# Time-Triggered Architecture

H.Kopetz
TU Wien
January 2000

# Outline

- Introduction
- Composability
- The Time-Triggered Architecture
- The TTP Protocols
- CNI as Temporal Firewalls
- Conclusion

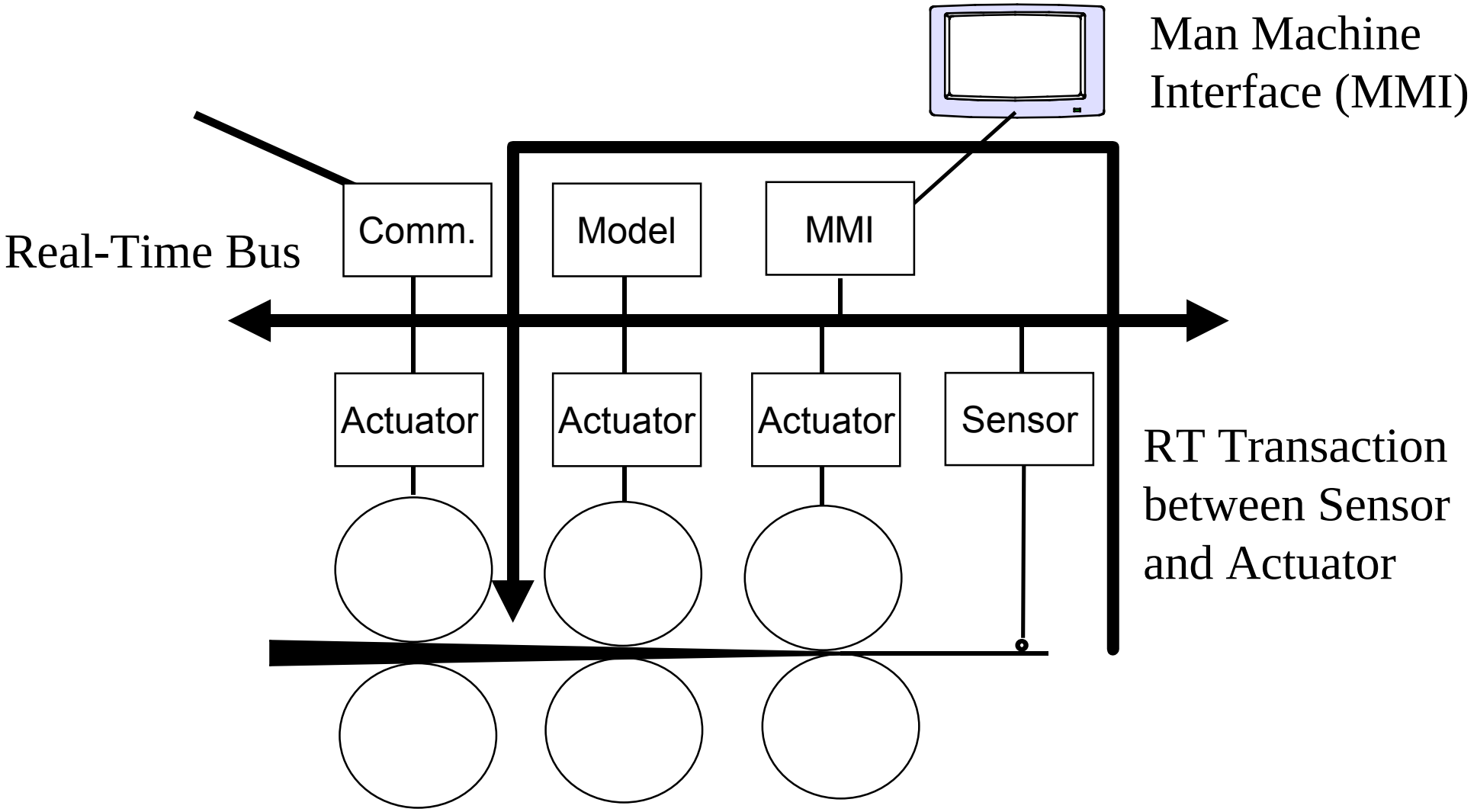
# Architecture Design *is* Interface Design

A good interface **within** a real-time system

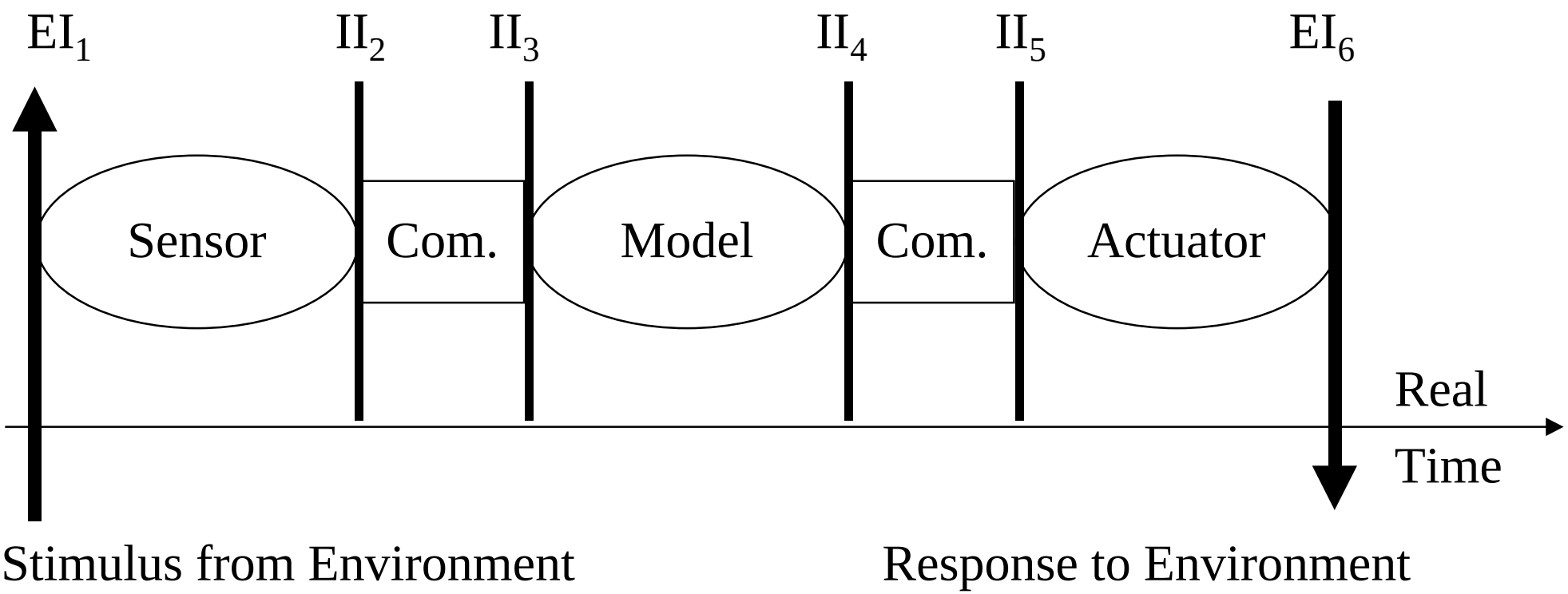
- is precisely specified in the value domain and in the time domain,
- provides the relevant abstractions of the interfacing subsystems and hides the irrelevant details,
- leads to minimal coupling between the interfacing subsystems,
- limits error propagation across the interface,

and thus introduces ***structure*** into an architecture.

# An Example: Rolling Mill



# Real-Time Transaction



# What is badly needed . .

is an architecture that supports a two level design methodology

- at the **System Integrator Level** the interactions between the subsystems are designed and the CNIs to the components are fully specified in *the value domain and in the temporal domain*.
- at the **Component Supplier Level** the components are designed and implemented, taking the precise CNI specifications as design constraints.

The composability property of the architecture ensures that the component properties (e.g., timeliness) are not invalidated by the system integration.

# System Integration

Given a set of subsystems that are integrated to form a system.

A subsystem is a self-contained system, including hardware, software, and its own autonomous control, that provides a specified service to its environment.

At the system level, we distinguish between two types of services:

- **Prior Services**: The sum of the services that are provided by the isolated subsystems prior to the integration.
- **Emerging Services**: New services that come into existence

# Composability

The prior service of each subsystems $S_i$ can be characterized by prior properties $P_{ik}$. Assume that a given prior property $P_{ik}$ of subsystems $S_i$ has been established at the subsystem level before the integration.

An architecture is said to be *composable* with respect to this property $P_{ik}$ if this property also holds at the system level after the integration.

Examples of such properties are:

Timeliness

Testability

**Composability ensures the stability of prior properties.**

# An Ideal Component is a Hardware/Software Unit

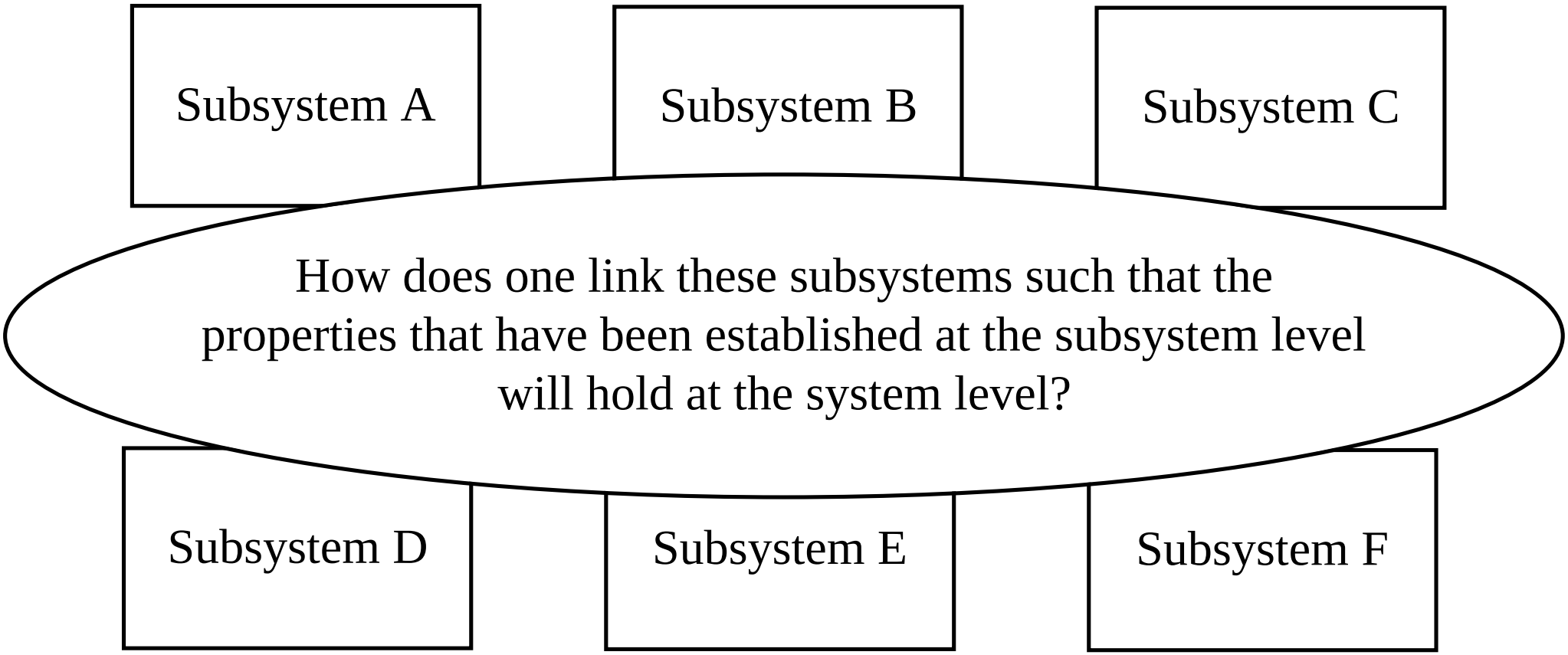


It provides its services across a stable standardized communication network interface to its environment *(temporal firewall*):

- The interface must be fully defined in the value domain and in the temporal domain.
- The user must know precisely, when using a component, what to provide and what he can expect (component reuse).
- Composability must be supported.
- Present integration technologiesSoftware components do not have a temporal behavior

# Composability: The Role of the Network

"The network is the only mechanism suitable to enforce and manage real-time operation of distributed systems" (Caro 1998) .



# Three Types of Component Interfaces

- **Real-Time (RS) Service Interface:** provides the time critical service to its users
- **Diagnostic and Management (DM) Interface:** Opens a communication channel to the internals of the component
- **Configuration Planning (CP) Interface:** Used to connect the component to other components in the system.

**From the point of view of composability and reuse, only the the RS interface is important.**

# The Four Principles of Composability

A composable architecture must support:

(i) Independent development of components

(ii) Stability of prior services

(iii) Constructive integration of components

(iv) Replica determinism

If anyone of these principles is violated, the architecture is not composable. It is of questionable value to talk about "partial composability".

# Composability and Reuse of Components

Composability and the effortless reuse of availabe components are highly intertwined:

- The precisely defined component interfaces of a composable architecture specify clearly what a user has to supply and what a user can expect from an existing component.
- The “stability of prior service” principle ensures that the functions of the existing component are not disturbed by the integration.
- The “constructive integration” principle ensures that the component integration is linear and not circular.

# Is the Time-Triggered Architecture a Solution?

The Time-Triggered Architecture (TTA) is an architecture for distributed real-time systems in high-dependability applications:

- There is a sparse global time of known precision in every component
- The time-triggered architecture decomposes a real-time system into clusters, Fault-Tolerant Units (FTU), and nodes.
- There are two types of nodes in the architecture, a fail-silent TTA-system node (TTA SOC) and a smart sensor node.
- FTUs are formed by a group of nodes that operate in replica determinism, such that the fail-silent failure of one node is masked.
- The Communication Network Interface (CNI) between a communication controller and the host computer is a strict data-sharing interface that is fully specified in the temporal domain and in the value domain and contains state messages.

# Event Message versus State Message

Event Messages are event triggered:

- contain *event information*
- queued and consumed
- external control outside the communication system n the software in the host computer of a node.

State Messages are time triggered:

- contains *state information*
- atomic update in place by single sender, not consumed on reading, many readers
- sent periodically, autonomous control within communication system

State messages are appropriate for control applications.

# Event-Triggered vs. Time-Triggered

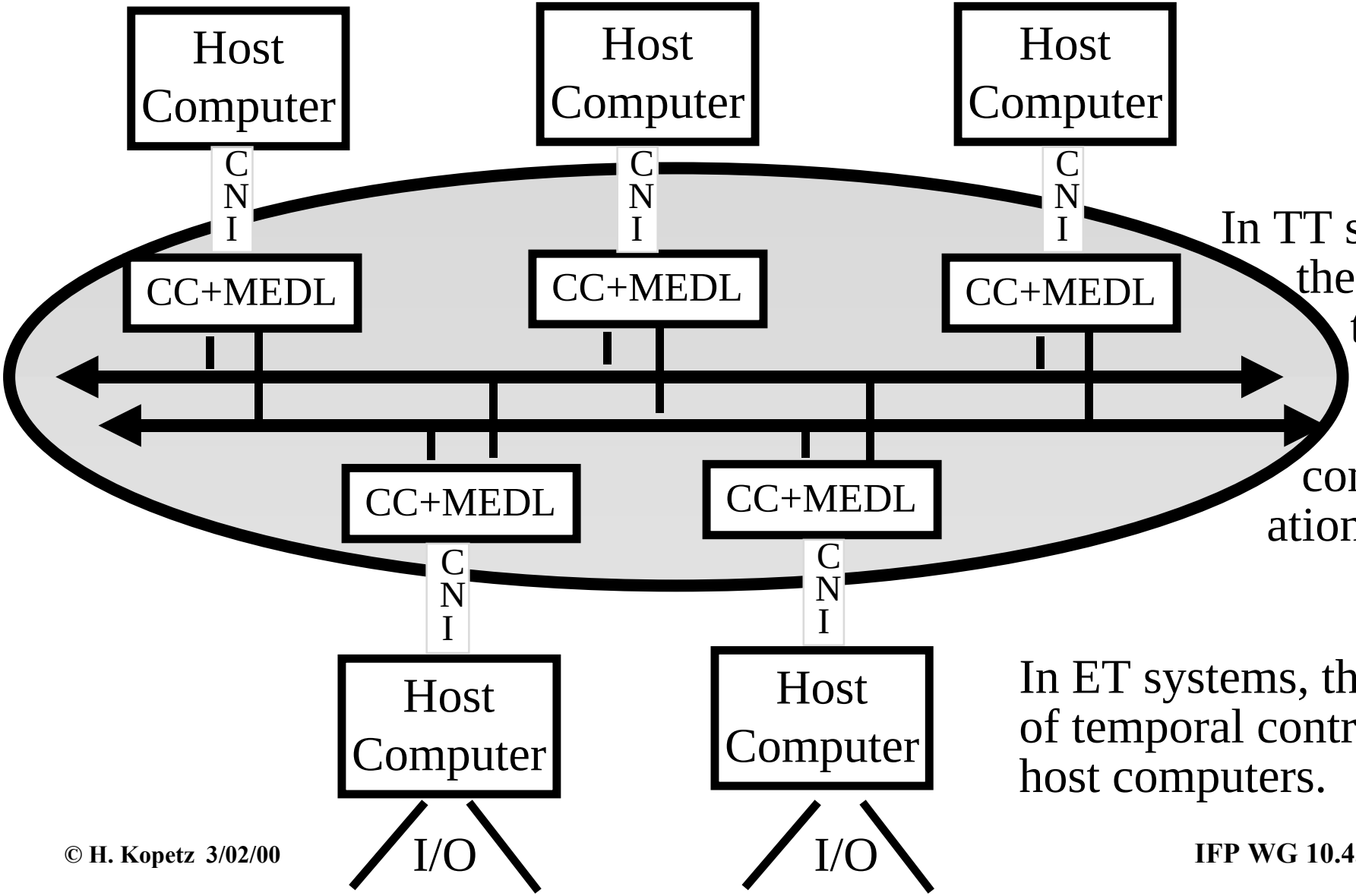
Event Triggered (e.g, CAN)

- Event messages
- Temporal control derived from the occurrence *unpredictable*
- Flexibility
- Large Jitter
- No precise temporal specification of interfaces
- Good for sporadic data
- Membership difficult
- Probabilistic Access

Time Triggered (e.g., TTP)

- State messages
- Temporal control derived from the progression of time *predictable*
- Interoperability
- Minimal Jitter.
- Precise temporal specification of interfaces
- Good for regular data
- Membership easy
- Replica Determinism

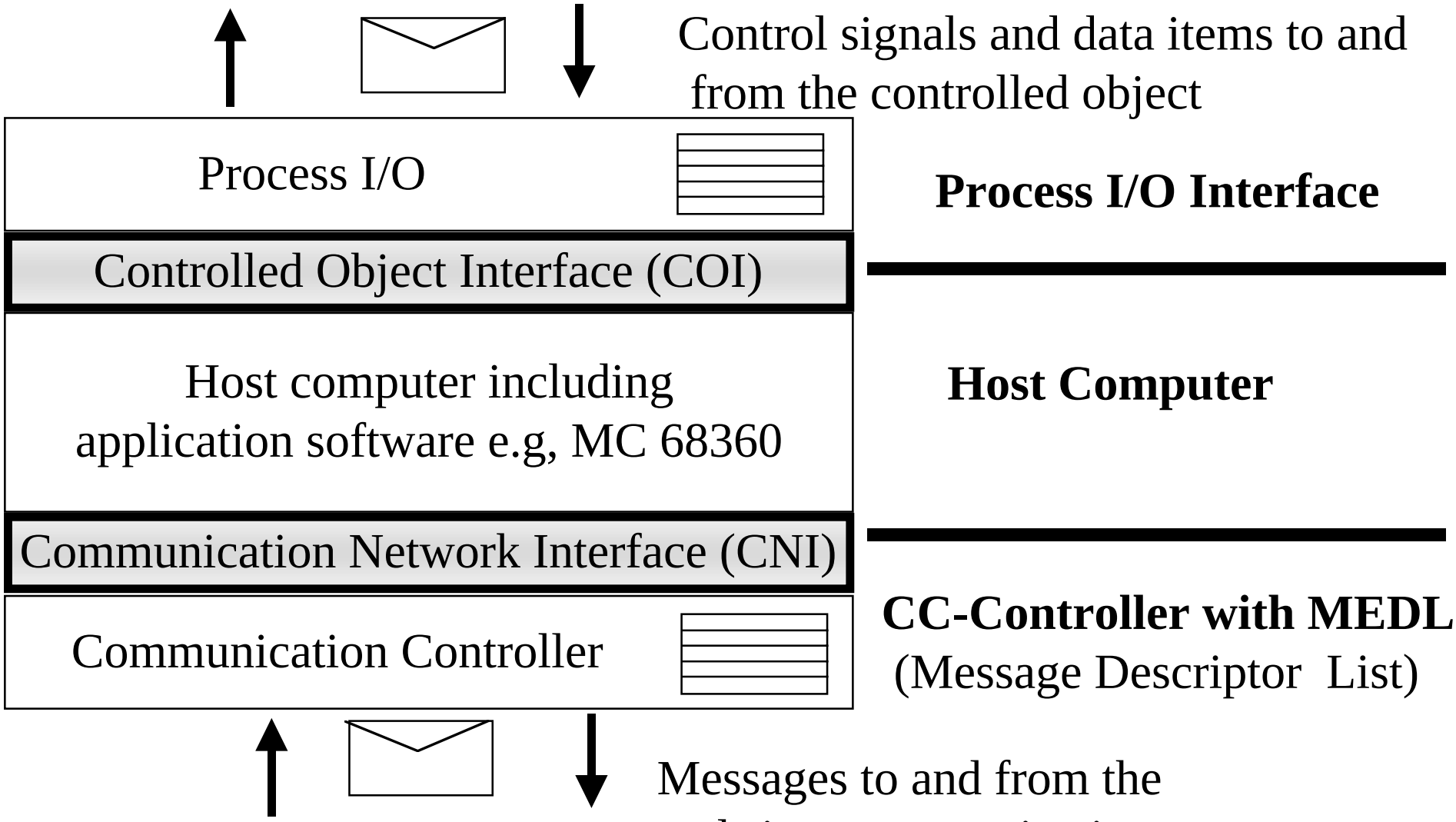
# Global Interactions versus Local Processing




In TT systems, the locus of temporal control is in the communic-ation system.

In ET systems, the locus of temporal control is in host computers.

# A Node consist of three Subsystems:



# Key Concept of the Time-Triggered Architecture:

The time-triggered architecture **temporal properties of the real-time data** in the Communication Network Interface (CNI):

- Every real-time data element has an associated validity interval e.g., the statement "*the traffic light is green*" is only valid for a limited period of time.
- The data elements that are the input to a real-time application task must be temporally valid at the time of use.
- The temporal validity of data can only be traced in a distributed real-time system if there exists a *global notion of time*.
- The TTA provides such a global time base.

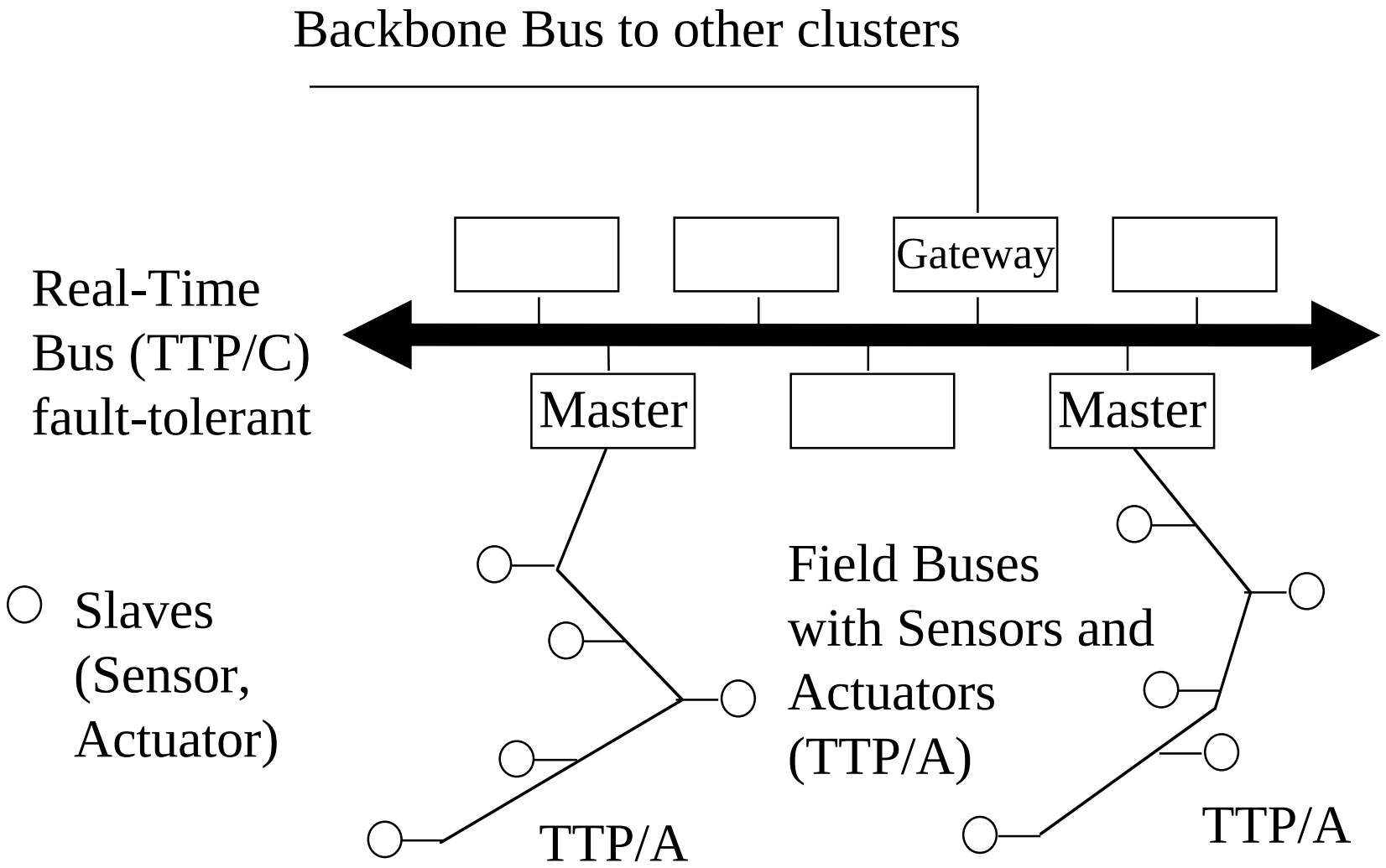
# Two Real-Time Networks in the TTA

The following two different real-time network types are needed in distributed control applications for economic reasons (from the technical point of view a single system network type would be sufficient):

- System Bus: The system bus connects the system nodes, the powerful SOCs. System networks must have distributed communication control and provide fault-tolerance.
- Field Bus: The field bus connects one or more system nodes to the smart sensors and actuators. The field bus can be a multi-master network where one or more powerful masters control a set of simple associated smart sensor nodes.

**The CNIs to both networks are syntactically and semantically alike.**

# Real-Time Bus versus Field Bus

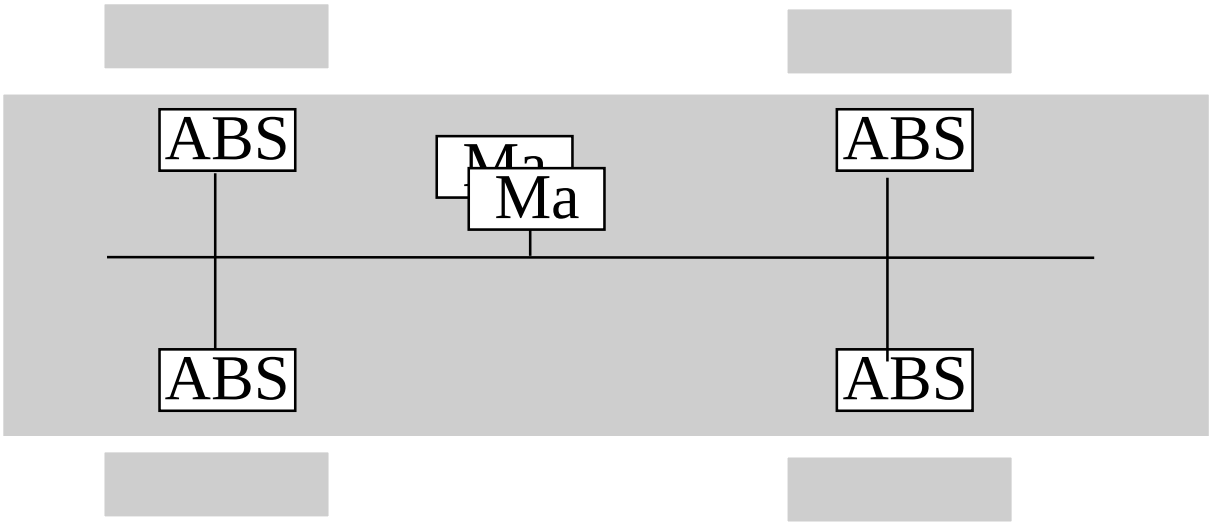


# TTP/C Protocol for System Bus

The Time-Triggered Protocol (TTP), connecting the nodes of the system, is at the core of the Time-Triggered Architecture. It provides the following services:

- guaranteed for composability by full specification of the temporal properties of the interfaces.
- timely membership service (fast error detection)
- replica determinism
- replicated communication channels (support of fault- tolerance)
- good data efficiency
- limited flexibility (preplanned mode changes)

# Membership Service: Intelligent ABS




The ABS design is fail safe. This implies that the control system must provide consistent error detection with a short latency. This is provided by a **membership service**.

# TTP-A Protocol for Smart Transducers

- Composability and Testability
- Provides Standard Interface File System (IFS)
- Universal Smart Transducer Interface
- Latency Guarantee for Control Applications, Clock Synchronization better than .1 msec
- Good Error Detection for fail safe operations
- Low Cost for intelligent sensors, smallest implementation less than 2 kbytes of ROM, 64 bytes of RAM (including IFS, software UART at 10 kbits on single wire)
- Fault tolerance at system level (duplicated buses)

# Well-specified CNI is the Key to Composability

- partitions a system into nearly autonomous subsystems that can be developed and tested independently
- enables a predictable timely information flow from one subsystem to the other subsystems at the proper level of abstraction without introducing unnecessary dependencies among the subsystems
- provides a global time base
- inhibits the propagation of errors from one subsystem to another subsystem and provides simple external failure modes, e.g., fail-silence.
- enables the timely detection of errors of nodes (membership)
- hides the internal details of the subsystem implementation

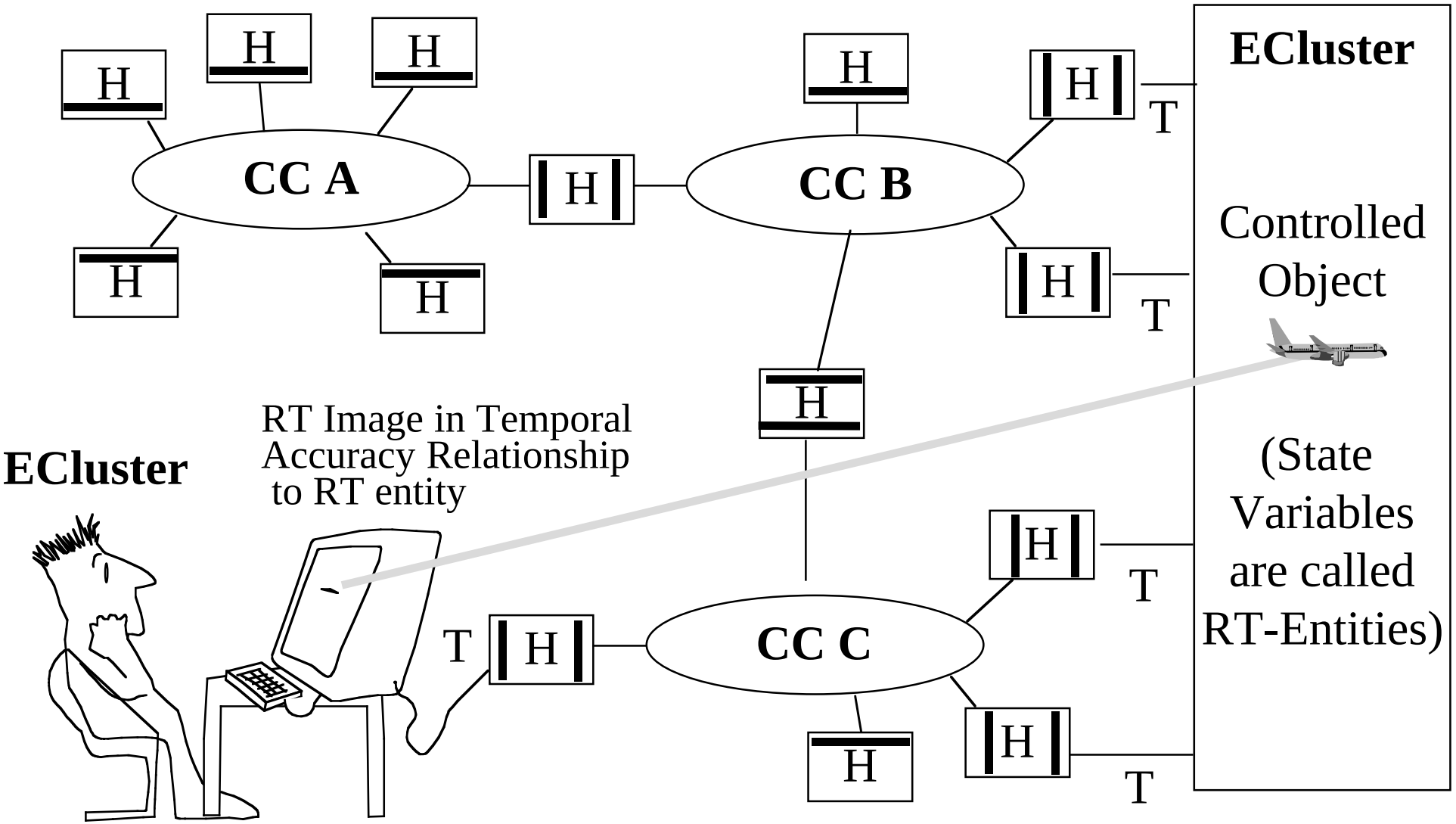
# The CNI forms *Temporal Firewalls*



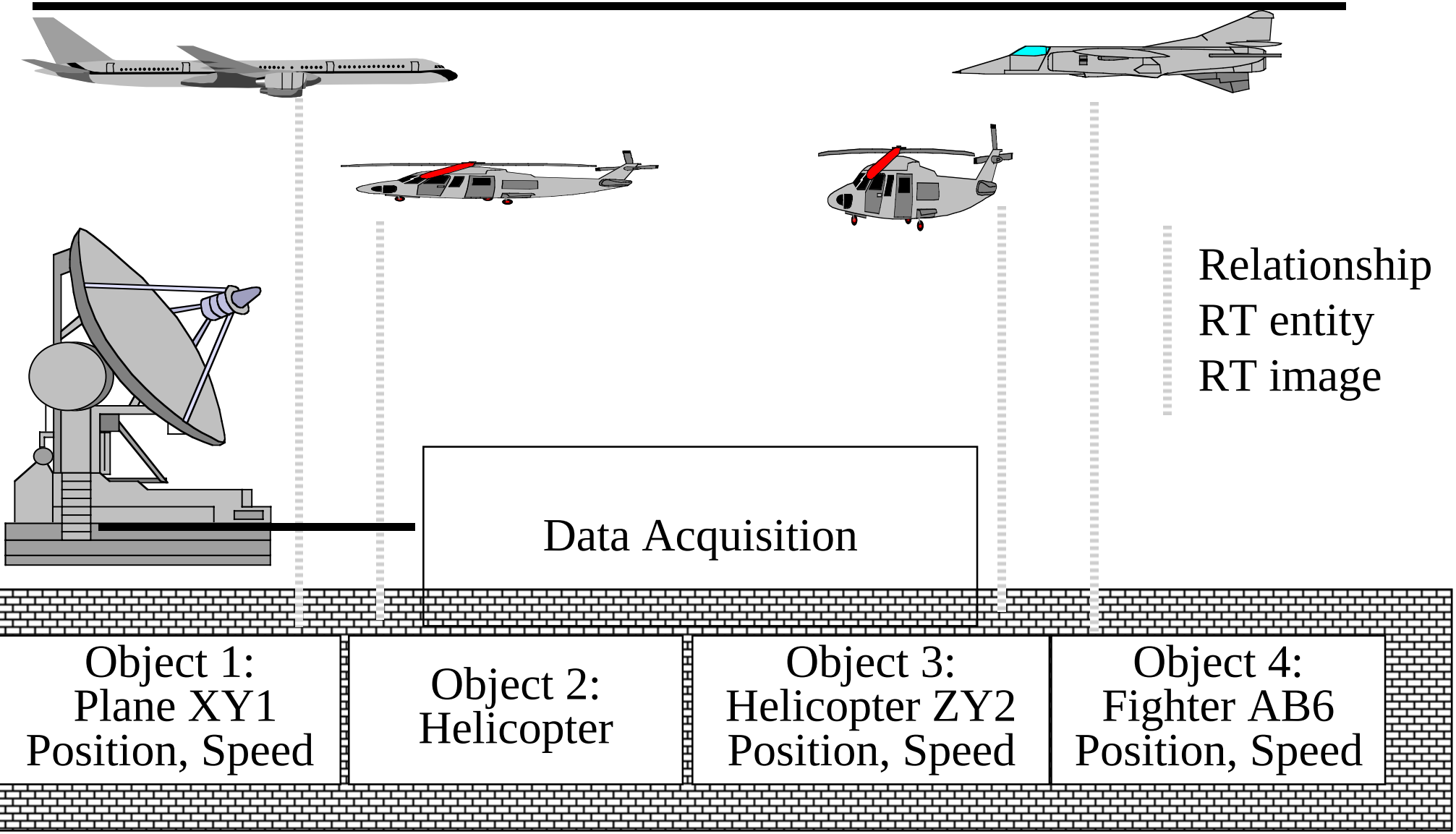
A *temporal firewall* is a unidirectional data-sharing interface with state-data semantics where at least one of the interfacing subsystems accesses the temporal firewall according to an *a priori* known schedule and where the temporal accuracy of the information in the temporal firewall is known *a priori*.

On the conceptual level, the TTP-CNI between the host computer and the communication network (or the controlled object) can be seen as erecting two unidirectional temporal firewalls an *input firewall* and an *output firewall* that connect the component to its environment.

# Example: A Five Cluster System



# Example of a Temporal Firewall



# Top Down Architecture Design in the TTA

A top down architecture design phase in the TTA is concerned with the following issues:

- partition the system into a set of nearly autonomous components
- specify the functionality of the components at an high application specific level
- specify the timely information flow among the components, the interaction patterns
- Design the MEDLs of the communication controllers and thus determine the points in time when the communication system accesses the CNIs.

The first design phase results in the specification of temporal firewalls between the host computers and the communication system.

# Component Design

- The components are designed considering the constraints of the architecture design phase.
- The known points in time when information must be delivered at the CNI are the deadlines for the components
- The components can be tested with respect to the available interface specification.

# Bottom-up Design--Reuse of Components

The bottom up design takes advantage of the existing COTS components:

- The input firewall parameters determine what a user is expected to supply
- The output firewall parameters determine what a user can rely upon

The architecture design must proceed taking these component characteristics as constraints.

The temporal firewalls of the new components can be designed according to the top-down process.

# Conclusions

- The automotive industry has the volume to determine the proper composable COTS components for the real-time systems market.
- With the announcement of MOTOROLA, the world's premier supplier of automotive chips, to support the time-triggered technology with a number of system chips, the TTA is in a good position to be the "technology of choice" for many future industrial applications.
- A paradigm shift from today's dominating event-triggered architectures to the time-triggered architecture can thus be expected.