language everywhere
  - Standardize on terminology
  - Keep it orthogonal and clear
- Interacting Entities
  - Architectural model of any system
  - Interactions are as important as Entities
  - Maps very well on concurrent software

Tools for productivity & predictability

Covering full value-chain from requirement to hardware to maximise added value and certifiability

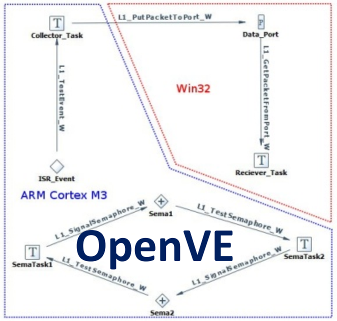


GoedelWorks



OpenCookbook Systems Grammar 10.11.2009

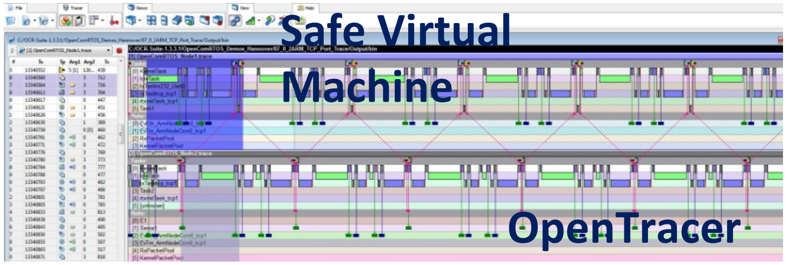
Methodology



OpenVE

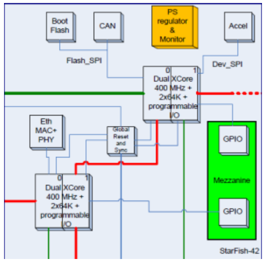
OpenComRTOS Designer

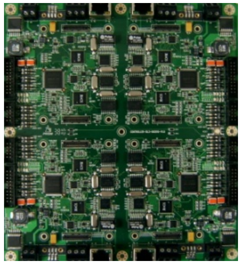
StarFish



Safe Virtual Machine

OpenTracer





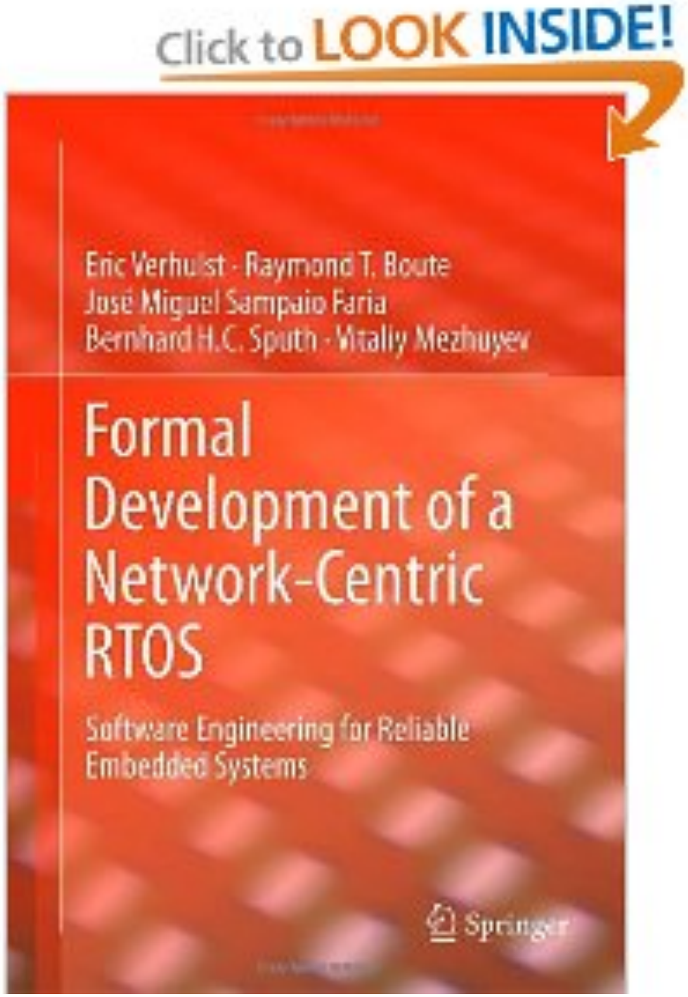
OpenComRTOS Designer

Program seamlessly
embedded real-time applications
from 1 CPU to
a network of heterogeneous processors
in just 5-10 KBytes/node

Unique software technology

- Formalised but straightforward approach
- OpenComRTOS is a unique programming system, a unique network-centric RTOS, quasi-universal, MP by default
  - Formally developed and verified
  - Concurrency at the core (“Interacting Entities”)
  - Pragmatic superset of CSP (Hoare)
  - Scalable yet very small: typically 5 to 10 kiB/node
  - Real-time communication as system level service
  - Unique support for Distributed priority inheritance
  - Heterogeneous target /communication support (2**24 nodes)
  - Integrate seamlessly “legacy OS” nodes
  - Virtual Single Processor model
  - Visual modelling/ programming with code generators
  - OpenComRTOS nominated embedded award 2009
  - Capable of fault-tolerance and resource management

Book on the formal development



Click to LOOK INSIDE!

Eric Verhulst · Raymond T. Boute
José Miguel Sampaio Faria
Bernhard H.C. Sputh · Vitaliy Mezhuyev

Formal Development of a Network-Centric RTOS

Software Engineering for Reliable Embedded Systems

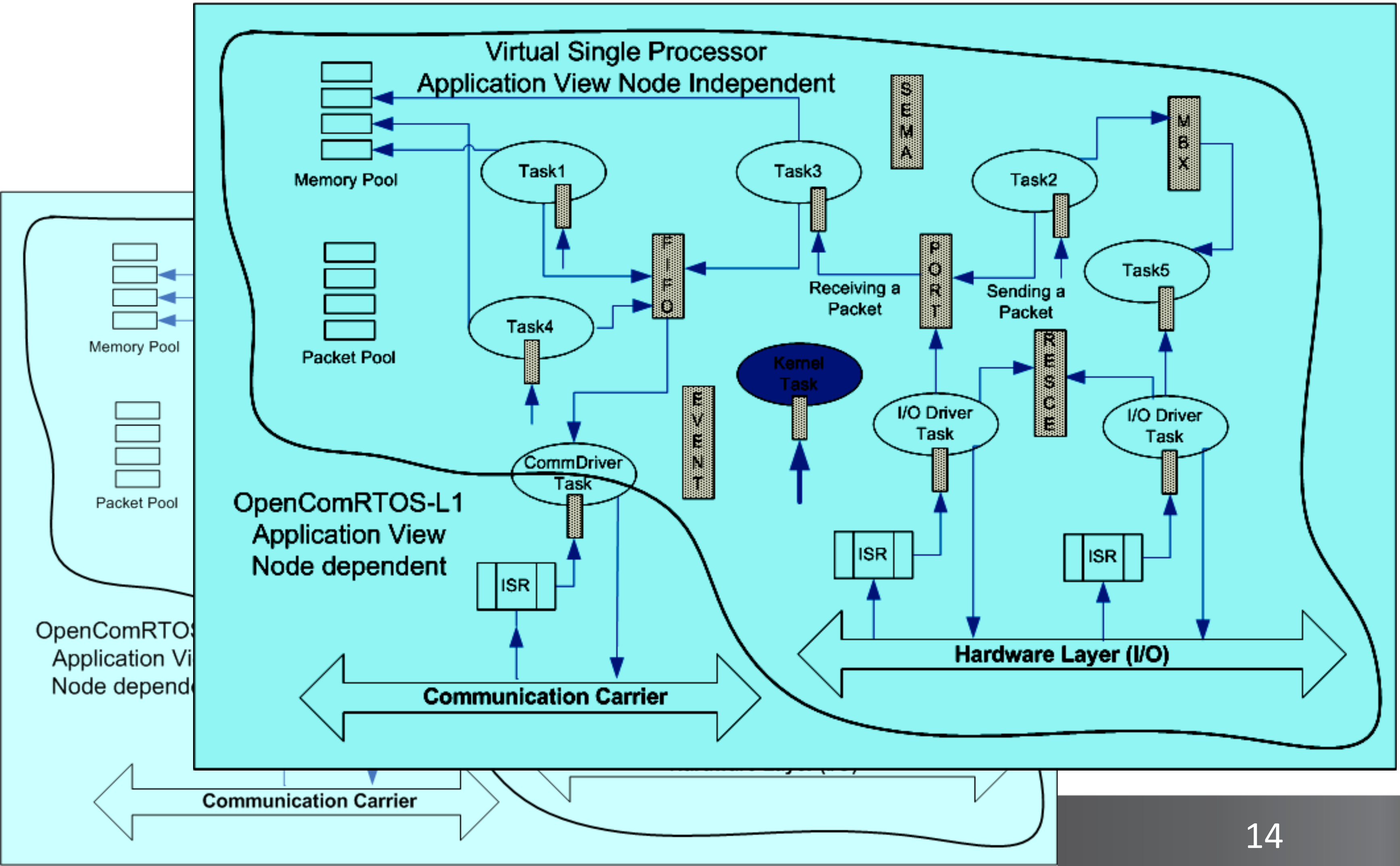
Springer

Embedded computing

- Works on real-time data (GBytes/sec I/O)
- Often power and size constraints
- Hard real-time = predictable (guaranteed)
- Soft real-time = statistical (best effort)
- Performance = latency + througput
- Many processor types:
  - Microcontrollers, RISC, DSP, FPGA
  - Single core to many-core
- How to program: OpenComRTOS Designer

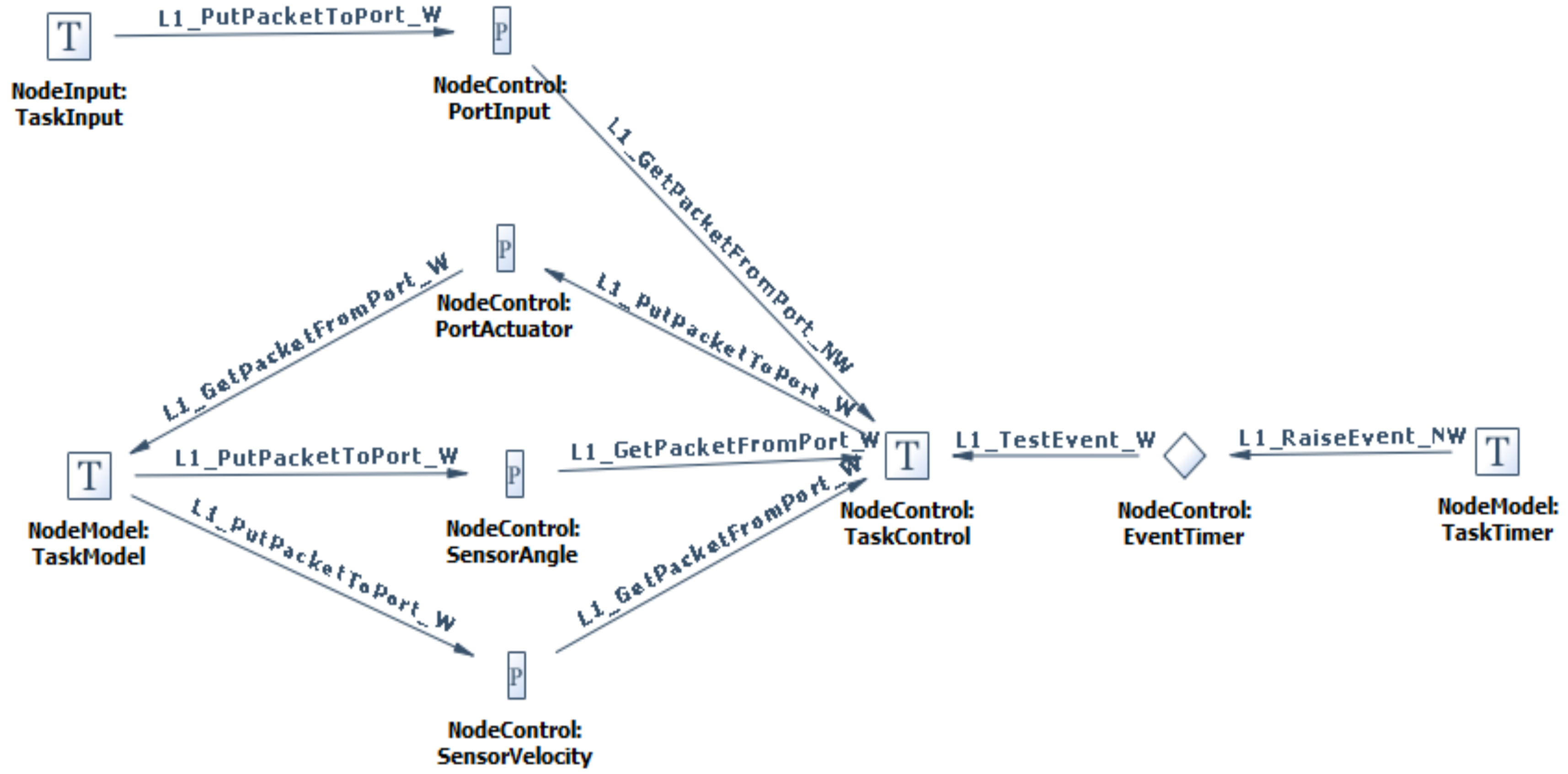
OpenComRTOS Interacting Entities

Any entity can be mapped anywhere in the target system

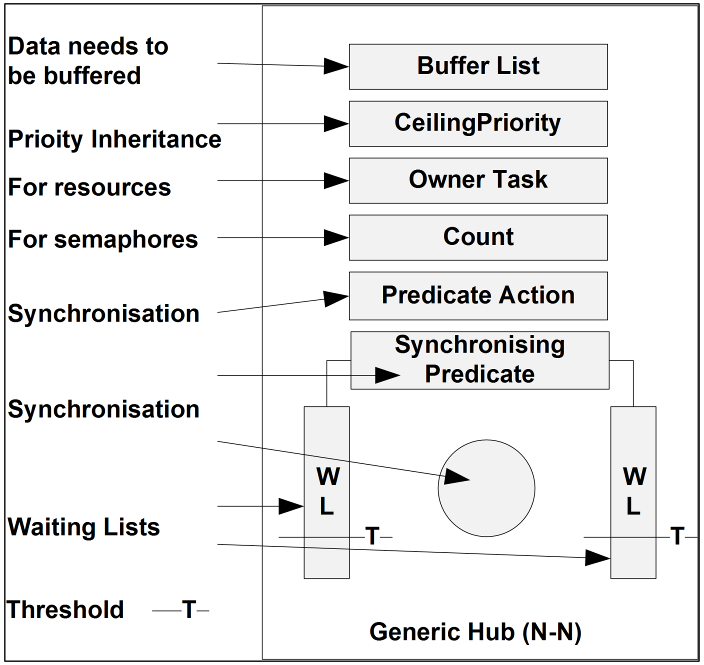


Visual Designer

- Model visually, regenerate model from program

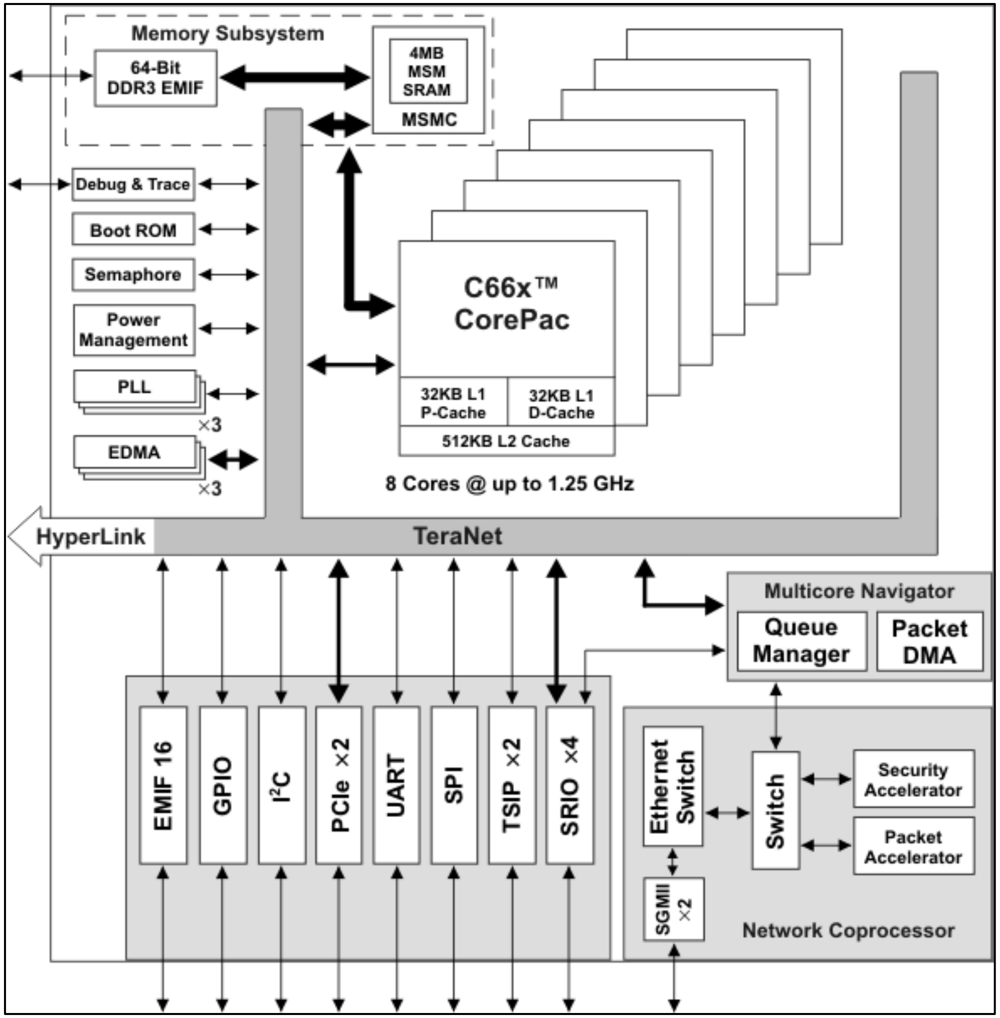


OpenComRTOS' Hub



- Hub as core interaction mechanism between processing Tasks
  - Event, semaphore, resource, FIFO, Port, memory pool, …
- Decouples tasks with N-to-N semantics
- Acts like "Guarded Atomic Action"

TI TMS320C6678



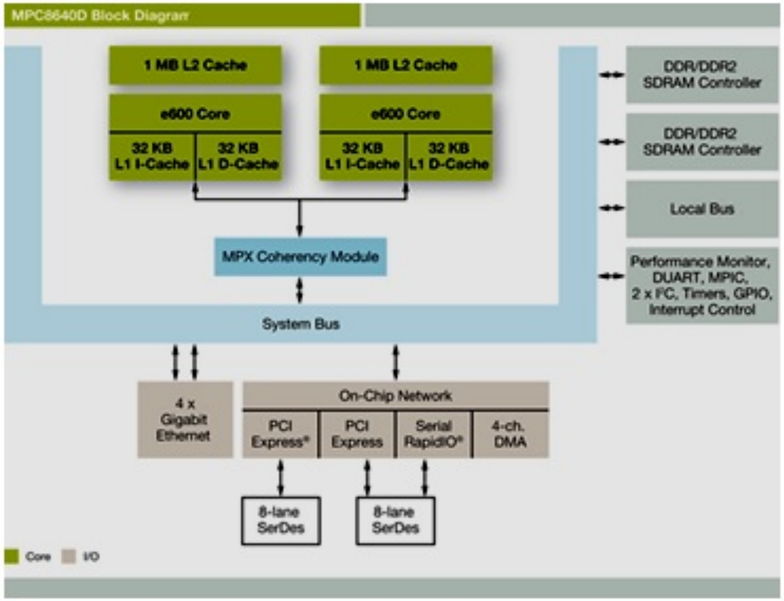
OpenComRTOS code size:

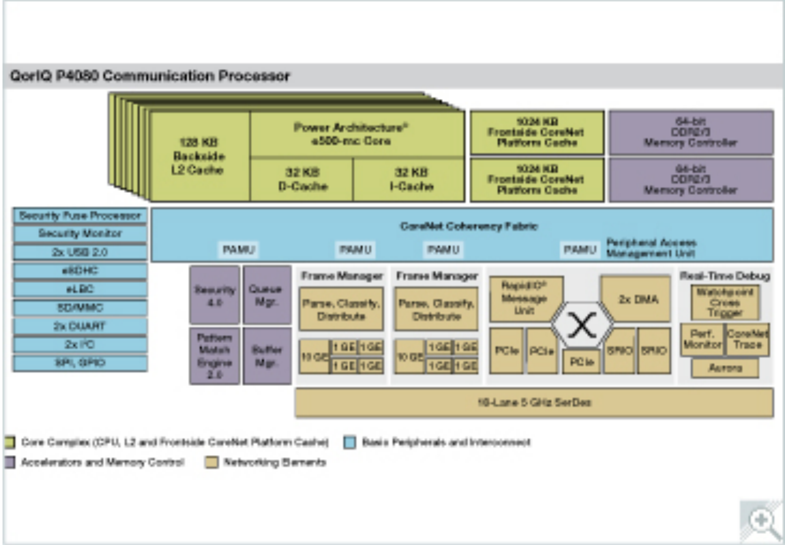
5056 to 7648 Bytes

Interrupt latency tot Task: 1367 cycles

Task to Task switch: about 1125 cycles

Freescale e500/e600 based PPC





OpenComRTOS code size (e600 with Altivec support):

7128 to 9764 Bytes

Interrupt latency tot Task: 896 cycles

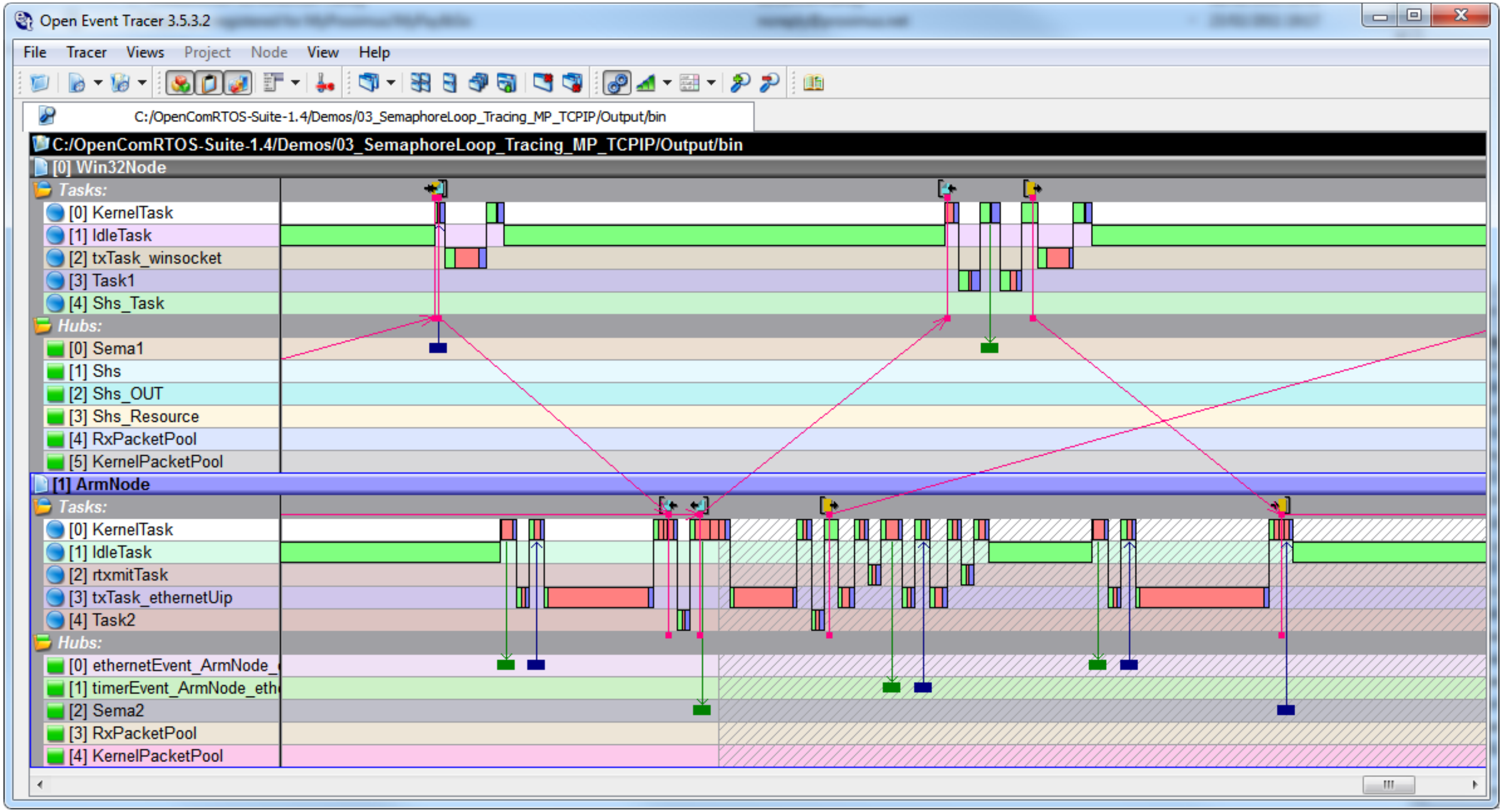
Task to Task switch: about 410 cycles

OpenComRTOS benefits

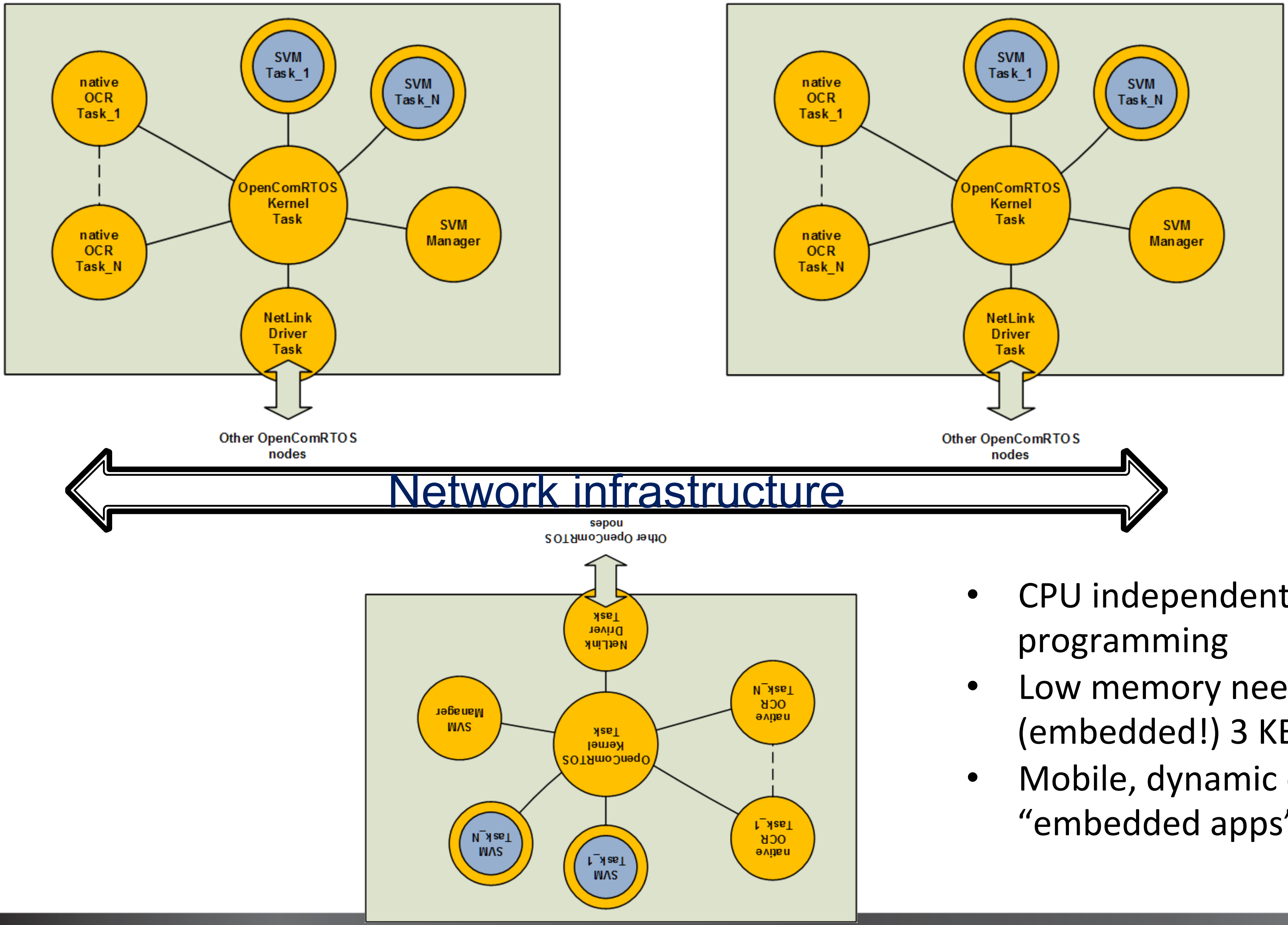
- Small code size:
  - Full kernel fits easily in L1 cache
  - Leaves more performance to applications
- Low latency:
  - Latency is the bottleneck in communications
  - Packet based switching network
- High bandwidth:
  - Communication can be split over all available links
- Trust: Safety and security by design
- Less power requirements
  - Less code, less data, less memory I/O

Event Tracer

- Visualizes: Context Switches, Hub Interactions, Packet exchanges between Nodes.



Safe Virtual Machine



- CPU independent programming
- Low memory needs (embedded!) 3 KB
- Mobile, dynamic code => “embedded apps”

Applications

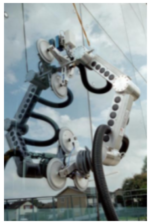
- *Embedded Systems and Control (src EU) has an estimated market size of* **~ €188 billion with av. growth of 8% until 2020**
- ***55% will use standards*** *(src VDC): safety standard become a must*
- *<u>Average</u> lifetime of a processor is only 20 yrs and shrinking*
- *Energy use is increasingly a serious issue*

Safety engineering + low energy = complex control

➡ ***Requires engineering process and trustworthy tools and hardware***

Altreonic has the technology

Distributed Control



- **smart machines**
- **robotic machines**
- **sensing networks**
- **safety critical**
- **avionics**

Fault tolerant system

- **process control**
- **infrastructure**
- **e-vehicles**
- **medical**

Ultra low power

NXP CoolFlux DSP

- *hearing aids*
- *building control*
- *sensors*

Multicore/manycore devices, *Intel*

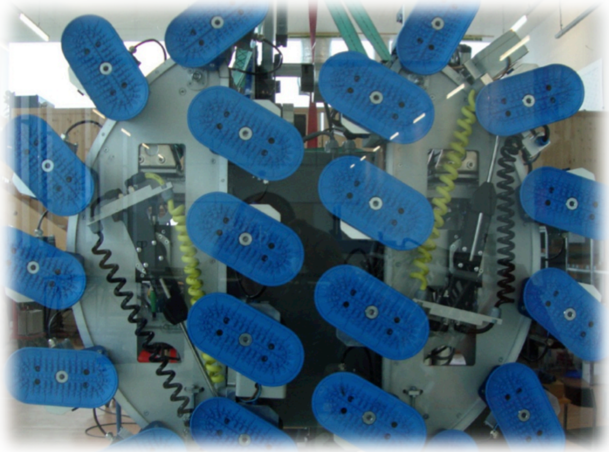
- *servers*
- *smart control*

Parallel embedded supercomputing

ex. TI DSP, PPC, …

- *radar, sonar*
- *image processing*

Use case: distributed robot controller

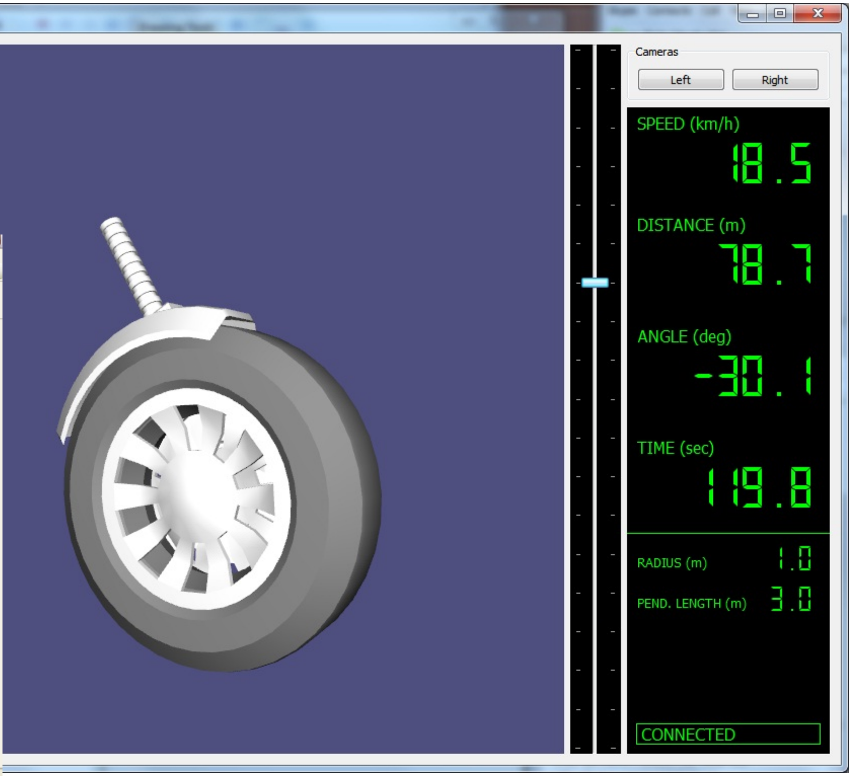


- Smart robot
- Can climb walls
- 42 feet + central controller
- Original design: 7000 €
- Redesign with OpenComRTOS: 1000 €
- Benefit: more scalability, lower cost

- Same architecture applies to the design of electric and hybrid vehicles

eWheel Controller Simulation

- This demonstration simulates a Segway type wheel, and consists of the following parts:
  - eWheel Visualisation
  - eWheel Controller
  - Physical Model


eWheel Demo 1.0.1.6

Cameras

SPEED (km/h) 18.5

DISTANCE (m) 78.7

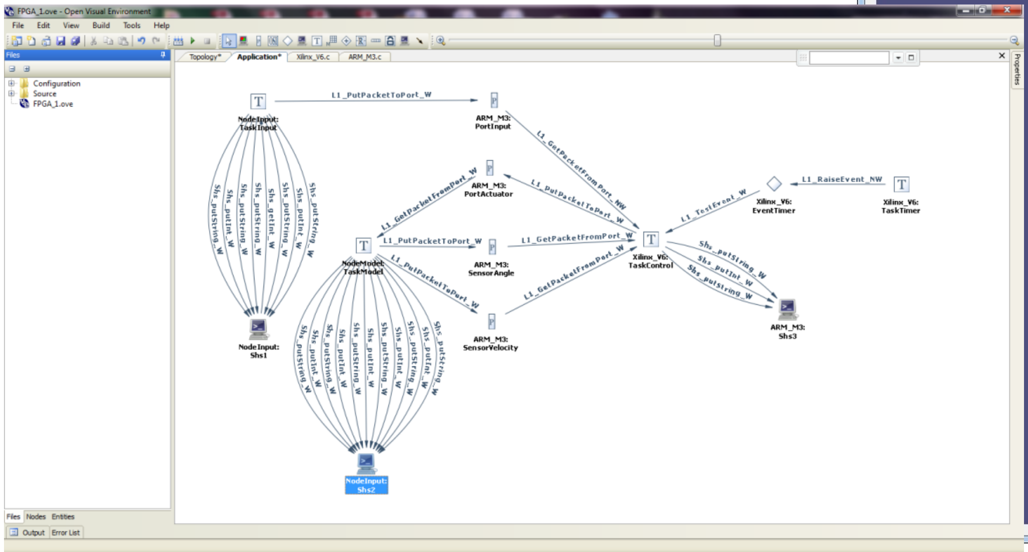
ANGLE (deg) -30.1

TIME (sec) 119.8

RADIUS (m) 1.0

PEND. LENGTH (m) 3.0

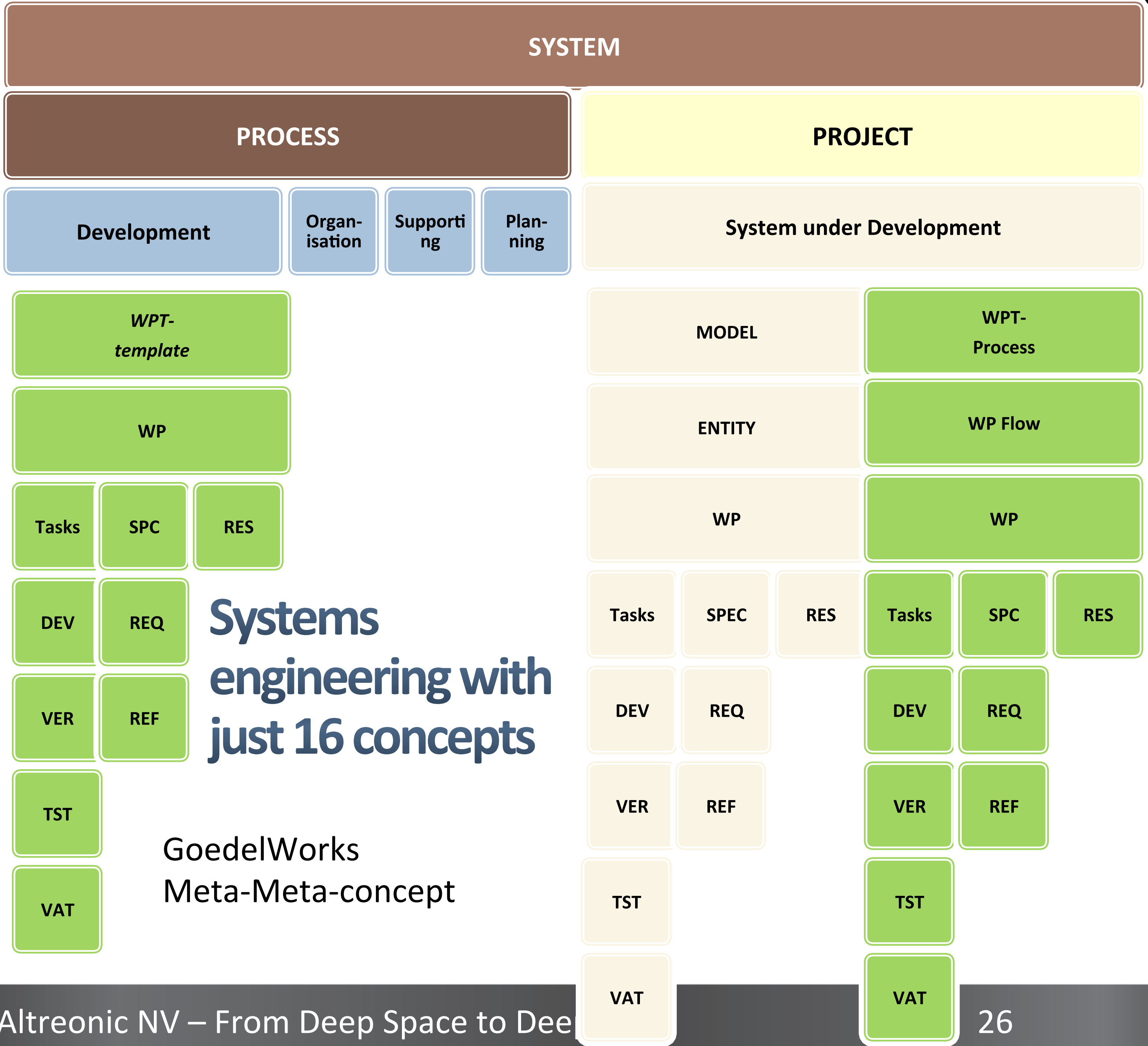
CONNECTED


FPGA_1.ove - Open Visual Environment

GoedelWorks

Develop certifiable products and systems by generating the evidence during development

Systems engineering with just 16 concepts



SYSTEM

PROCESS

PROJECT

Development

Organisation

Supporting

Planning

System under Development

WPT-template

WP

Tasks

SPC

RES

DEV

REQ

VER

REF

TST

VAT

MODEL

WPT-Process

ENTITY

WP Flow

WP

WP

Tasks

SPEC

RES

Tasks

SPC

RES

DEV

REQ

DEV

REQ

VER

REF

VER

REF

TST

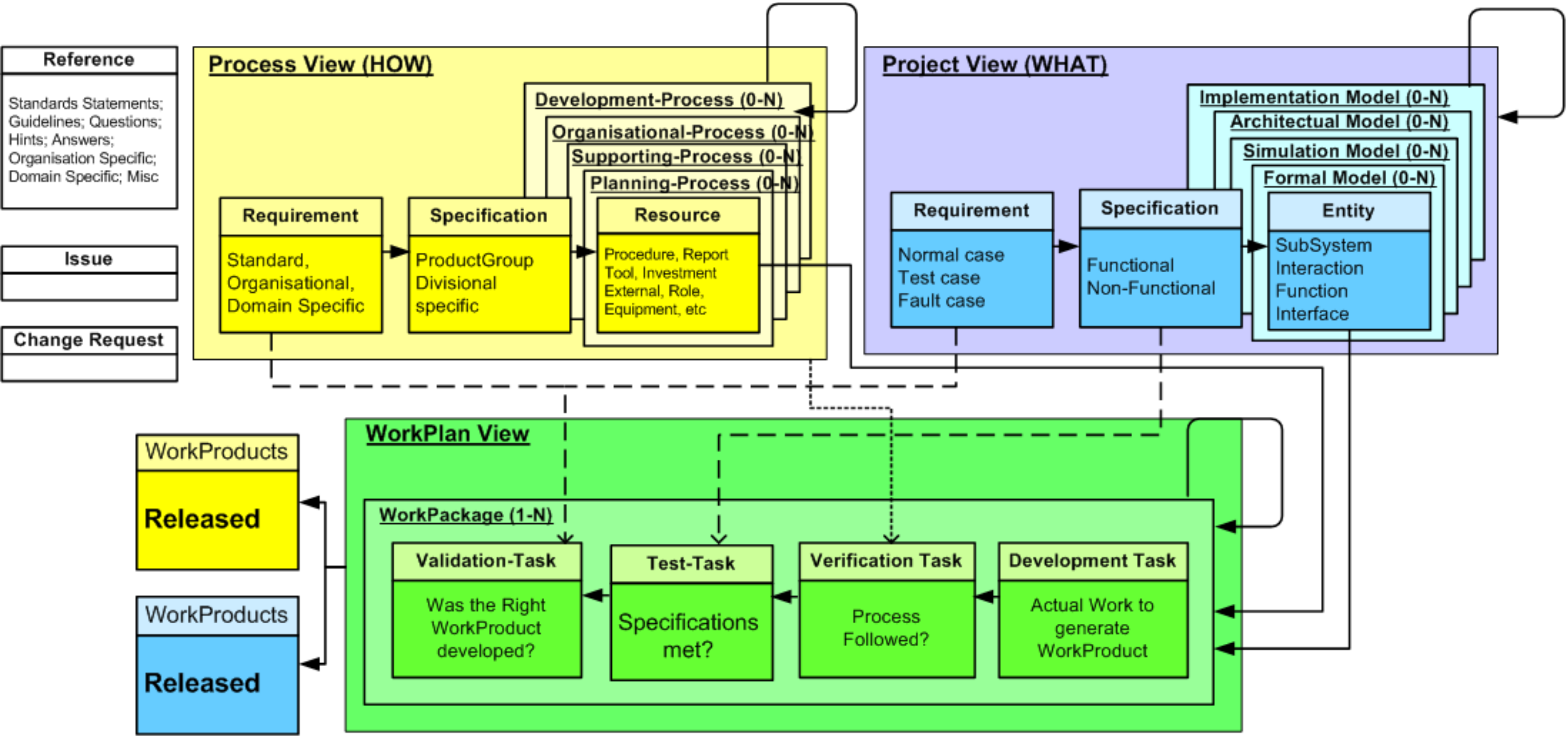
TST

VAT

VAT

GoedelWorks
Meta-Meta-concept

GoedelWorks' combines Process+ Project+WorkPlan views



Reference

Standards Statements; Guidelines; Questions; Hints; Answers; Organisation Specific; Domain Specific; Misc

Issue

Change Request

Process View (HOW)

Development-Process (0-N)

Organisational-Process (0-N)

Supporting-Process (0-N)

Planning-Process (0-N)

Requirement

Standard, Organisational, Domain Specific

Specification

ProductGroup Divisional specific

Resource

Procedure, Report Tool, Investment External, Role, Equipment, etc

Project View (WHAT)

Implementation Model (0-N)

Architectural Model (0-N)

Simulation Model (0-N)

Formal Model (0-N)

Requirement

Normal case Test case Fault case

Specification

Functional Non-Functional

Entity

SubSystem Interaction Function Interface

WorkProducts

Released

WorkProducts

Released

WorkPlan View

WorkPackage (1-N)

Validation-Task

Was the Right WorkProduct developed?

Test-Task

Specifications met?

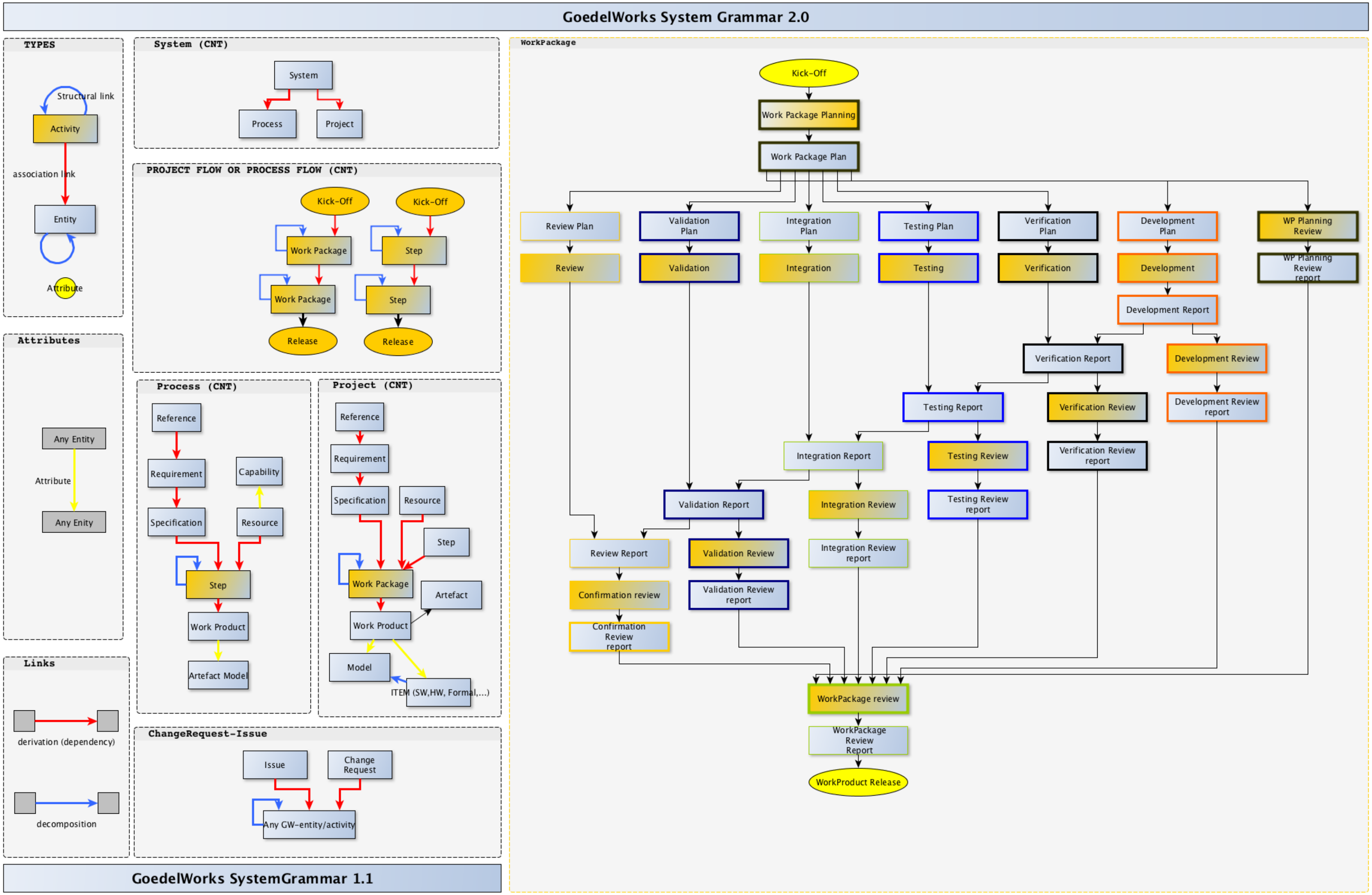
Verification Task

Process Followed?

Development Task

Actual Work to generate WorkProduct

Standard template for WP



GoedelWorks System Grammar 2.0

TYPES

Structural link

Activity

association link

Entity

Attribute

System (CNT)

System

Process

Project

PROJECT FLOW OR PROCESS FLOW (CNT)

Kick-Off

Kick-Off

Work Package

Step

Work Package

Step

Release

Release

Attributes

Any Entity

Attribute

Any Enity

Process (CNT)

Reference

Requirement

Capability

Specification

Resource

Step

Work Product

Artefact Model

Project (CNT)

Reference

Requirement

Specification

Resource

Step

Work Package

Artefact

Work Product

Model

ITEM (SW,HW, Formal,...)

Links

derivation (dependency)

decomposition

ChangeRequest-Issue

Issue

Change Request

Any GW-entity/activity

GoedelWorks SystemGrammar 1.1

WorkPackage

Kick-Off

Work Package Planning

Work Package Plan

Review Plan

Validation Plan

Integration Plan

Testing Plan

Verification Plan

Development Plan

WP Planning Review

Review

Validation

Integration

Testing

Verification

Development

WP Planning Review report

Development Report

Verification Report

Development Review

Testing Report

Verification Review

Development Review report

Integration Report

Testing Review

Verification Review report

Validation Report

Integration Review

Testing Review report

Review Report

Validation Review

Integration Review report

Confirmation review

Validation Review report

Confirmation Review report

WorkPackage review

WorkPackage Review Report

WorkProduct Release

Work Package template

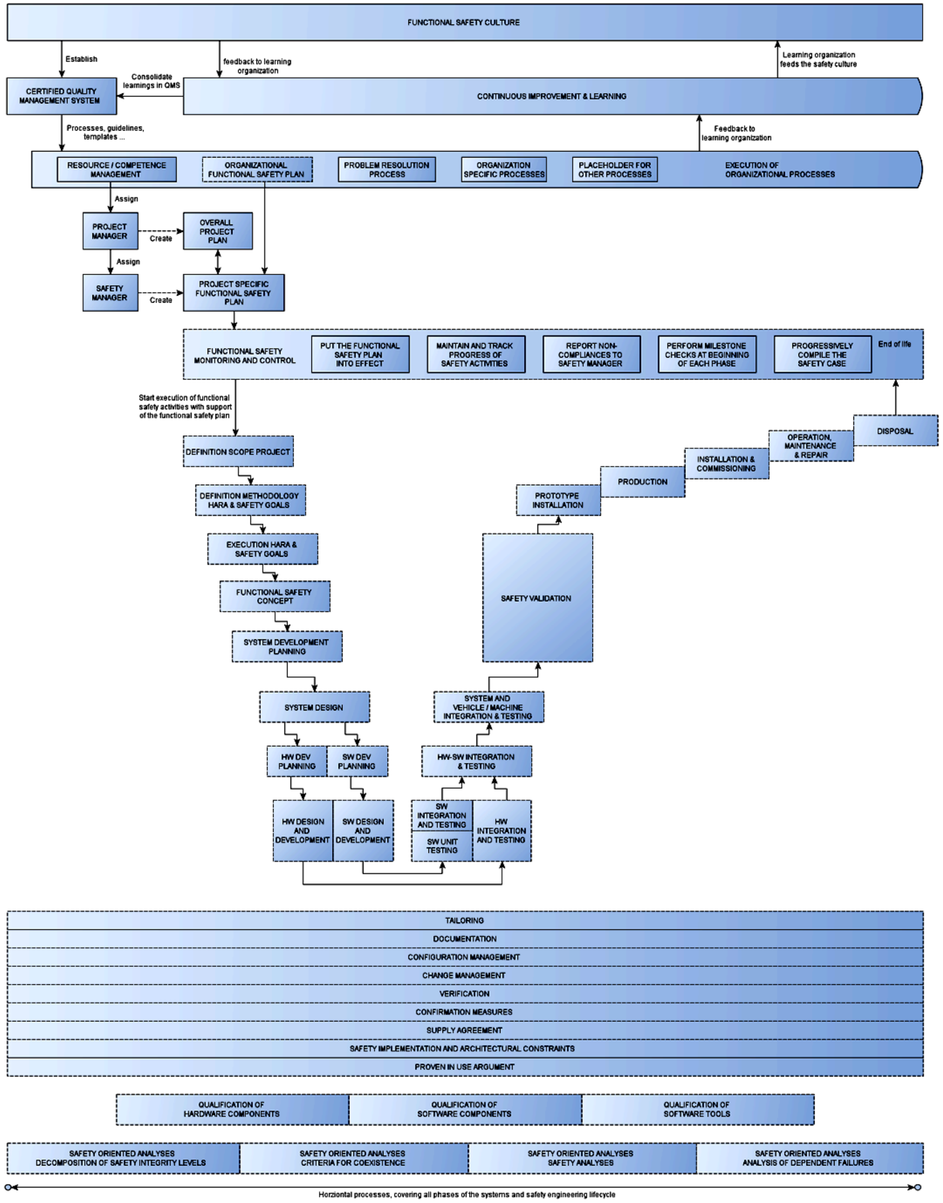
- 7 activities:
  - Planning – Development – Verification – Testing – Integration – Validation – Review
- 4 phases each:
  - Planning – Doing – Document - Confirmation

Validation of GoedelWorks

- Input: ASIL project of Flanders Drive
  - **A**utomotive **S**afety **I**ntegrity **L**evel
- Goal: develop common safety engineering process based on existing standards:
  - Automotive: off-highway, on-highway
  - Machinery
- IEC 61508, IEC 62061, ISO DIS 26262, ISO 13849, ISO DIS 25119 and ISO 15998
- Partners:
  - Altreonic, DANA, EIA, Flanders Drive, Punch Powertrain, Triphase, TüV Nord
- Other standards: customer specific

ASIL V-model

- Organisational
- Safety and Engineering/ Development
- Supporting



FUNCTIONAL SAFETY CULTURE

Establish

feedback to learning organization

Learning organization feeds the safety culture

CERTIFIED QUALITY MANAGEMENT SYSTEM

Consolidate learnings in QMS

CONTINUOUS IMPROVEMENT & LEARNING

Processes, guidelines, templates ...

Feedback to learning organization

RESOURCE / COMPETENCE MANAGEMENT

ORGANIZATIONAL FUNCTIONAL SAFETY PLAN

PROBLEM RESOLUTION PROCESS

ORGANIZATION SPECIFIC PROCESSES

PLACEHOLDER FOR OTHER PROCESSES

EXECUTION OF ORGANIZATIONAL PROCESSES

Assign

PROJECT MANAGER

Create

OVERALL PROJECT PLAN

Assign

SAFETY MANAGER

Create

PROJECT SPECIFIC FUNCTIONAL SAFETY PLAN

FUNCTIONAL SAFETY MONITORING AND CONTROL

PUT THE FUNCTIONAL SAFETY PLAN INTO EFFECT

MAINTAIN AND TRACK PROGRESS OF SAFETY ACTIVITIES

REPORT NON-COMPLIANCES TO SAFETY MANAGER

PERFORM MILESTONE CHECKS AT BEGINNING OF EACH PHASE

PROGRESSIVELY COMPILE THE SAFETY CASE

End of life

Start execution of functional safety activities with support of the functional safety plan

DEFINITION SCOPE PROJECT

DEFINITION METHODOLOGY HARA & SAFETY GOALS

EXECUTION HARA & SAFETY GOALS

FUNCTIONAL SAFETY CONCEPT

SYSTEM DEVELOPMENT PLANNING

SYSTEM DESIGN

HW DEV PLANNING

SW DEV PLANNING

HW DESIGN AND DEVELOPMENT

SW DESIGN AND DEVELOPMENT

SW INTEGRATION AND TESTING

SW UNIT TESTING

HW INTEGRATION AND TESTING

HW-SW INTEGRATION & TESTING

SYSTEM AND VEHICLE / MACHINE INTEGRATION & TESTING

SAFETY VALIDATION

PROTOTYPE INSTALLATION

PRODUCTION

INSTALLATION & COMMISSIONING

OPERATION, MAINTENANCE & REPAIR

DISPOSAL

TAILORING

DOCUMENTATION

CONFIGURATION MANAGEMENT

CHANGE MANAGEMENT

VERIFICATION

CONFIRMATION MEASURES

SUPPLY AGREEMENT

SAFETY IMPLEMENTATION AND ARCHITECTURAL CONSTRAINTS

PROVEN IN USE ARGUMENT

QUALIFICATION OF HARDWARE COMPONENTS

QUALIFICATION OF SOFTWARE COMPONENTS

QUALIFICATION OF SOFTWARE TOOLS

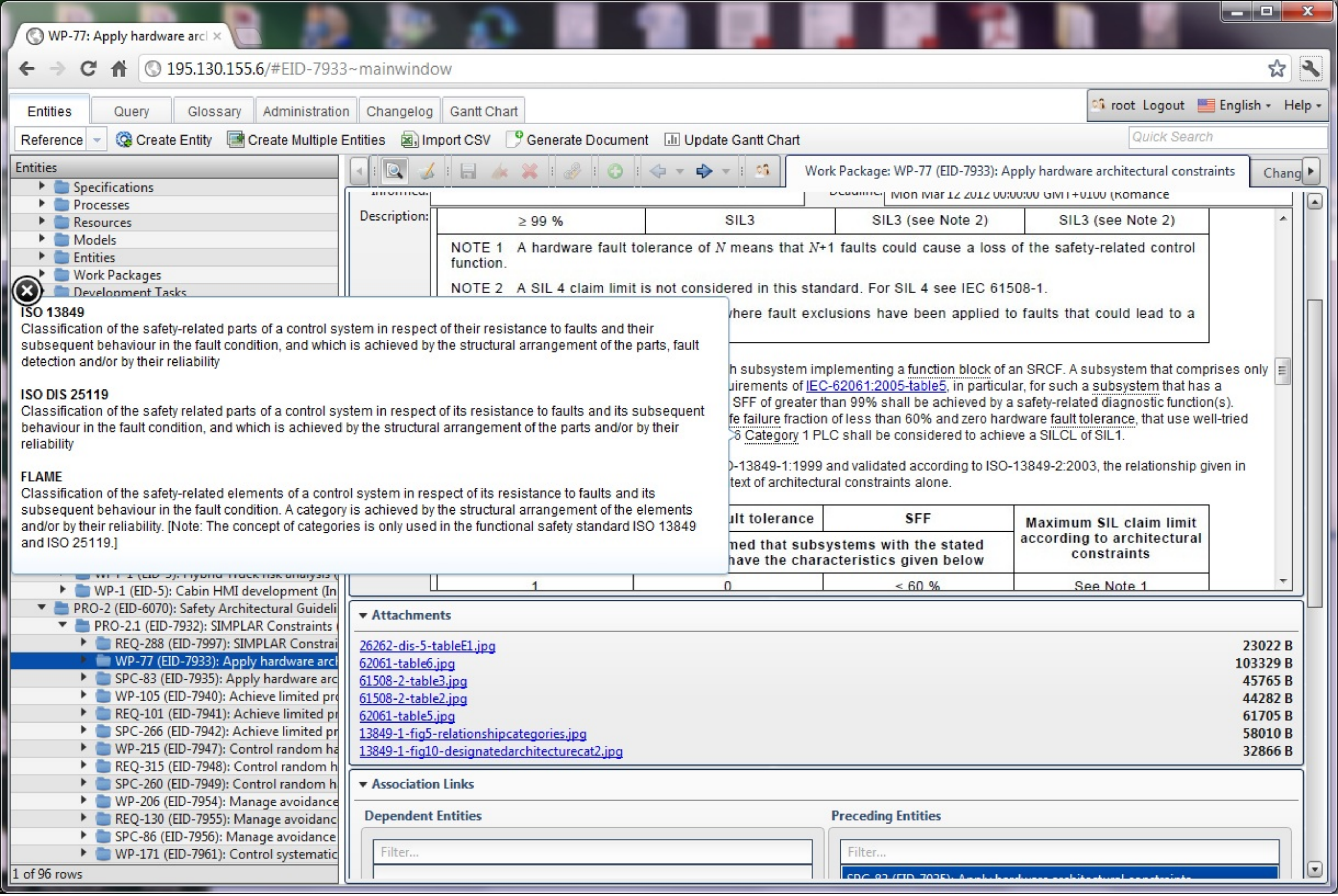
SAFETY ORIENTED ANALYSES DECOMPOSITION OF SAFETY INTEGRITY LEVELS

SAFETY ORIENTED ANALYSES CRITERIA FOR COEXISTENCE

SAFETY ORIENTED ANALYSES SAFETY ANALYSES

SAFETY ORIENTED ANALYSES ANALYSIS OF DEPENDENT FAILURES

Horziontal processes, covering all phases of the systems and safety engineering lifecycle



WP-77: Apply hardware arcl ×

195.130.155.6/#EID-7933~mainwindow

Entities

- Specifications
- Processes
- Resources
- Models
- Entities
- Work Packages
- Development Tasks
- WP-1 (EID-5): Cabin HMI development (In
- PRO-2 (EID-6070): Safety Architectural Guideli
  - PRO-2.1 (EID-7932): SIMPLAR Constraints (
    - REQ-288 (EID-7997): SIMPLAR Constrai
    - WP-77 (EID-7933): Apply hardware arch
    - SPC-83 (EID-7935): Apply hardware arc
    - WP-105 (EID-7940): Achieve limited pro
    - REQ-101 (EID-7941): Achieve limited pr
    - SPC-266 (EID-7942): Achieve limited pr
    - WP-215 (EID-7947): Control random ha
    - REQ-315 (EID-7948): Control random h
    - SPC-260 (EID-7949): Control random h
    - WP-206 (EID-7954): Manage avoidance
    - REQ-130 (EID-7955): Manage avoidanc
    - SPC-86 (EID-7956): Manage avoidance
    - WP-171 (EID-7961): Control systematic

1 of 96 rows

ISO 13849
Classification of the safety-related parts of a control system in respect of their resistance to faults and their subsequent behaviour in the fault condition, and which is achieved by the structural arrangement of the parts, fault detection and/or by their reliability

ISO DIS 25119
Classification of the safety related parts of a control system in respect of its resistance to faults and its subsequent behaviour in the fault condition, and which is achieved by the structural arrangement of the parts and/or by their reliability

FLAME
Classification of the safety-related elements of a control system in respect of its resistance to faults and its subsequent behaviour in the fault condition. A category is achieved by the structural arrangement of the elements and/or by their reliability. [Note: The concept of categories is only used in the functional safety standard ISO 13849 and ISO 25119.]

Work Package: WP-77 (EID-7933): Apply hardware architectural constraints

Chang

Deadline: Mon Mar 12 2012 00:00:00 GMT+0100 (Romance

Description:

≥ 99 %	SIL3	SIL3 (see Note 2)	SIL3 (see Note 2)

NOTE 1 A hardware fault tolerance of N means that N+1 faults could cause a loss of the safety-related control function.

NOTE 2 A SIL 4 claim limit is not considered in this standard. For SIL 4 see IEC 61508-1.

where fault exclusions have been applied to faults that could lead to a

h subsystem implementing a function block of an SRCF. A subsystem that comprises only uirements of IEC-62061:2005-table5, in particular, for such a subsystem that has a SFF of greater than 99% shall be achieved by a safety-related diagnostic function(s). fe failure fraction of less than 60% and zero hardware fault tolerance, that use well-tried 6 Category 1 PLC shall be considered to achieve a SILCL of SIL1.

)-13849-1:1999 and validated according to ISO-13849-2:2003, the relationship given in text of architectural constraints alone.

	ult tolerance	SFF	Maximum SIL claim limit according to architectural constraints
med that subsystems with the stated have the characteristics given below			
1	0	< 60 %	See Note 1

Attachments

26262-dis-5-tableE1.jpg	23022 B
62061-table6.jpg	103329 B
61508-2-table3.jpg	45765 B
61508-2-table2.jpg	44282 B
62061-table5.jpg	61705 B
13849-1-fig5-relationshipcategories.jpg	58010 B
13849-1-fig10-designatedarchitecturecat2.jpg	32866 B

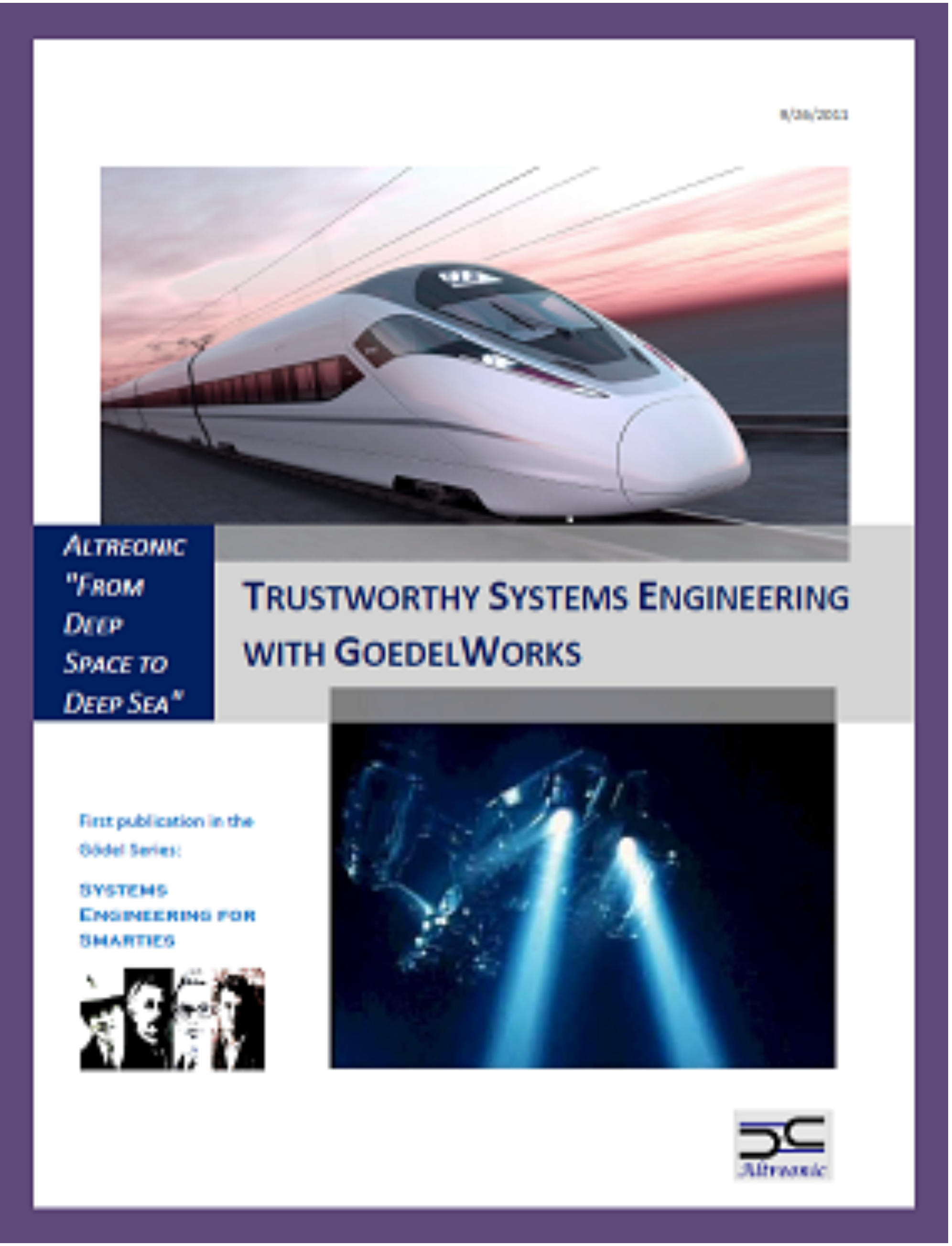
Association Links

Dependent Entities

Filter...

Preceding Entities

Filter...



8/26/2013

ALTREONIC "FROM DEEP SPACE TO DEEP SEA"

TRUSTWORTHY SYSTEMS ENGINEERING WITH GOEDELWORKS

First publication in the Gödel Series:

SYSTEMS ENGINEERING FOR SMARTIES

Altreonic

More info at www.altreonic.com

http://www.altreonic.com/sites/default/files/Systems%20Engineering%20with%20GoedelWorks.pdf

Business model

- Binary license (RTOS) or SaaS (GoedelWorks)
- Open Technology License:
  - Get all the technology + source + documents
  - Rebrand/resell/….
  - Often in conjunction with customer specific developments
- Qualification Package: as a GoedelWorks
- Customer specific engineering:
  - Porting code to new hardware
  - Formal development
  - Training

Contact:

www.altreonic.com

eric.verhulst (@) altreonic.com, CEO/CTO

Thanks for your attention