



Overview of the Architectural Style of GENESYS

H. Kopetz
February 2009



- ◆ Introduction
- ◆ What are the *Characteristics* of GENESYS?
- ◆ The GENESYS Component Model
- ◆ Hierarchy of Services
- ◆ Integration into Cyberspace
- ◆ Benefits of GENESYS
- ◆ Conclusion



Embedded Systems

3

Embedded Systems enable the real-time computer control of physical devices and systems, ranging from mobile phones, to television sets, to automotive engines and to industrial robots (to take a few examples) in order to achieve an ***unprecedented level of utility and performance***.

Embedded systems are also called *Cyber-Physical Systems* (CPS) to denote the emphasis and the close synergetic interactions of a real-time information processing subsystem (the *Cyber System*) with a physical device or subsystem that is to be controlled (the *Physical System*).

© H. Kopetz 2/5/2009



History of Embedded Systems

4

Driven by the dramatically increasing performance/price ratio of the semiconductor industry, many mechanical, hydraulic or electronic-analogue control systems have been replaced by digital micro-electronics over the past *twenty years*:

- ◆ *Bottom-up development* in the different application domains (automotive, industry, multimedia).
- ◆ *Domain-specific* protocols, standards and certification rules (e.g., CAN, Profibus, Firewire, etc.).
- ◆ *Need to integrate* requires a consolidation.

© H. Kopetz 2/5/2009



Semiconductor Industry Trends

5

- ◆ According to the ITRS 2007 p.84, the marginal production cost of one million bits of DRAM (packaged) was about 1 cent in 2007-- going to decrease to about .06 cent in 2015. This means that today about **10 billion bits** can be produced for the **cost of one engineering hour**.
- ◆ Parallel with this production-cost decrease goes a **dramatic increase in the fixed cost of chip manufacturing**, such as cost of equipment, design, mask cost, test-cost etc.
- ◆ An ever larger number of chips of a given design must be produced to recover the fixed cost.
- ◆ The time-honored trend that the failure rate per transistor decreases as fast or faster than the increase in the number of transistors per chip will not continue in the near future due to the **growing susceptibility of the ever smaller micro-devices to internal and external disturbances** (e.g., ambient cosmic radiation).

© H. Kopetz 2/5/2009



Technology Trends that Force a Consolidation

6

- ◆ Economies of scale of the semiconductor industry--an SoC cannot support *many* different protocols in hardware
- ◆ Interoperability of systems from different domains-- Integration of systems into the *Internet-of-Things*
- ◆ Pressing need to reduce the number of nodes and cables-- need for multi-criticality nodes
- ◆ *Security, robustness, and diagnosis* are cross-domain issues.
- ◆ Education and limited human resources.
- ◆ Investments in software and tools.

© H. Kopetz 2/5/2009



The European Response: *ARTEMIS*

7

- In order to meet these challenges, the *European ES Industry* in cooperation with the EU and the national authorities have formed the European technology platform ARTEMIS with the intent to improve the world-wide competitiveness of the European ES industry by developing a cross-domain embedded system architecture.
- A *system architecture* is a framework for the construction of a system that **constrains** an implementation in such a way that the ensuing system is understandable, maintainable, extensible, and can be built cost-effectively.

© H. Kopetz 2/5/2009



Requirements for the ARTEMIS Architecture

8

In a two year effort, following requirements have been identified for the cross-domain embedded system architecture by an ARTEMIS expert group:

- ◆ *Composability*
- ◆ *Networking and Security*
- ◆ *Robustness*
- ◆ *Diagnosis and Maintenance*
- ◆ *Integrated Resource Management*
- ◆ *Evolvability*
- ◆ *Self Organization*

Detailed requirements document at the ARTEMIS website
https://www.artemis-association.org/downloads/RAPPORT_RDA.pdf

© H. Kopetz 2/5/2009



GENESYS--Driven by ARTEMIS Requirements⁹

GENESYS is an FP 7 project that is developing an architectural framework for the design and implementation of cross-domain embedded systems that meets the ARTEMIS requirements.

© H. Kopetz 2/5/2009



What are *Characteristics* of GENESYS?

10

The following properties are characteristic for the *cross-domain architectural style* of GENESYS:

- ◆ Strict Component Orientation
- ◆ Openness
- ◆ Different Integration Levels
- ◆ Hierarchy of Services
- ◆ Deterministic Core
- ◆ Standard Internet Integration

© H. Kopetz 2/5/2009



Complexity Management in GENESYS

11

The architectural style of GENESYS deploys the following *simplification strategies* to reduce the complexity of a design:

- ◆ **Partitioning**: The partitioning of a system into nearly autonomous subsystems (components).--*Physical Structure*
- ◆ **Abstraction**: The introduction of abstraction layers whereby only the relevant properties of a lower layer are exposed to the upper layer--*Structure and Behavior*
- ◆ **Segmentation**: The *temporal decomposition* of complex behavior into small parts that can be processed sequentially (“step-by-step”)--*determinism* helps

© H. Kopetz 2/5/2009



What is a GENESYS Component?

12

A GENESYS *component* is a

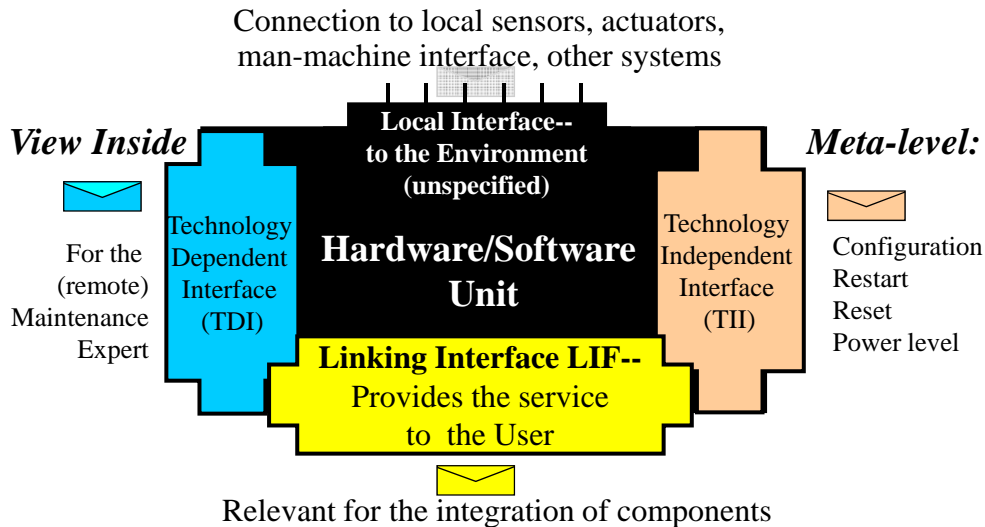
- ◆ **Hardware/software unit** that accepts input messages, provides a useful service, maintains internal state, and produces after some *elapsed time* output messages containing the results. It is aware of the progression of *physical time*.
- ◆ **Unit of abstraction**, the behavior of which is captured in a *high-level concept* that is used to capture the services of a subsystem.
- ◆ **Fault-Containment-Unit (FCU)** that maintains the abstraction in case of fault occurrence and contains the immediate effects of a fault (a fault can propagate from a faulty component to a component that has not been affected by the fault only by erroneous messages).
- ◆ **Unit of restart, replication and reconfiguration** in order to enable the implementation of robustness and fault-tolerance.

© H. Kopetz 2/5/2009



The Interfaces of a *GENESYS* Component (i)

13



© H. Kopetz 2/5/2009



The Interfaces of a *GENESYS* Component (ii)

14

- ◆ **Linking Interface (LIF):** Provides the operational services of the Component, *must be precisely specified in the domains of value and time*, relevant for the User and for Composability
- ◆ **Technology Independent Interface (TII--Meta Level):** Focus on *Non-functional Properties*, such as configuration, naming, download of software, reset of a computer, restart, resource management, etc.
- ◆ **Technology Dependent Interface (TDI--Inside Look):** Relevant for an expert who has deep knowledge about the component implementation
- ◆ **Local Interfaces:** Connects a component to the outside world (Process I/O, other systems), semantics are part of the LIF specification.

Depending on the view point, an *IP-core* of an MPSoc, a *chip*, a *device* or a *subsystem* can be seen as a component.

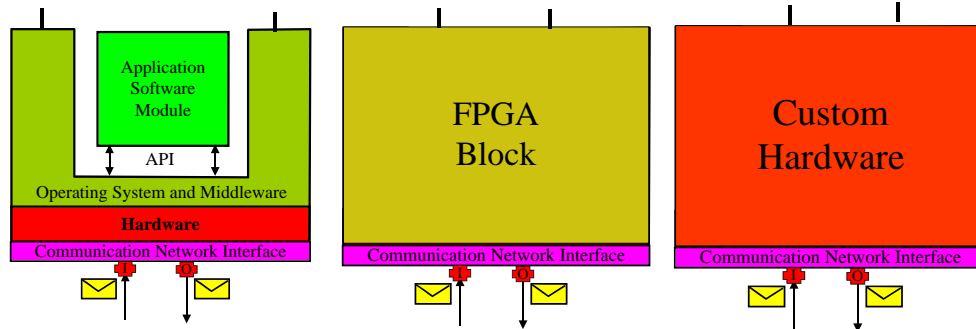
© H. Kopetz 2/5/2009



Openness: *Soft versus Hard Components*

15

Local Interfaces--Open Components



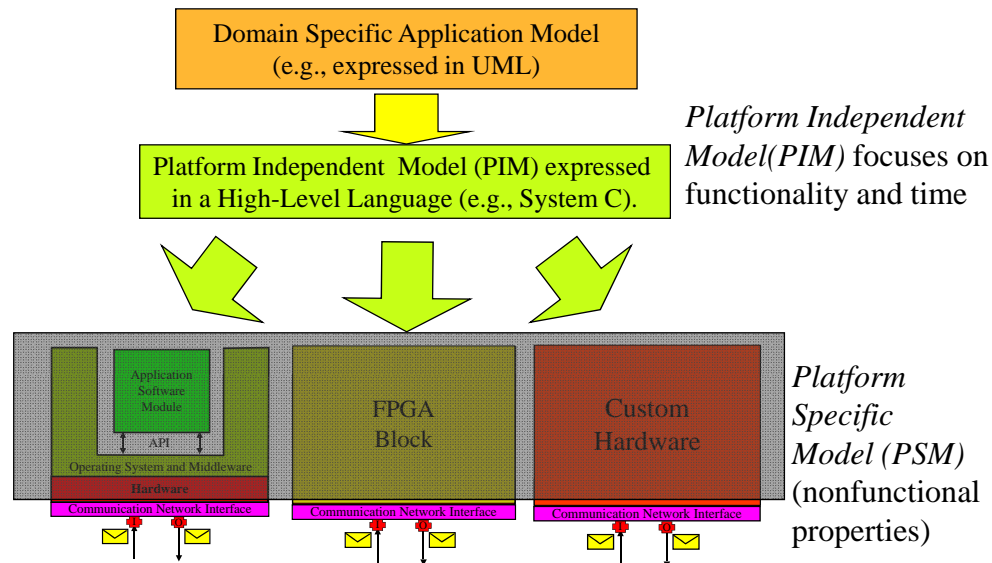
The Linking Interface (LIF) of all three different component implementations should have the same *syntax*, *timing* and *semantics*. For a user, it should not be discernible which type of component is behind the LIF (*Technology Agnosticism*).

© H. Kopetz 2/5/2009

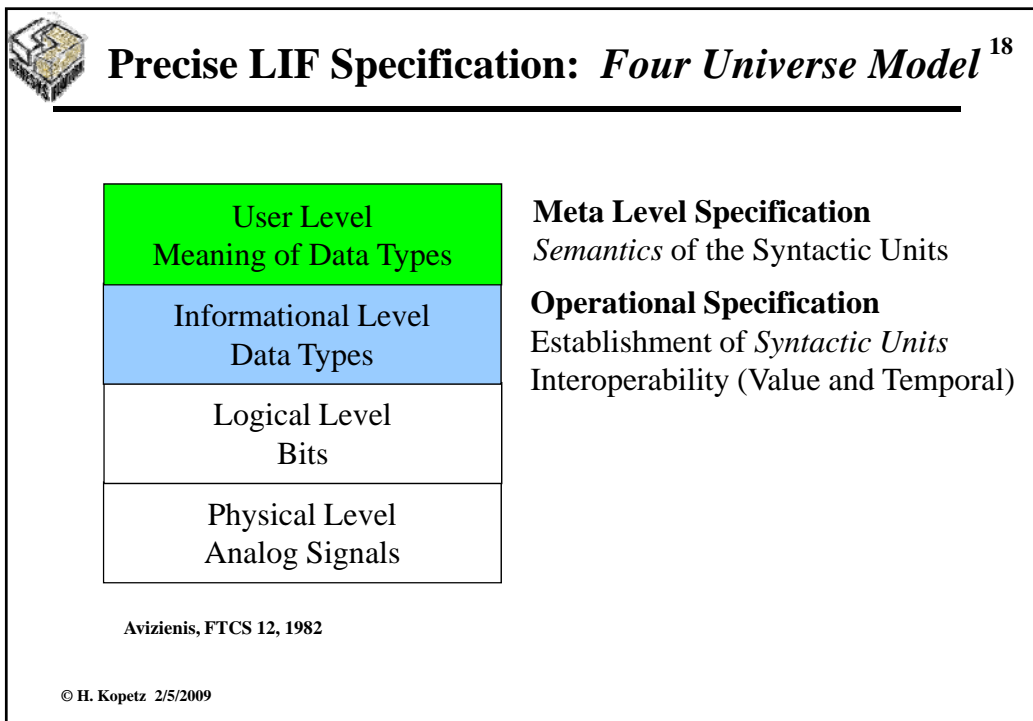
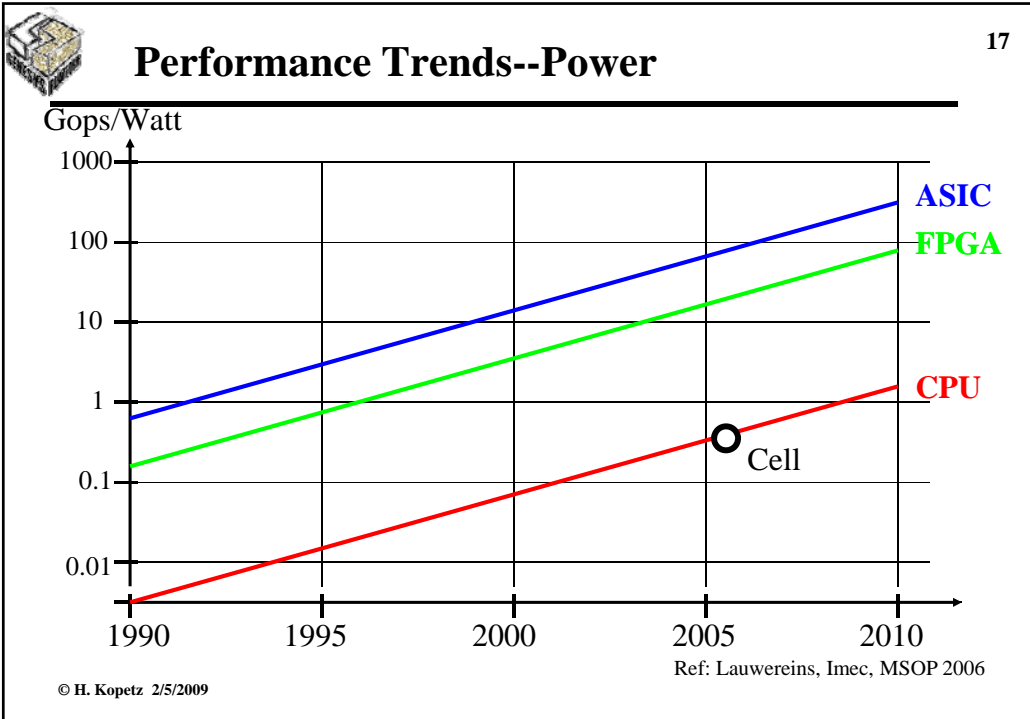


Abstraction: *Model Driven Design*

16



© H. Kopetz 2/5/2009





Linking Interface (LIF) Specification

19

Operational Specification-- *formal*:

- ◆ **Operational Input Interface Specification**
 - Syntactic Specification (e.g. by IDL)
 - Temporal Specification (receive instants, data accuracy)
 - Input Assertion
- ◆ **Operational Output Interface Specification**
 - Syntactic Specification (e.g. by IDL)
 - Temporal Specification (send instants, data accuracy)
 - Output Assertion
- ◆ **Interface State:** contents of *declared interface data structure* at observation instant

Meta-level Specification:

- ◆ Meaning of the data elements: Means-and-ends model
- ◆ Interface Model: Data transformation of Component **including model of the connected environment.**

© H. Kopetz 2/5/2009



Integration: Principles of Composability

20

- (1) **Independent Development of the Components** (Architecture)
The interfaces of the components must be *precisely specified* in the value domain and in the *temporal domain* in order that the component systems can be developed in isolation.
- (2) **Stability of Prior Services** (Component Implementation)
The prior services of the components must be maintained after the integration and should not fail if a partner fails.
- (3) **Non-Interfering Interactions** (Communication System)
The communication system transporting the messages must meet the given temporal requirements under all specified operating conditions.
- (4) **Preservation of the Component Abstraction in the case of failures** (Architecture) and provision of a communication system with error containment.

© H. Kopetz 2/5/2009



Communication in GENESYS

21

At the core GENESYS components communicate by
***unidirectional deterministic multi-cast* messages**

- ◆ ***Uni-directionality*** is required in order to decouple communication from computation (*fate-sharing* principle).
- ◆ ***Determinism*** is required to
 - establish timeliness
 - simplify the reasoning about the behavior (*modus ponens*)
 - simplify testing (repeatable test cases)
 - be able to implement active replication (TMR)
 - support the certification
- ◆ ***Multi-cast*** is required to support the independent observation of the component behavior

© H. Kopetz 2/5/2009



GENESYS Communication Services

22

GENESYS introduces three communication services

- ◆ ***Sporadic Messages***
 - characterized by two queues, one at the sender site and one at the receiver site
 - Exactly once semantics (*wine*)
 - Normally best effort timing
- ◆ ***Periodic Messages***
 - No queues, non-consuming read, update in place (*milk*)
 - Temporal guarantees
- ◆ ***Real-time data streams***
 - Guaranteed bandwidth and timing
 - Queues with watermark management

***Openness:* Any communication protocol (wire-bound or wireless) that provides these services can be used in GENESYS**

© H. Kopetz 2/5/2009

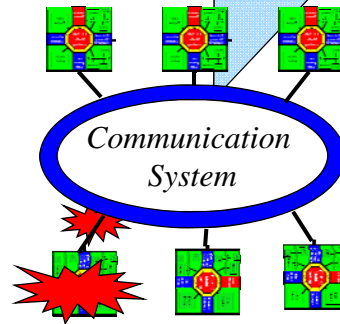


Error Containment by the Communication System²³

It is *impossible* to maintain the communication among the correct components of a RT-cluster if the temporal errors caused by a faulty component *are not contained*.

Error containment of an arbitrary temporal node failure requires that the Communication System is a self-contained FCU that has *application specific temporal information* about the allowed behavior of the nodes.

Temporal Error Containment Boundary



Babbling idiot

© H. Kopetz 2/5/2009



Partitioning: Integration Levels

24

In GENESYS we introduce three integration levels:

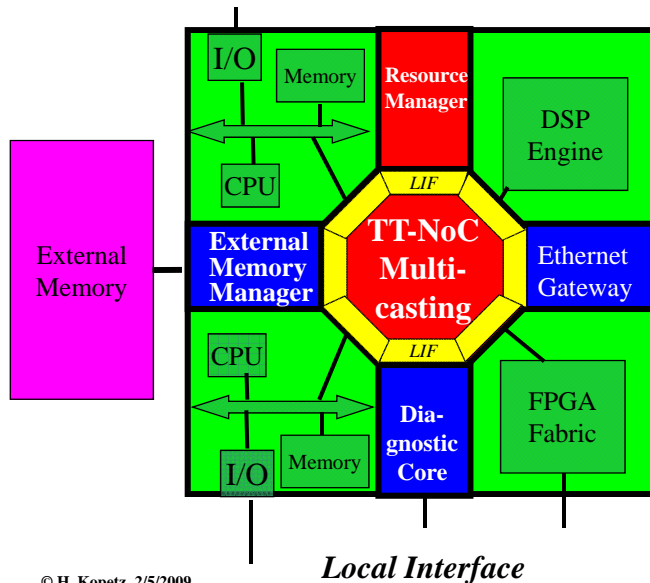
- ◆ **Chip Level:** the components are IP-cores, interconnected by a NoC (network on Chip) to form a Chip
- ◆ **Device Level:** the component are chips interconnected by an inter-chip communication system to form a Device. A device can be an addressable entity in the Internet and can have an IP-Address (as well as a chip, if desired).
- ◆ **System Level:** The components are devices that are interconnected by a wire-bound or wireless communication service:
 - *Closed Systems:* System structure is *static*.
 - *Open Systems:* System structure is *dynamic*, i.e., devices can come and go

© H. Kopetz 2/5/2009



Chip-Level: A GENESYS System-on-a-Chip

25



Standard components (IP-cores):

- ◆ External Memory Manager
- ◆ Resource Manager
- ◆ Diagnostic Core
- ◆ Ethernet Gateway

Application components

- ◆ Two CPUs with application software
- ◆ DSP Engine
- ◆ FPGA Fabric

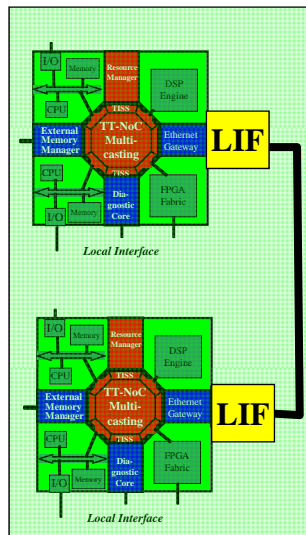
© H. Kopetz 2/5/2009

Local Interface



Device-Level: Integration of Chips

26



Chips are linked by *intra-device-level LIFs* to form a device.

Viewed from the *intra-chip level*, the *intra-device level LIF* is a *local interface (and vice versa)*.

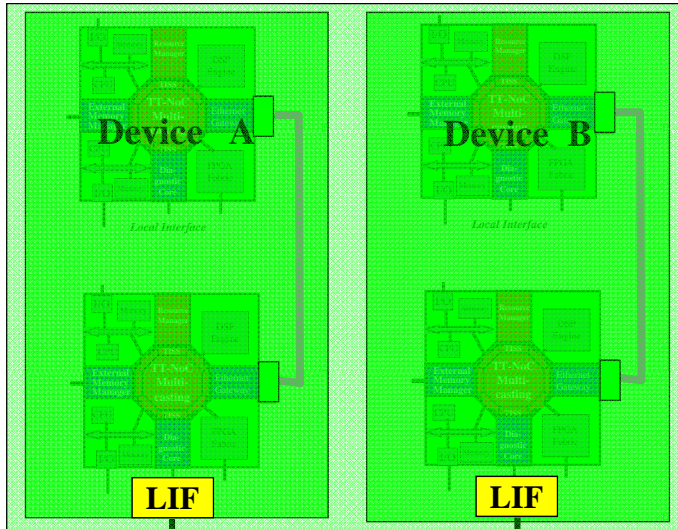
The intra-device level LIF carries its own LIF Specification that comprises all subsystems that are connected to this device.

Openness: The open LIF specification makes it possible to integrate legacy systems.

© H. Kopetz 2/5/2009



System Level: *Integration of Devices (static or ad-hoc)*²⁷



Devices are linked by

System Level LIFs

We distinguish between

- ◆ Open
- ◆ Closed

Systems.

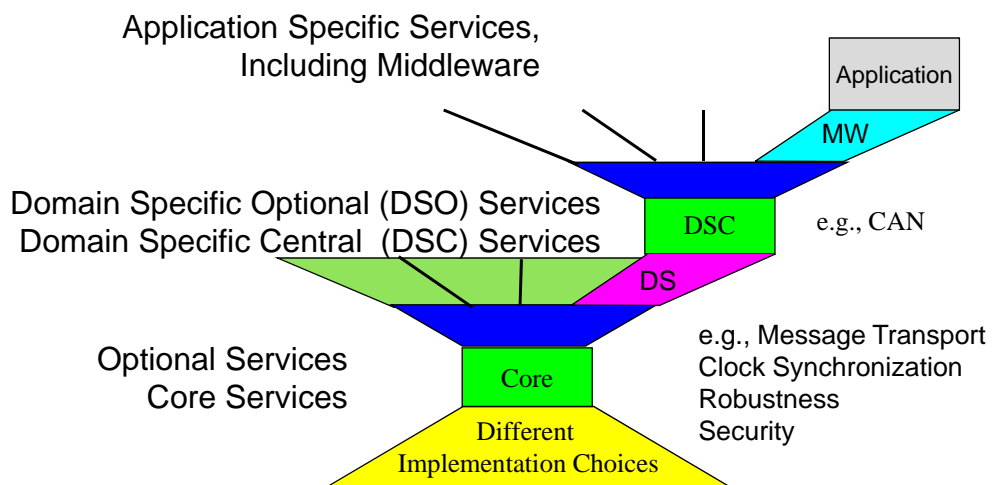
© H. Kopetz 2/5/2009

Wirebound or Wireless



Abstraction: *Hierarchy of GENESYS Services*

28



© H. Kopetz 2/5/2009



Typical Optional Services

29

- ◆ Integrated Resource Management
- ◆ Diagnostic and Robustness Services
- ◆ Security Services
- ◆ Man-Machine Interface via a *Web Browser*
- ◆ Internet Connectivity-- *Internet of Things*
- ◆ Voting Service for Triple Modular Redundancy
- ◆ Overlay Network Service for domain-specific protocols (e.g., CAN, Most, ETHERNET)
- ◆

© H. Kopetz 2/5/2009



Resource Management

30

- ◆ The GENESYS Architecture supports *local* (component-level) and *global* resource management, considering performance, power and energy.
- ◆ The *TII (Technology Dependent Interface)* can be used to control the performance parameters of components by a resource management components.
- ◆ The component-structure of GENESYS simplifies the implementation of *power-gating*.

© H. Kopetz 2/5/2009



Robustness

31

Robustness is concerned with the provision of an acceptable level of service in the face of all types of disturbances (e.g., physical faults, design faults, specification change, incorrect operation):

- ◆ A diagnostic component monitors the *ground states* of components at the reintegration instant to detect state corruption.
- ◆ In case of a state corruption, the diagnostic component performs a state estimation for the next reintegration instant.
- ◆ It resets the component at the next reintegration instant and sends the estimated state to the component via the TII.

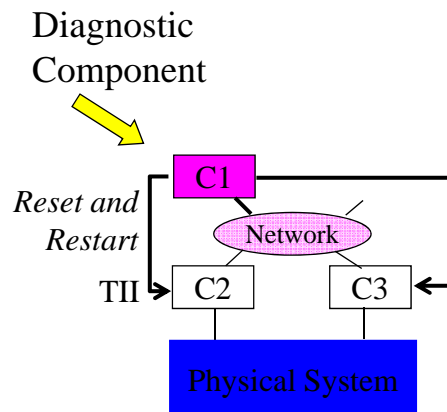
© H. Kopetz 2/5/2009



Diagnostic Component-- Standardized Component³²

A diagnostic component (C1) observes the behavior of the other components:

- ◆ Is an autonomous FCU (Fault Containment Unit)
- ◆ Gets copies of relevant messages (via the multicast mechanism)
- ◆ Reasons about the Health State of the Components
- ◆ *May have the authority to reset and restart a component via the TII*



© H. Kopetz 2/5/2009



Security

33

GENESYS provides the following security services

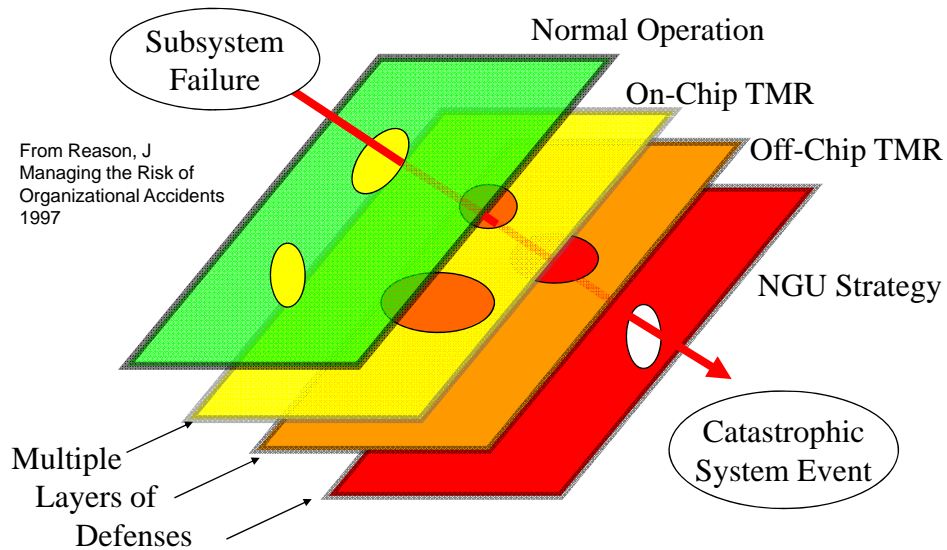
- ◆ The security model covers *external* and *internal* (insider) attacks and download authorization.
- ◆ At the chip level, every SoC has unique *tamper-proof credentials* as a basis for the implementation of security services.
- ◆ Cross-domain security services, such as authentication, intrusion detection, confidentiality are homogenized and follow the established Internet standards
- ◆ A *dedicated cross-domain gateway component* contains a firewall w.r.t. traffic from the Internet

© H. Kopetz 2/5/2009



Approach to Safety: The Swiss-Cheese Model

34



© H. Kopetz 2/5/2009



Three Levels of Fault Mitigation

35

- (i) Normal Operation
- (ii) Swift Component Recovery or On-Chip TMR: to handle transient faults within a chip
- (iii) Off-Chip TMR: to handle a transient and permanent fault of a total chip.
- (iv) NGU (Never-Give-Up) Strategy: to handle multiple correlated transient faults.

© H. Kopetz 2/5/2009



What is Needed to Implement TMR?

36

What architectural services are needed to implement Triple Modular Redundancy (TMR) at the architecture level?

- ◆ Replica Deterministic (*which includes timely*) Operation of the three Lanes
- ◆ Provision of an Independent Fault-Containment Region for each one of the Replicas and the Communication System.
- ◆ Replicated Independent Communication Channels--Temporal Error Detection by the Communication System
- ◆ Synchronization Infrastructure
- ◆ Predictable Multicast Communication
- ◆ Support for Voting

GENESYS provides all these services, either as core services or as optional services.

© H. Kopetz 2/5/2009

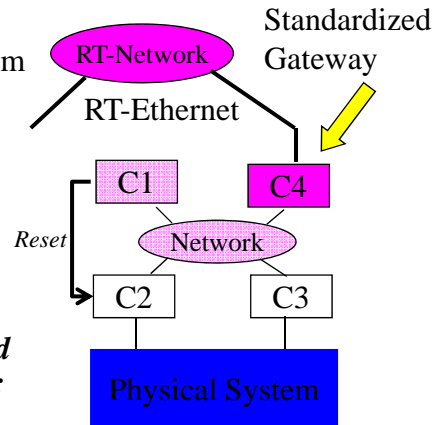


Component Integration-- Openness

37

A standardized gateway component (C4) provides for the integration of a subsystem with another subsystem by a *real-time Intranet* or the *Internet*:

- ◆ Every subsystem has an IP-Address
- ◆ Contains a standardized Web Server
- ◆ Man-machine communication via a remote Browser
- ◆ **Heterogenous subsystems are linked by an open standardized technology: there is no lock-in to a single manufacturer.**



© H. Kopetz 2/5/2009

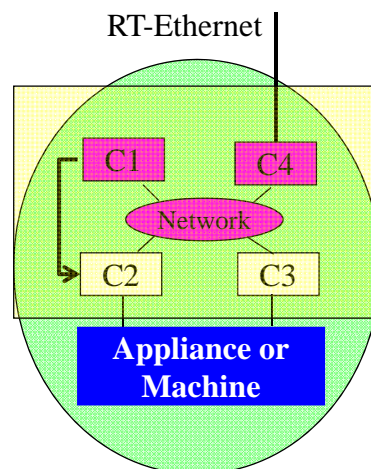


Generic Island of Automation (IoA)

38

We can form a standardized Island of Automation (IoA) of an *appliance* or a *machine* by connecting standardized GENESYS components. An IoA

- ◆ Has an IP-Address
- ◆ Contains a standardized Web Server
- ◆ Man-machine communication via a remote Browser
- ◆ Can be connected to the Internet by an *Internet Gateway*
- ◆ **The Cyber Part of the IoA could be implemented on a single GENESYS chip**

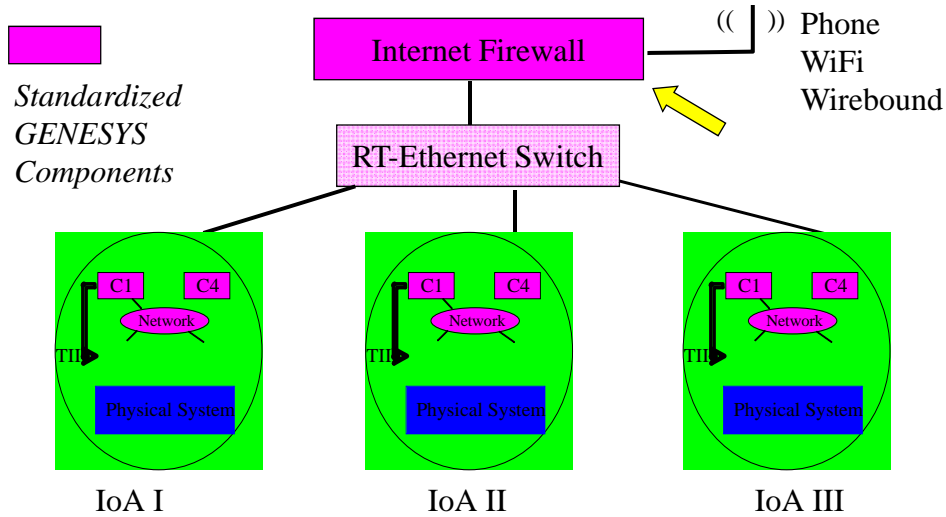


© H. Kopetz 2/5/2009



Connection of IoAs to the Internet

39



© H. Kopetz 2/5/2009



An Internet Firewall links IoAs to Cyberspace

40

The *Internet Firewall* is a *standardized* component that

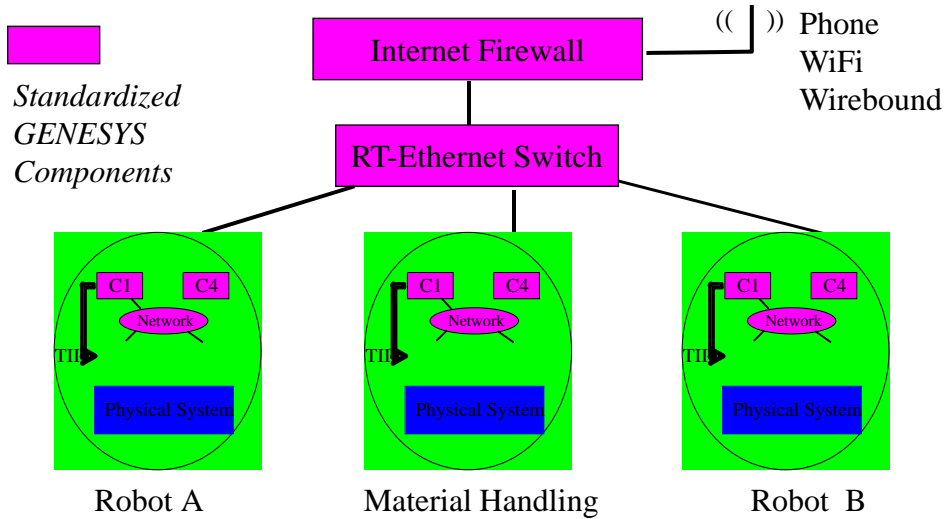
- ◆ Establishes the link to the external world, by
 - Phone or
 - WiFi or
 - Wirebound
- ◆ Contains a security firewall (*peripheral security*) and acts as the local Certification Authority for the *IoAs* at this site (to protect the *IoA* from *insider attacks*).
- ◆ Provides the global time to all connected *IoAs*, e.g., by reference to GPS-time
- ◆ Supports secure remote access, e.g. for man-machine interaction, remote diagnosis or software download

© H. Kopetz 2/5/2009



Example: *Industrial Automation*

41

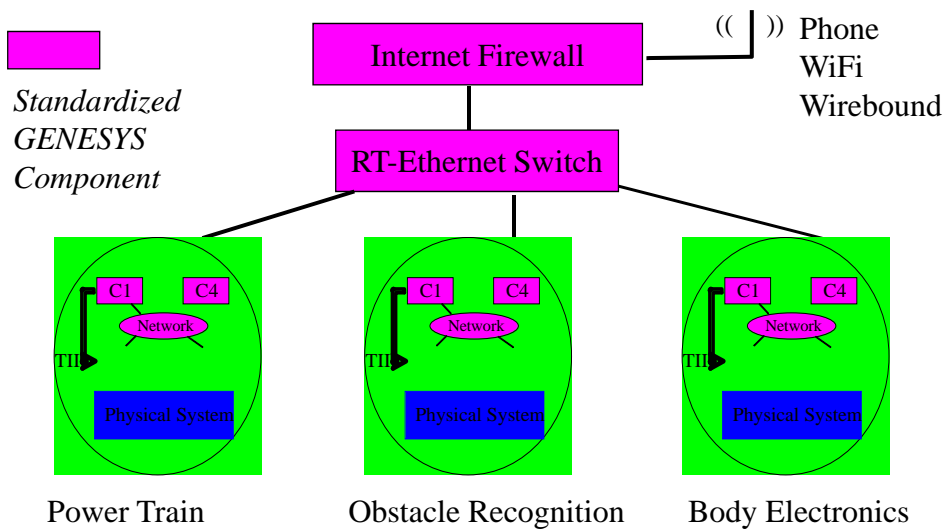


© H. Kopetz 2/5/2009



Example: *Automotive Electronics*

42

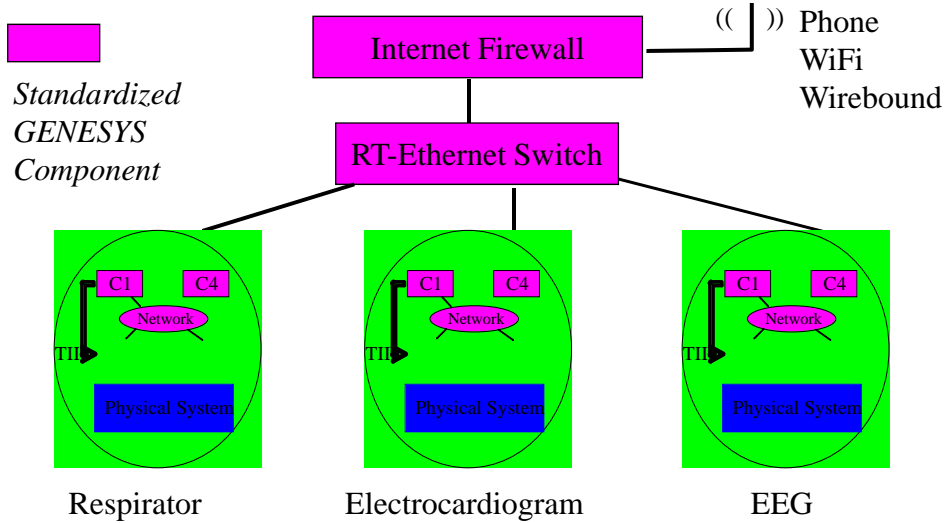


© H. Kopetz 2/5/2009



Example: *Medical Intensive Care Unit*

43

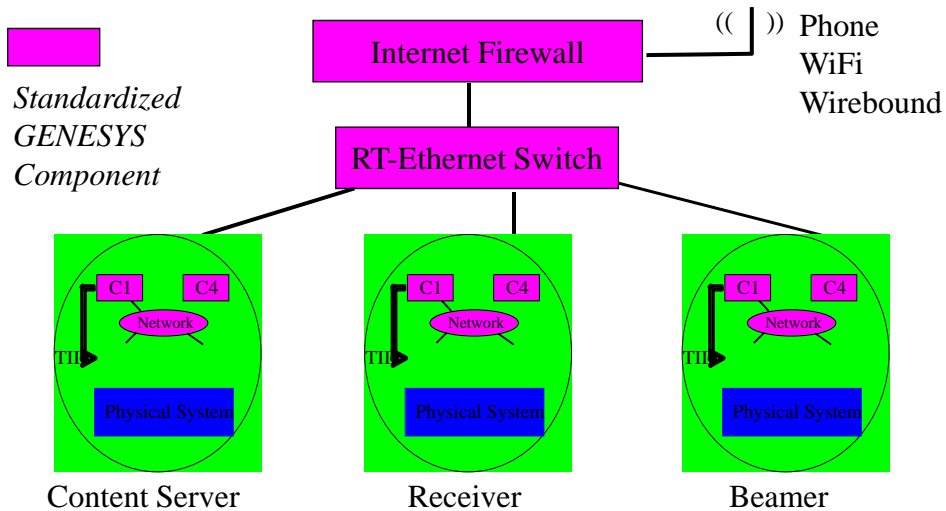


© H. Kopetz 2/5/2009



Example: *Multimedia System*

44



© H. Kopetz 2/5/2009



What are the *Benefits* of GENESYS?

45

What are benefits of the characteristic properties of the *cross-domain* architecture GENESYS?

- ◆ Strict Component Orientation
- ◆ Openness
- ◆ Hierarchy of Services
- ◆ Deterministic Core
- ◆ Standard Internet Integration

© H. Kopetz 2/5/2009



Benefits of a *Strict Component Orientation*

46

- ◆ **Simplification** of the understanding and consequent a *Reduction of the Complexity* of large systems by lifting the Abstraction Level
- ◆ **Cross-domain reuse** of components based on the precise specification of the Linking Interface (LIF)
- ◆ **Economies of scale**: Standard components can be manufactured in large quantities, taking advantage of the *economies of scale* of the semiconductor industry.
- ◆ **Robustness**: Individual components can be reset, restarted, replicated and reconfigured in order to mitigate the effects of transient and permanent component failures.
- ◆ **Reduction of Integration efforts** in static systems and in an dynamic *plug-and-play* environment.

© H. Kopetz 2/5/2009



Benefits of *Openness*

47

- ◆ **No lock-in:** *Heterogeneous components* from different suppliers can be integrated by standardized technology.
- ◆ **Technology obsolescence management:** The precise open interface specification (LIF) of components makes it possible to technologically update a component without any effect on the component environment.
- ◆ **Scalability:** Open systems are scalable and can evolve to meet requirements that we do not know about today.
- ◆ **Legacy Integration:** Precise open interfaces simplify the integration of legacy systems by application specific gateway components.
- ◆ **Cost reduction:** Standardized open Components (e.g., communication system, Internet Gateway, Firewalls) can be produced in large quantities

© H. Kopetz 2/5/2009



Benefits of the *Hierarchy of Services*

48

- ◆ **Complexity Management:** Hierarchic Decomposition is a well-accepted strategy to handle *complexity*. *No monolithic complex operating system*.
- ◆ **Reuse:** The small amount of generic core services can be implemented in hardware and ***tailored to be reused*** in many different application contexts.
- ◆ **Domain orientation:** *Domain specific central services* can be implemented on top of the core services by software (middleware) or hardware
- ◆ **Economy of Effort:** A domain expert is only required to understand the domain specific services

© H. Kopetz 2/5/2009



Benefits of a *Deterministic Core*

49

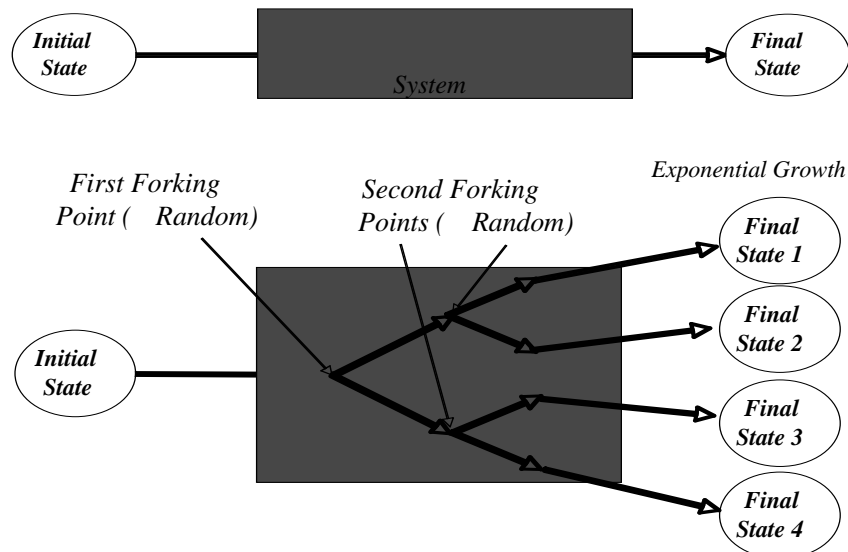
- ◆ **Timeliness:** Determinism implies *timeliness*. Timeliness is needed in many embedded applications.
- ◆ **Complexity Reduction:** The human mind is ill-equipped to handle non-deterministic behavior. *Determinism* supports *segmentation* by imposing a linear temporal structure on the behavior.
- ◆ **Testability:** It is much easier to test a system if the same input always produces the same result.
- ◆ **Certification:** Certification authorities prefer deterministic behavior of the safety-relevant services
- ◆ **Fault Masking:** Deterministic behavior is a prerequisite for Fault-masking (e.g., by triple modular redundancy, TMR) at the logical level.

© H. Kopetz 2/5/2009



Complexity: *Determinism vs. Non-Determinism*

50



© H. Kopetz 2/5/2009



Benefits of a *Standard Internet Integration*

51

- ◆ **Protected Remote Access:** to an Appliance or Machine for *remote operation* and *remote diagnosis*.
- ◆ **Reduction of Integration Effort:** Standard *RT-Ethernet* reduces project specific development effort and takes advantage of the economies of scale of the semiconductor technology.
- ◆ **Generic Man Machine Interaction:** by a standard Web Browser--reduction in the number of dedicated MMI devices.
- ◆ **Security:** Standard Internet Firewalls ensures security without undue development and integration efforts.
- ◆ **Evolvability:** Heterogeneous *IoAs* can communicate locally and remotely using Internet standards to evolve and realize emergent services.

© H. Kopetz 2/5/2009



Conclusion

52

- ◆ The cross domain Architectural Style of GENESYS supports a strict *component-based design style* and supports the straightforward composition of systems out of components.
- ◆ The *stable core-services* of GENESYS, which can be implemented cost-effectively in hardware, are the basis for the realization of *flexible* domain specific higher-level services.
- ◆ The *dependability services* of GENESYS will lead to a higher robustness (improved security, reliability, diagnosis).
- ◆ The *smooth integration of existing legacy systems* into the GENESYS architecture makes it possible to introduce GENESYS in small steps.
- ◆ The *wide adoption of GENESYS* will lead to significant *cost-reductions* and a *reduced time-to market* for embedded systems.

© H. Kopetz 2/5/2009