

---

# **The Temporal Firewall--A Standardized Interface in the Time-Triggered Architecture**

H. Kopetz  
TU Vienna, Austria  
July 2000

- ◆ Introduction
- ◆ Temporal Accuracy of RT Information
- ◆ The Time-Triggered Model of Computation
- ◆ The Temporal Firewall
- ◆ The Time-Triggered Architecture
- ◆ Conclusion

# Essential Characteristics of RT Systems:

---

- **Physical time is a first order concept:** There is only one physical time in the world and it makes a lot of sense to provide access to this physical time in all nodes of the distributed real-time system.
- **Time-bounded validity of real-time data:** The validity of real-time data is invalidated by the progression of real-time.
- **Existence of deadlines:** A real-time task must produce results before the deadline--a known instant on the timeline--expires.
- **Inherent distribution:** Smart sensors and actuators are nodes of a distributed real-time computer system.
- **High dependability:** Many real-time systems must continue to operate even after a component has failed.

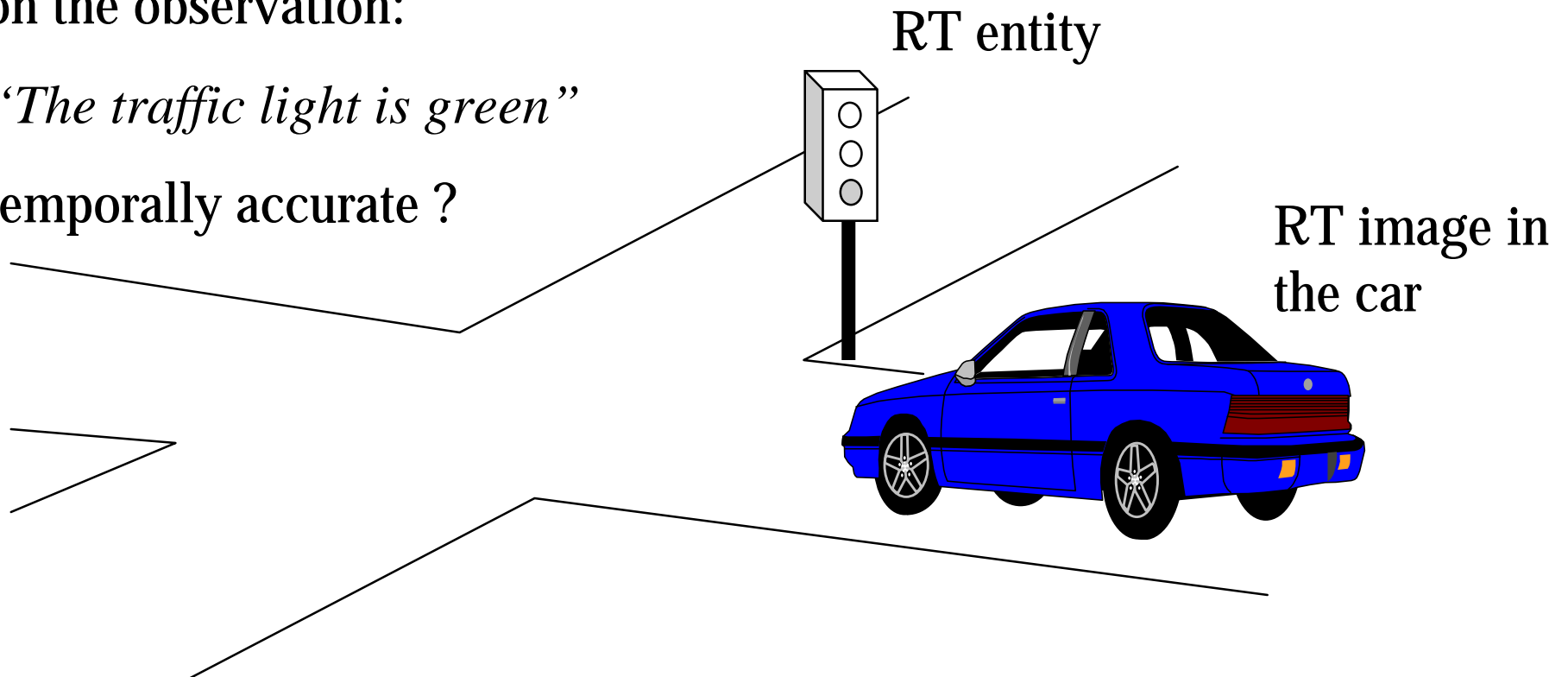
# Temporal Accuracy of Real-Time Information

---

How long is the RT image, based on the observation:

*“The traffic light is green”*

temporally accurate ?



## Definition: Temporal Accuracy

---

The temporal accuracy of a RT image is defined by referring to the recent history of observations of the related RT entity. A recent history  $RH_i$  at time  $t_i$  is an ordered set of time points  $\langle t_i, t_{i-1}, t_{i-2}, \dots, t_{i-k} \rangle$ , where the length of the recent history

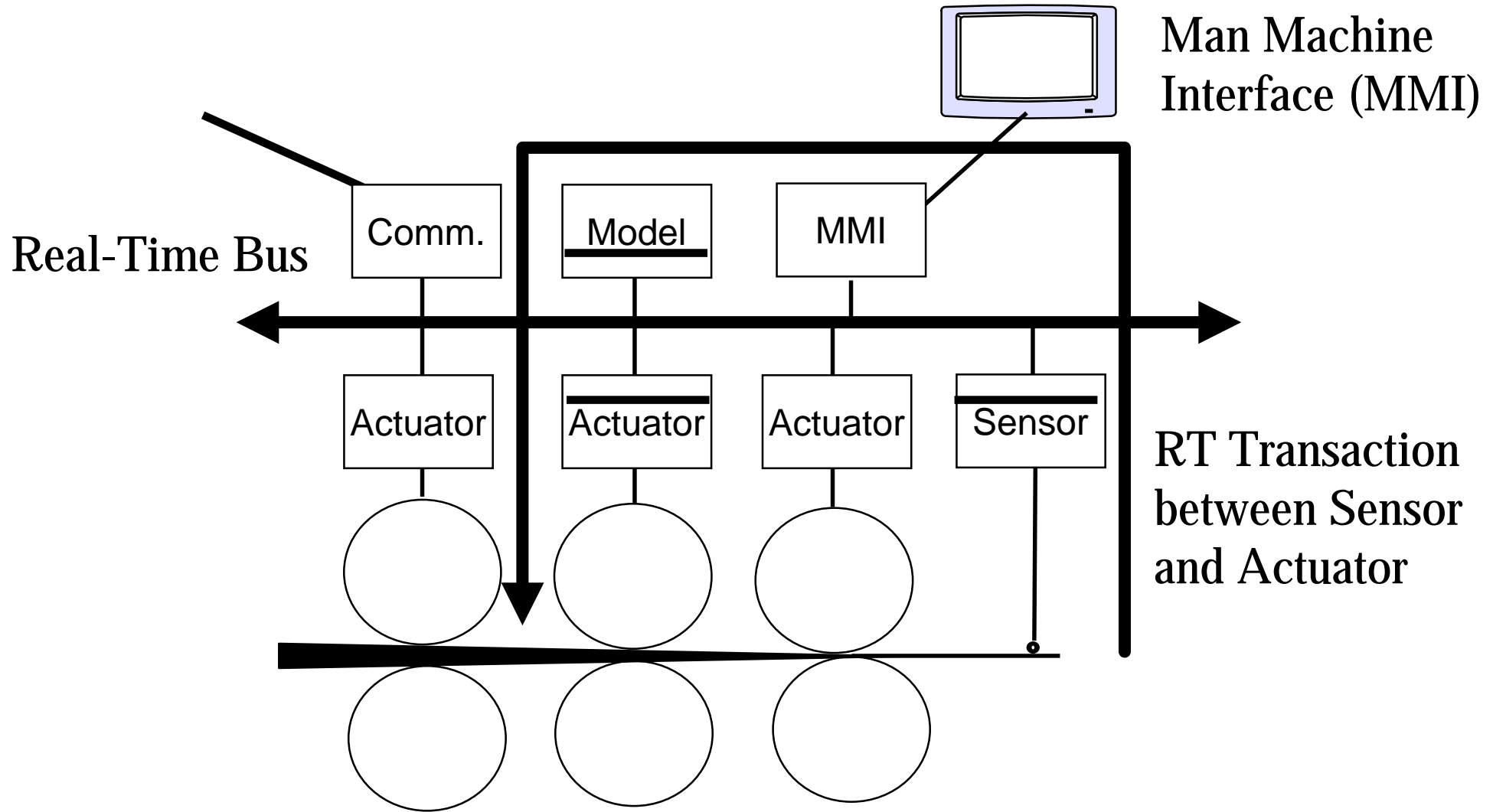
$$d_{acc} = t_i - t_{i-k}$$

is called the temporal accuracy. Assume that the RT entity has been observed at every time point of the recent history. A RT image is temporally accurate at the present time  $t_i$

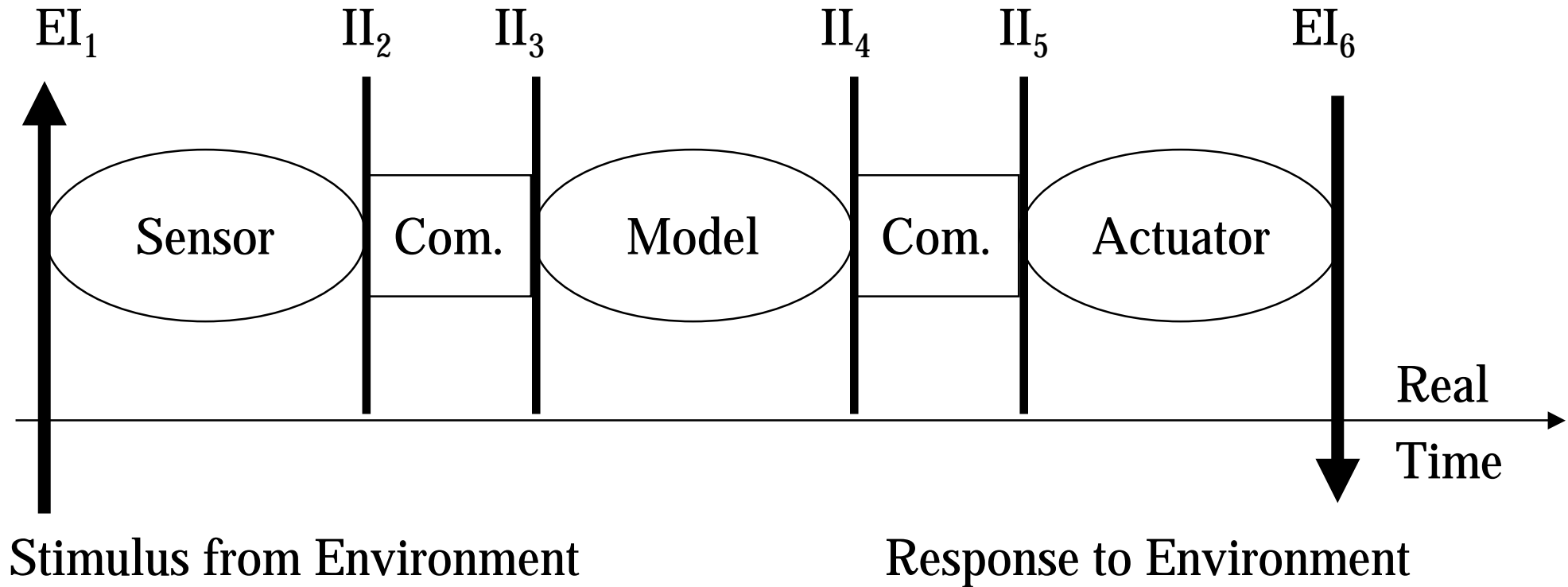
if

$$\exists t_j \in RH_i: \text{Value}(RTimage_{att_i}) = \text{Value}(RTentity_{att_j})$$

# An Example: Rolling Mill

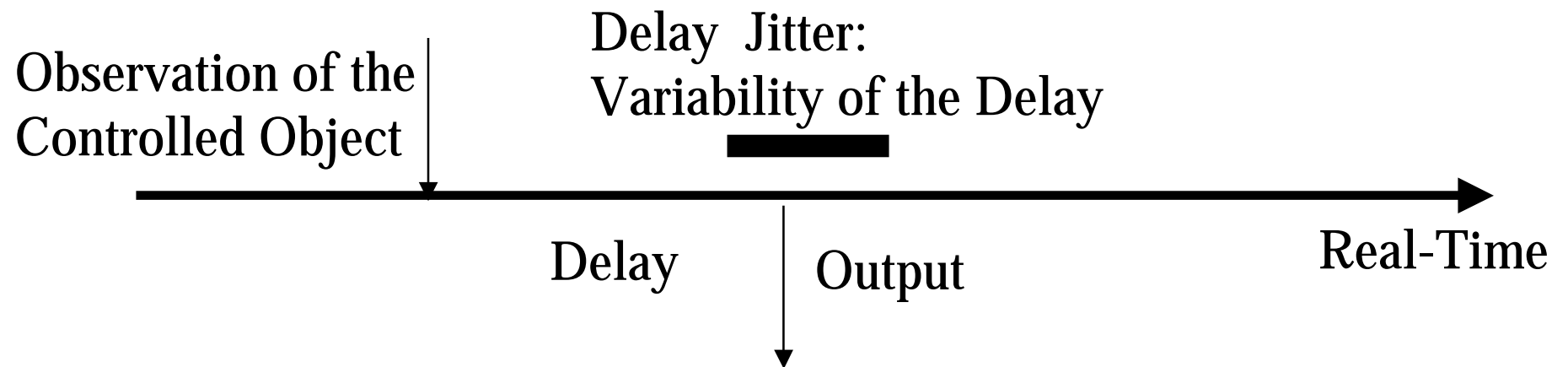


# Real-Time Transaction-Phase Alignment



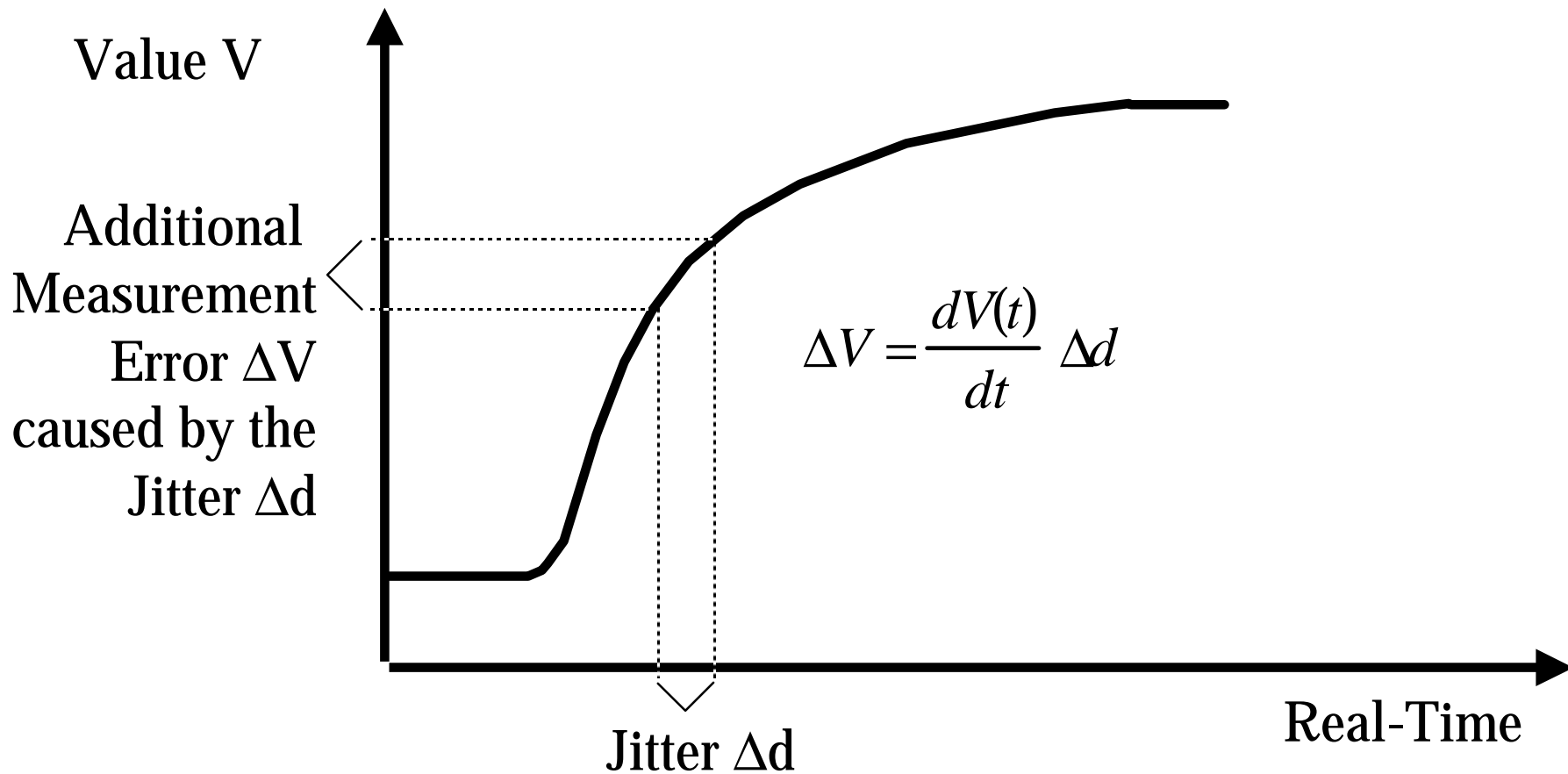
# End-to-End Delay Jitter

---



# The Effect of Jitter

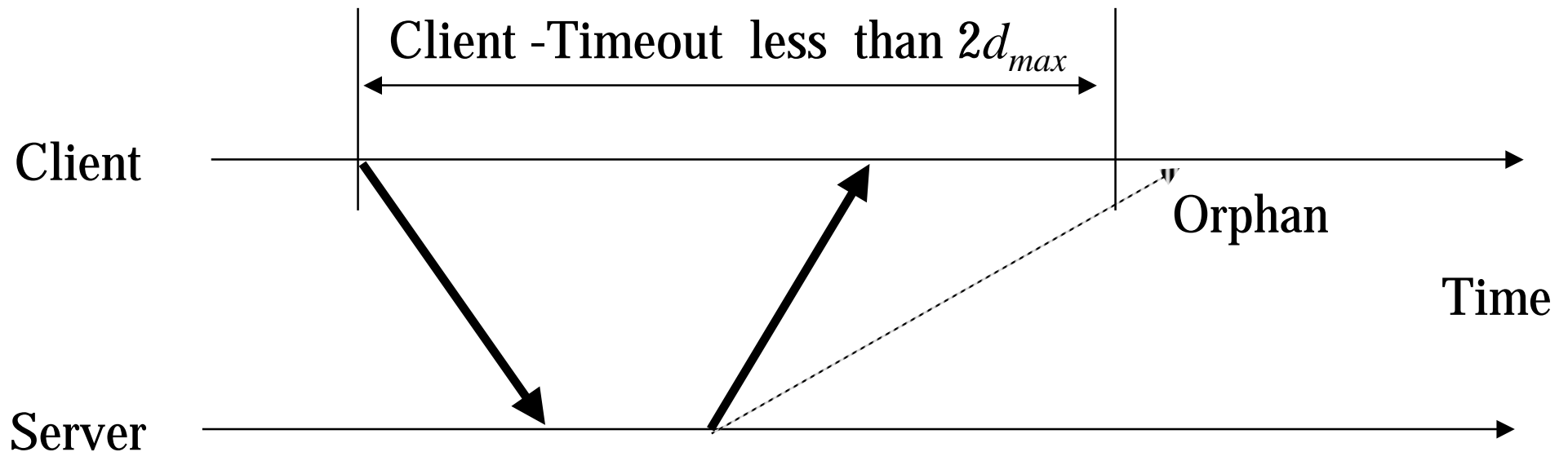
---



# The Effect of Jitter: Orphans

---

Request Response Transaction between a Client and a Server:



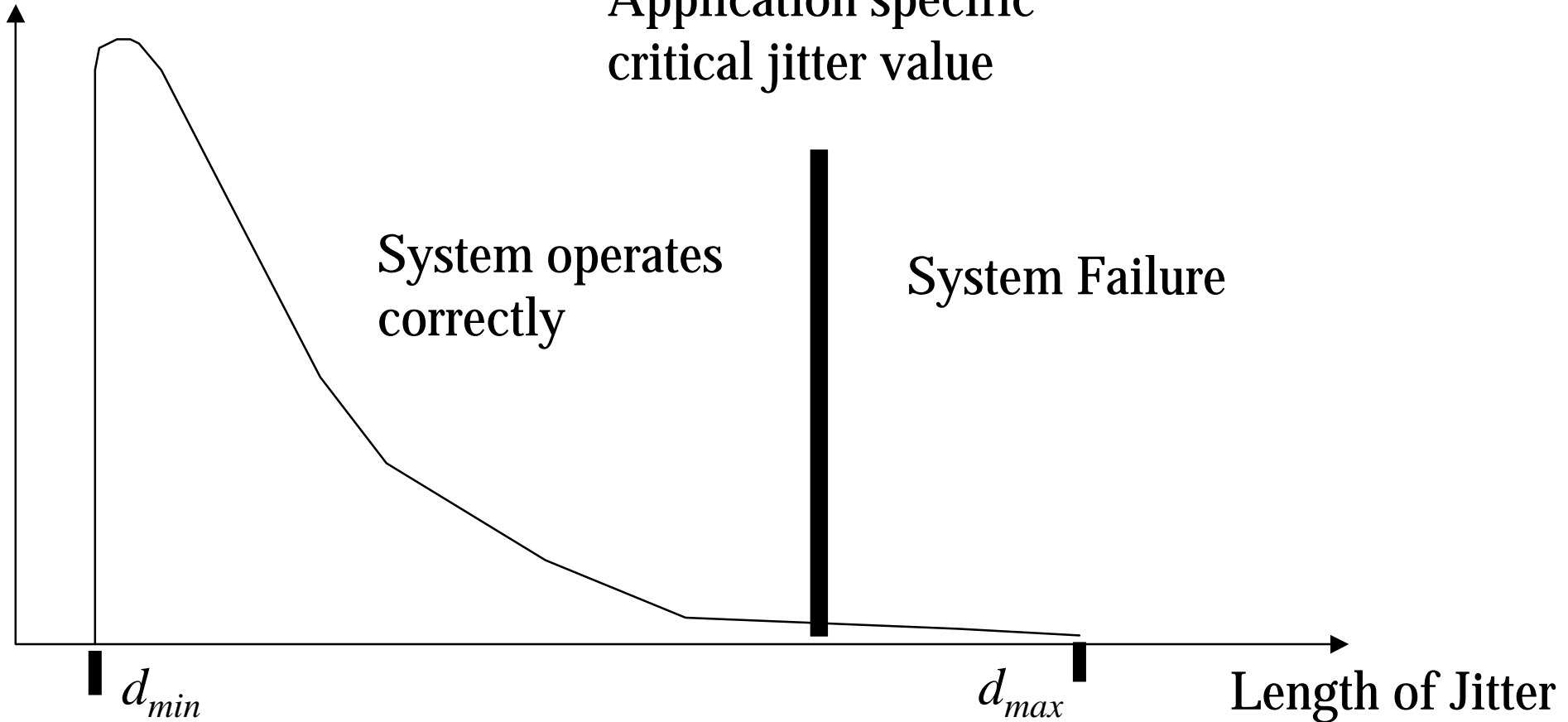
How large is  $d_{max}$  ?

It is not contained in the interface specification, available at the subsupplier.

# Probability of “Long” Jitter

---

Probability Density



Most of the time, the system will operate correctly.

# Time-Triggered (TT) Model

---

Describes real-time systems by the recursive application of the following building blocks:

- **Interface:** a data-sharing boundary between two communicating subsystems containing temporally accurate real-time data..
- **Communication subsystem:** transports real-time data from one interface to another interface.
- **Host computer:** Reads input data from an interface, performs a data transformation and writes output data into an interface.
- **Transducer:** Transforms data from an interface into a form required by the system environment and transforms data from the environment into the form required by an interface.

# Interface--Temporal Firewall

---

- \_ Fully specified in the value domain and in the temporal domain--unidirectional information and control flow.
- \_ Contains temporally accurate state information. In case the data has lost its validity because of the progression of time, the interface will invalidate the data (No queueing of information in the interface).
- \_ The points in time, defined on a sparse time-base, when the data are accessed by the communication system are known *a priori*.
- \_ Can be seen to erect a *temporal firewall* between the interacting subsystems.
- \_ Is an error propagation boundary.

# Communication System

---

- Transports state data from one interface to another interface within a known latency.
- Time-triggered: Autonomous and deterministic operation based on the progression of a global notion of time (train schedule).
- Separates the interaction patterns within the system from the information processing activities.

# Host Computer

---

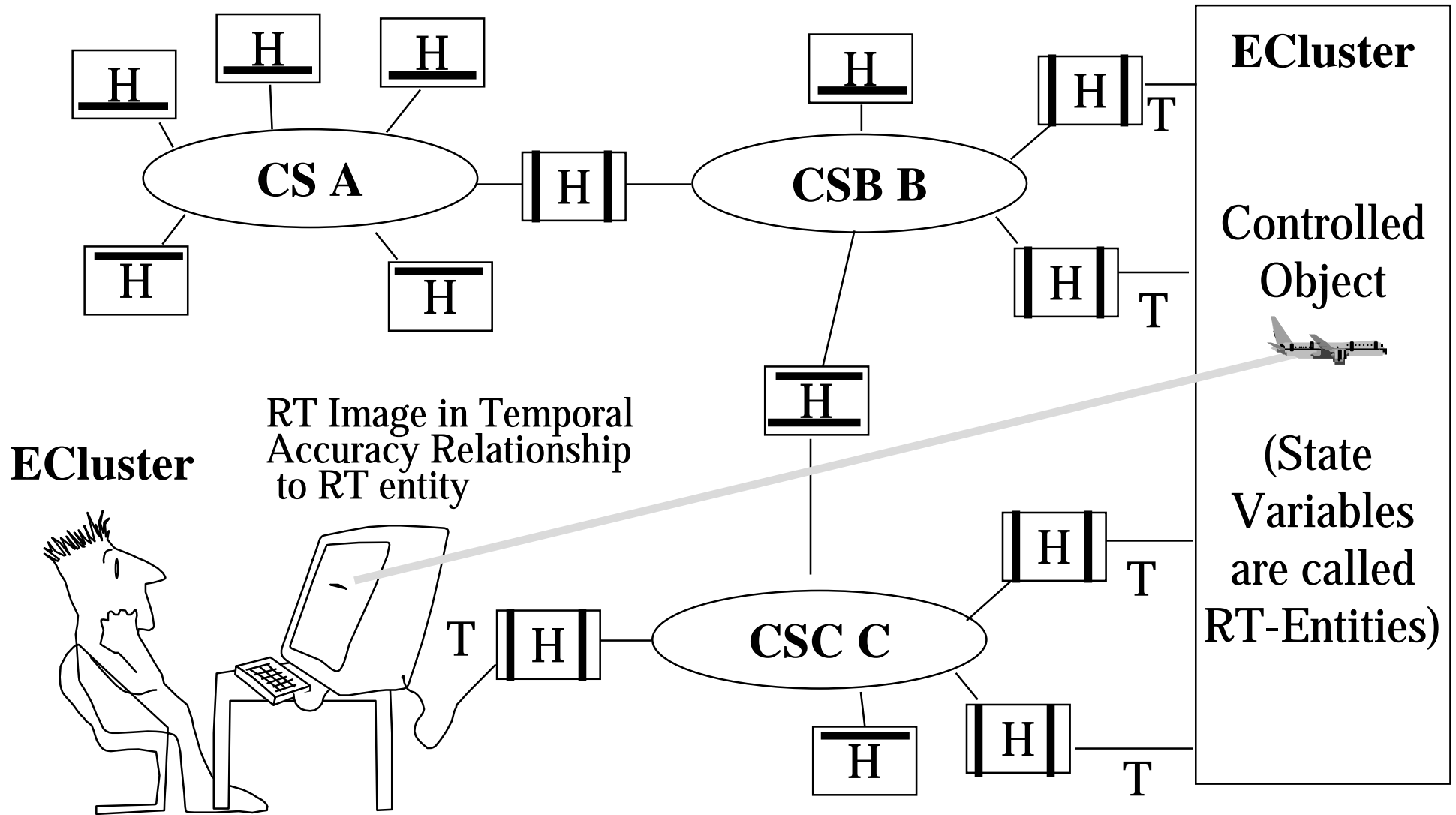
- Gets input data from an input interface at an *a priori* determined instant and must deliver output data to an output interface at an *a priori* defined later instant.
- Performs the intended data transformations within an *a priori* known WCET.
- Is free to schedule its work in whatever way is best, as long as the timing constraints at the interfaces are observed.

# Transducer

---

- \_ Connects an interface to a data source or a data sink in the environment.
- \_ Accesses the interface at *a priori* known instants to deliver input information.
- \_ Access the interface at *a priori* known instants to fetch the output information.
- \_ Changes the representation from internal to external.

# Example: A Five Cluster System



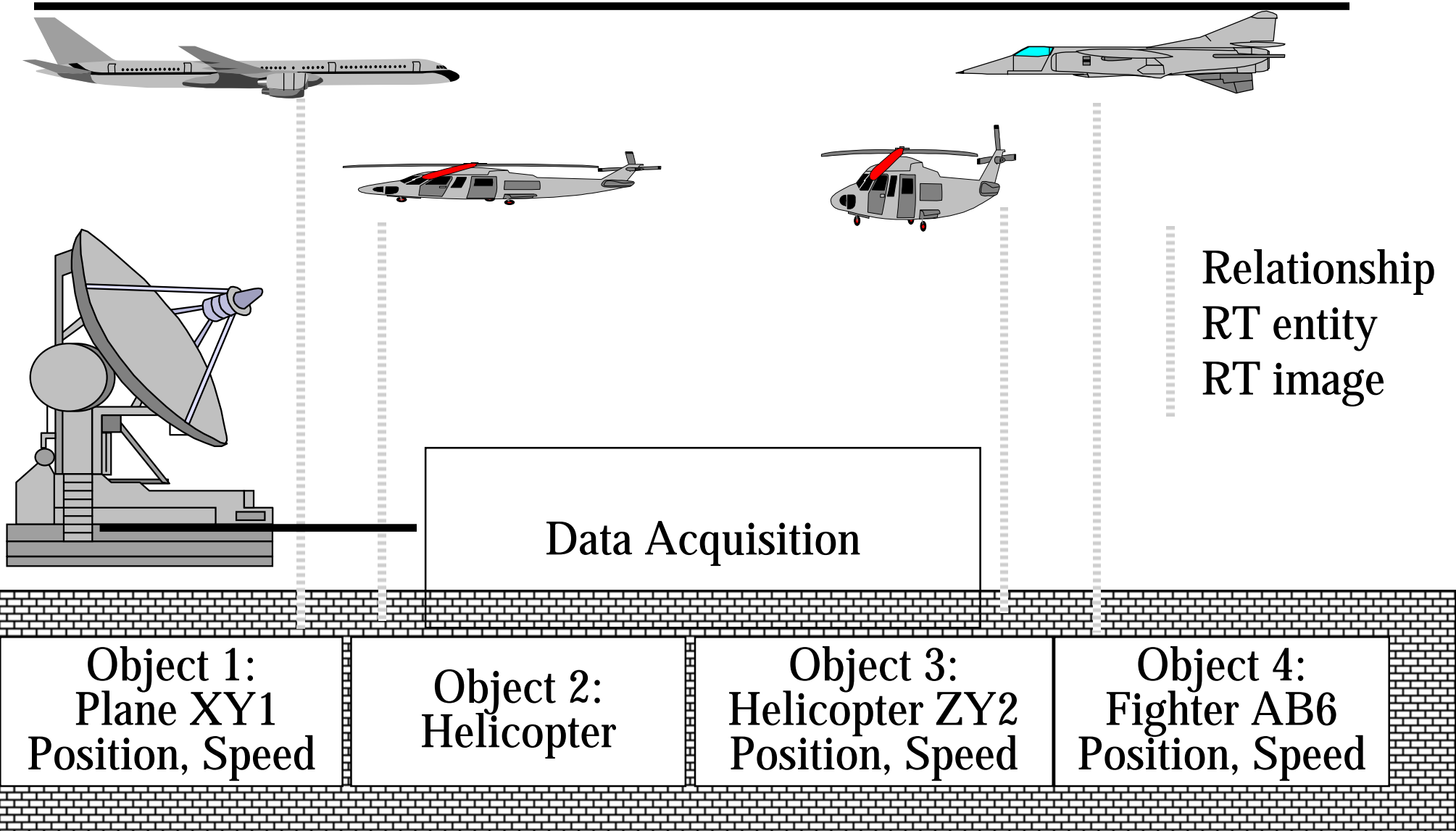
# Temporal Firewall

---

An Interface between a communication system and a host computer with the following characteristics:

- ◆ A data-sharing interface containing state data (no queues--update in place)--no strict synchronisation between *writer* and *reader(s)*.
- ◆ The instants when information is delivered by the communication system are known *a priori* (Input Firewall).
- ◆ The instants when information is fetched by the communication systems are known *a priori* (Output Firewall).
- ◆ The interface is aware of the global time.
- ◆ The interface acts as a control-error propagation boundary.

# Example of a Temporal Firewall



# A Temporal Firewall is a Natural Concept

---

- \_ A temporal firewall is a *high-level* abstract concept.
- \_ It is a small and stable unidirectional interface that provides understandable abstractions of the relevant properties of the interfacing subsystems.
- \_ *Timeliness* is an *integral* part of the temporal firewall concept.
- \_ Conceptually, the RT images in the temporal firewall are closely related to the image presented by a sensor of an analog RT entity in the environment.
- \_ Temporal firewalls are thus based on an accustomed view of the world.

# Stable Properties of Temporal Firewalls

---

The following stable properties of temporal firewalls are known *a priori* to all interfacing partners:

- The addresses (names) and the syntactic structure of the data items in the temporal firewall. The meaning of the data items is associated with these names.
- The *temporal accuracy* of the data items in the temporal firewall. This knowledge is important to guide the information consumer about the minimum rate of sampling the temporal firewall.
- The points on the global time base when the data items in the temporal firewall are accessed by the TT subsystem. This information enables the avoidance of race conditions between the producer and the consumer.

# Obligations of the Subsystems

---

- **Producer:** The producer of the RT-images stored in the temporal firewall is responsible that the *a priori* guaranteed *temporal accuracy* of the RT-images is always maintained. It must update the state information with such a frequency that the guaranteed temporal accuracy is sustained even immediately before the point of update.
- **Consumer:** Based on the *a priori* knowledge about the temporal accuracy of the RT images in the temporal firewall, the consumer must sample the information in the temporal firewall with a sampling rate that ensures that the accessed information is temporally accurate at *its time of use* of this information.

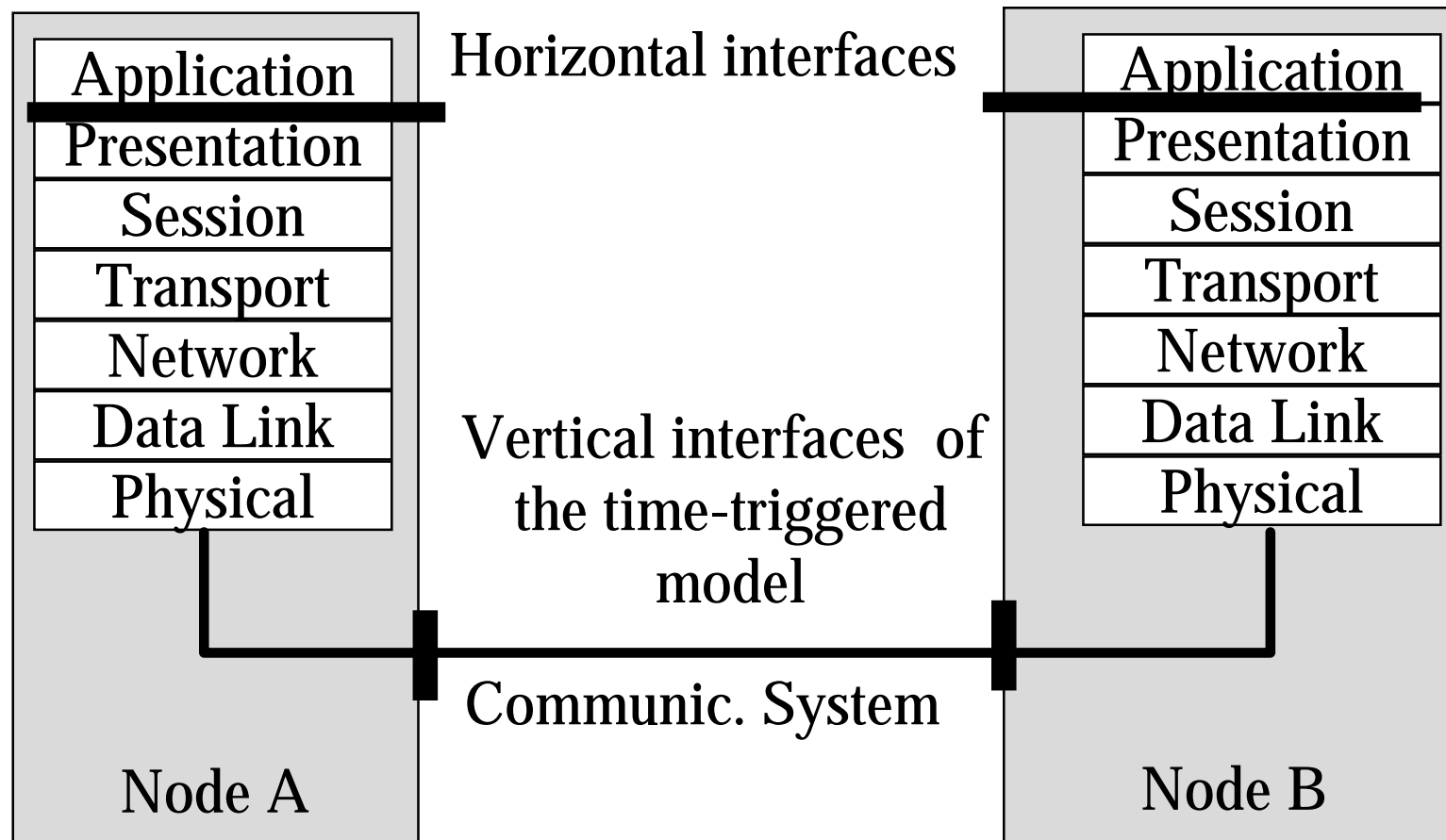
# Temporal Firewalls and Validation

---

Assume a component that is encapsulated between two temporal firewalls, and *input firewall* and an *output firewall*. These two firewalls form the only interfaces of this component to its environment.

- The stable properties of the input firewall form important *preconditions* for the validation of the component under consideration. Many assumptions about the environment are contained in the specification of this input firewall.
- The stable properties of the output firewall form important *postconditions* of the validation.
- In the validation process it must be demonstrated that the postconditions, given in the output firewall specification, are always TRUE, provided the preconditions associated with the input firewall hold.

# Comparison: TT versus Client-Server



# Comparison: TT versus Client-Server

---

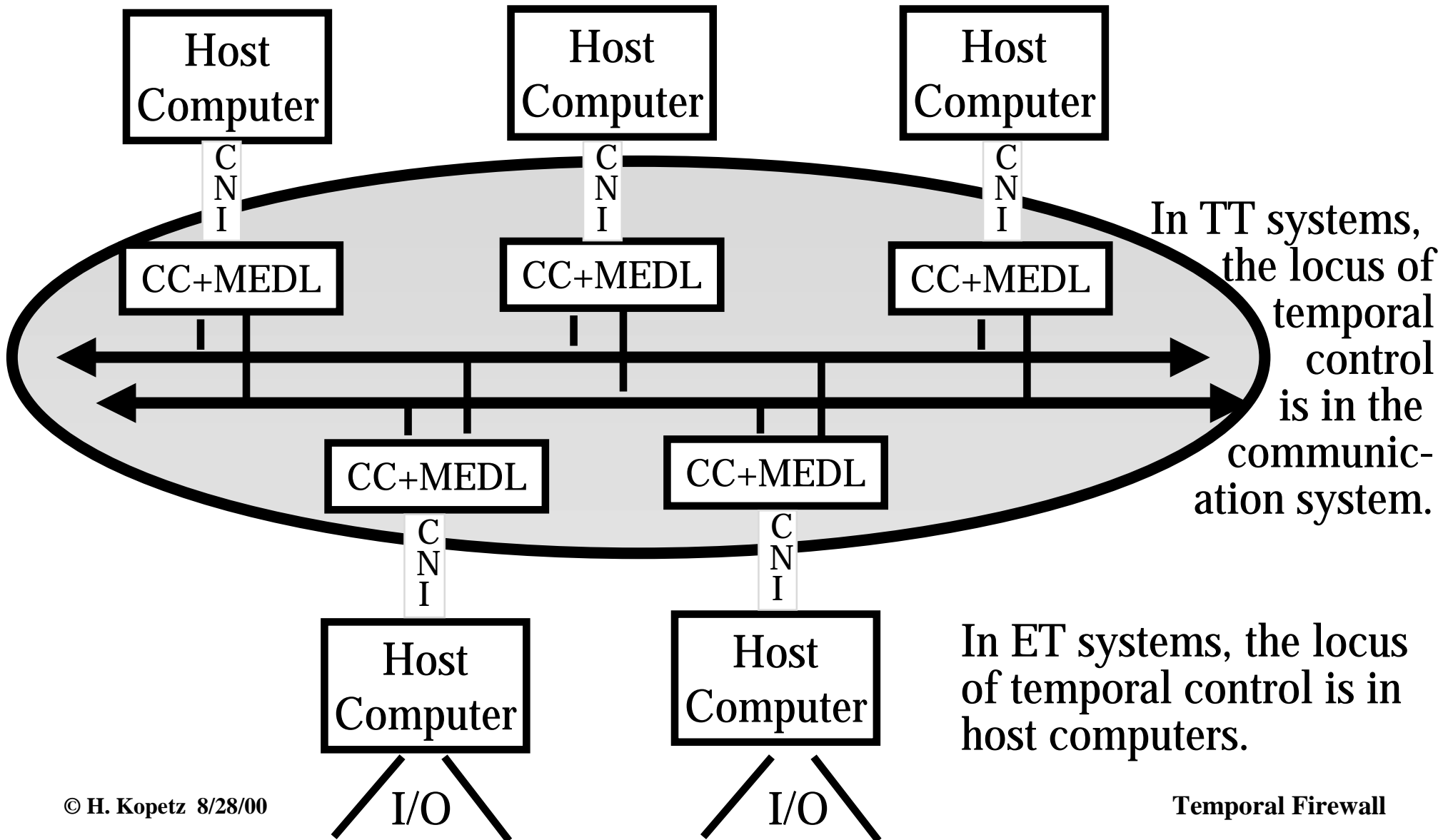
<u>Characteristic</u>	<u>Client-Server</u>	<u>TT-Model</u>
Focus	Process based	State based
Structuring	horizontal--layering	vertical--partitioning
Driven by	Event Messages	Sampling of states
Locus of Temporal Control	Global System wide events event-triggered	Local within subsystems, time-triggered
Temporal Concern	Timely request-response transactions	Temporal accuracy of interface data
Temporal Composability	Not supported	supported

# The Time-Triggered Architecture (TTA):

---

- ◆ Supports the decomposition of a large hard real-time system into nearly autonomous subsystems with precise (temporal and value) interface specifications--**the Communication Network Interfaces (CNI) are Temporal Firewalls.**
- ◆ Allows the independent development and testing of these subsystems versus the given interface specifications, avoiding unplanned integration effects.
- ◆ Provides services needed for the implementation of fault tolerance, such as a fault-tolerant global time base and a membership service.
- ◆ Makes it possible to design and implement real-time systems with *a priori* predictable temporal behaviour and thus provides a solution to the most pressing interoperability and software reuse problems.

# Global Interactions versus Local Processing



# Services of the TTP/C Protocol

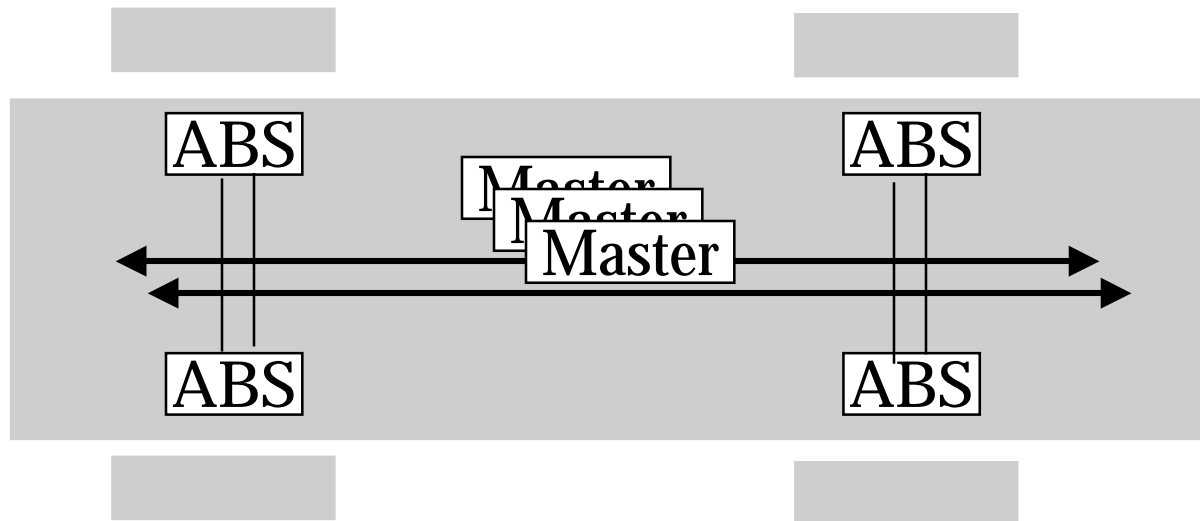
---

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

- ◆ Predictable communication with minimal jitter.
- ◆ Fault-tolerant clock synchronisation.
- ◆ Timely membership service (fast error detection).
- ◆ Replicated communication channels (support of fault-tolerance).
- ◆ Replica determinism.
- ◆ Good data efficiency.

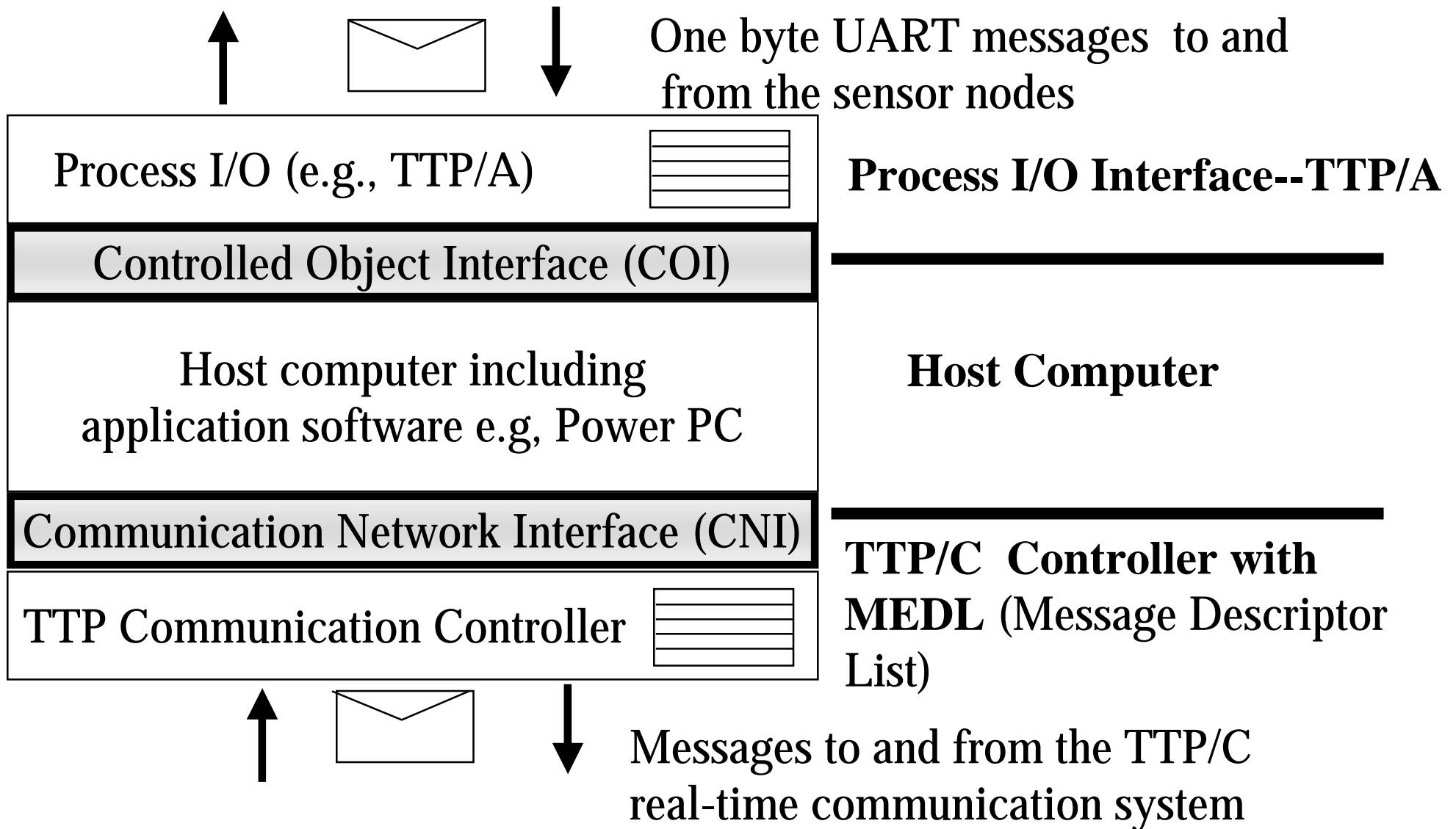
# Membership Service: Intelligent ABS

---



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**.

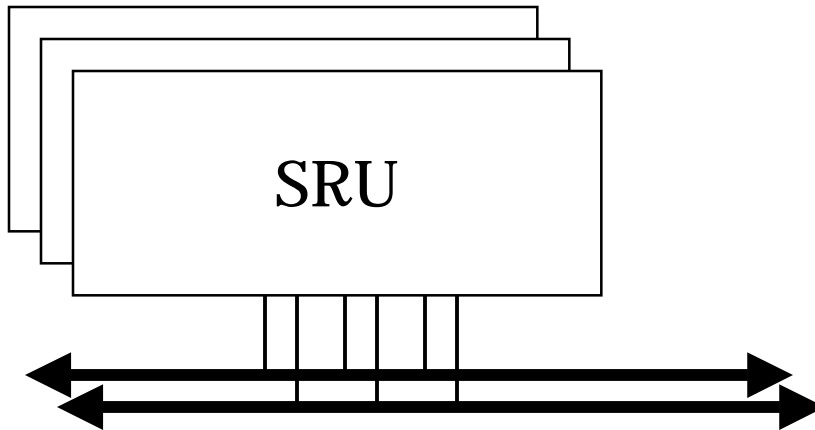
# A TTA System Node consist of three Subsystems:



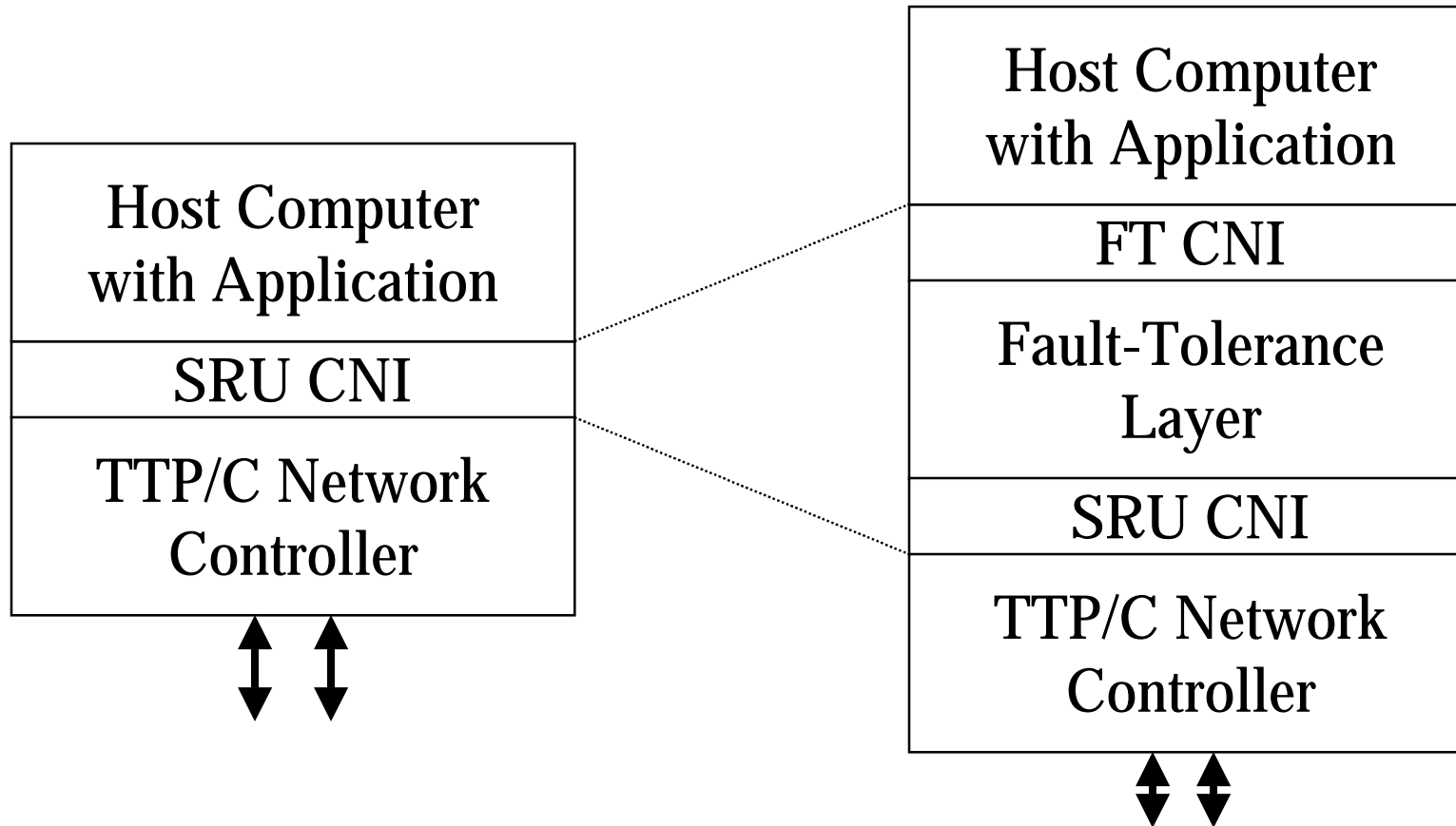
# Fault Tolerant Unit (FTU)

---

A number of SRUs are combined to form an Fault-Tolerant Unit.



# Transparent Implementation of Redundancy



The FT CNI has exactly the same structure, meaning and temporal behaviour as the SRU CNI.

# Required Software Structure in the Host

---

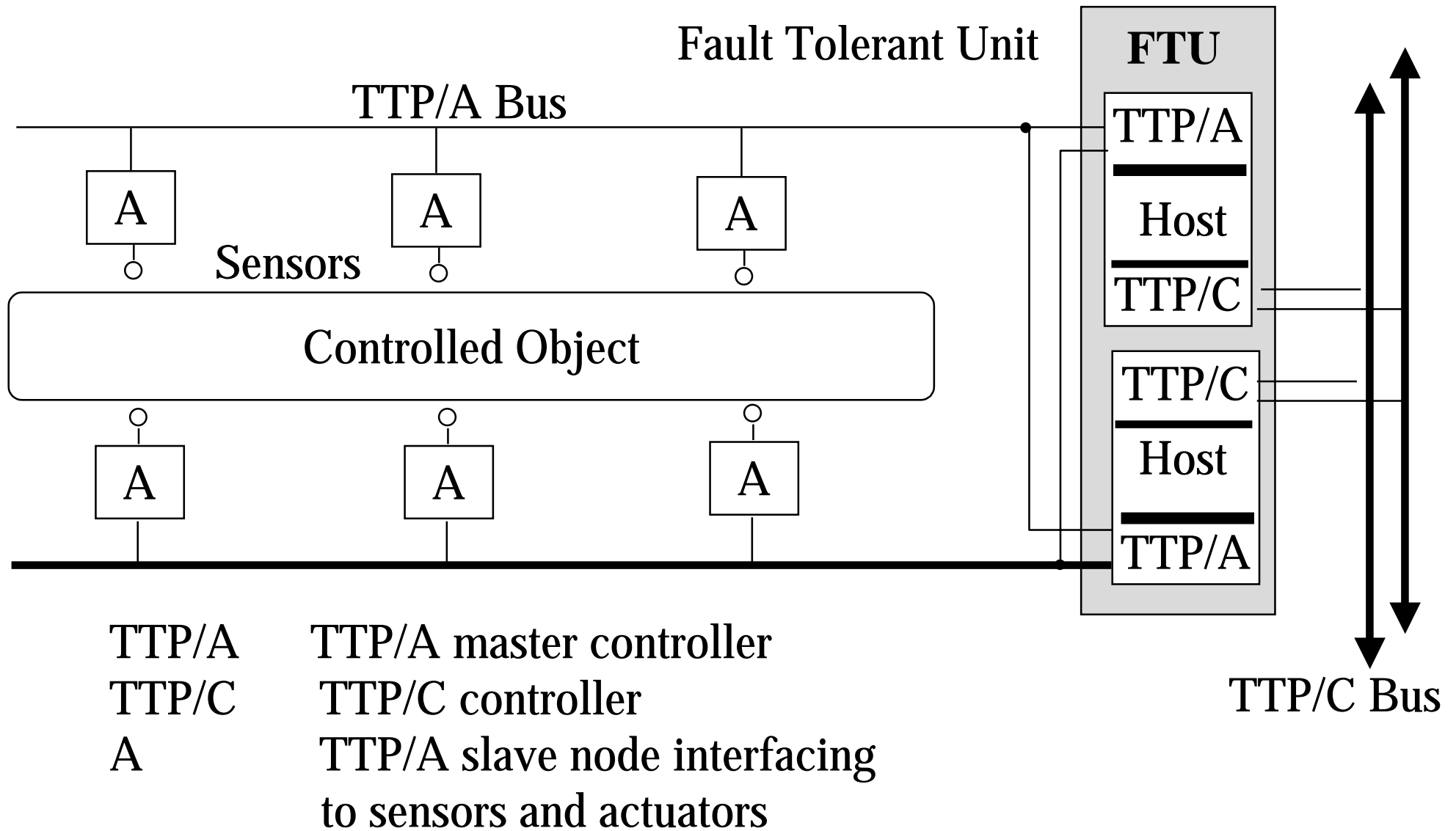
- ◆ The software execution in the host is periodic. The host reads the input data from the consumer area of the CNI and writes the output data into the producer area of the CNI.
- ◆ The execution schedule must be designed such that the host visits periodically a ground state, where no task is active and all communication channels are flushed.
- ◆ The ground state of the application must be periodically written into the producer area. It is voted by the FTU layer and presented to the host in the consumer area of the CNI.
- ◆ On startup, the SRUs are synchronised by voting on the (startup) ground state.

# Functions of the FTU Layer

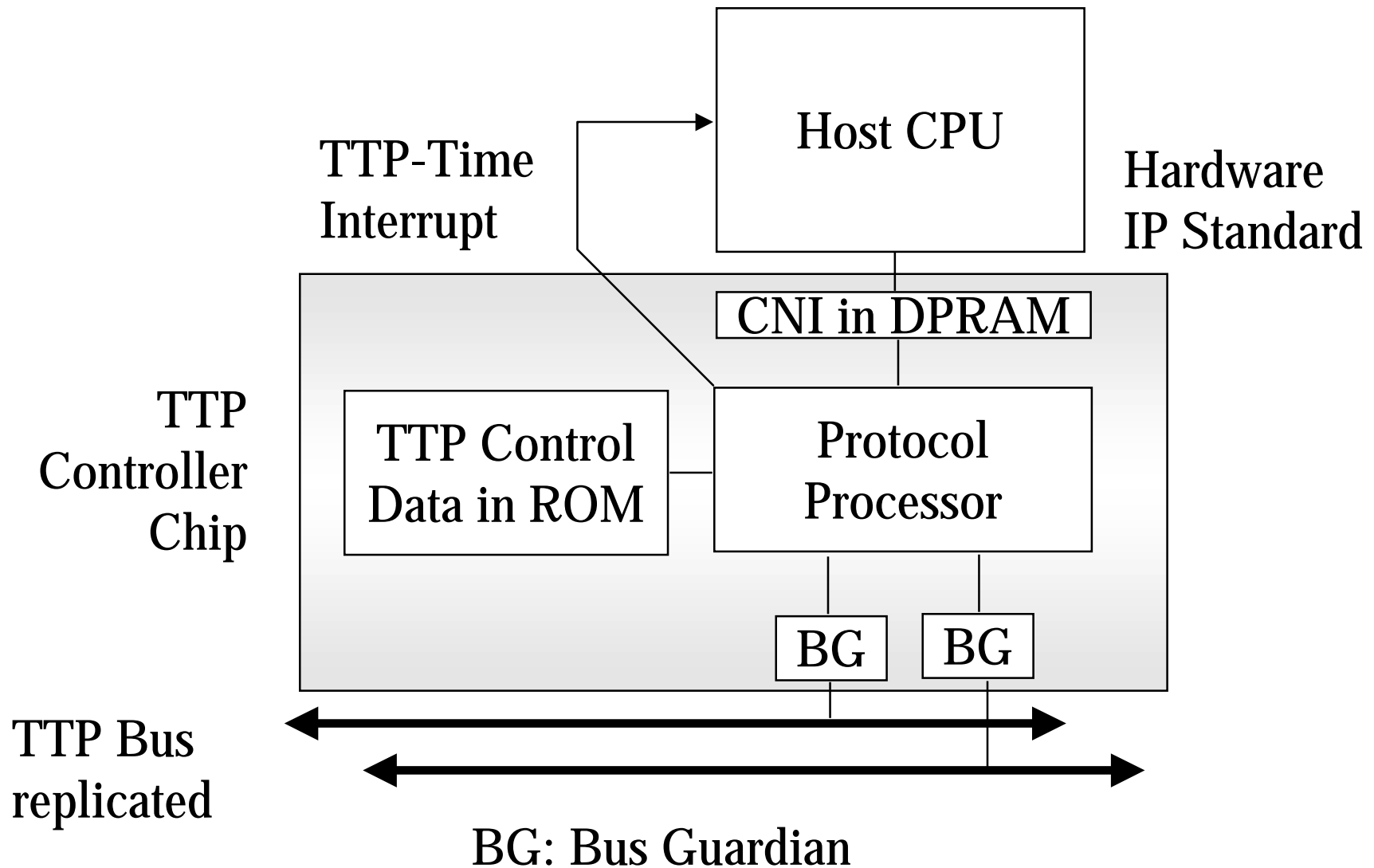
---

- ◆ Message Reduction by voting on incoming messages from the SRU CNI
- ◆ Establishment of state consistency by voting on the ground state of the SRUs forming an FTU
- ◆ Message Output to the SRU CNI
- ◆ Reconciliation of the Membership Information:  
Calculation of an FTU Membership out of the SRU membership provided by TTP/C

# Fault-Tolerant Sensor Connection



# TTP/C-Controller



# Prototype Implementation

---

- ◆ The existing TTP/C controller chip contains a protocol processor that is microprogrammed.
- ◆ The microprogram has been extended to add the FTU functionality
- ◆ The Message Descriptor List has been extended to include the FTU Layer information

On a 2 Mbit/s TTP/C system, the FTU Layer processing takes about the same Processing time as the TTP/C protocol processing.

The size of the microcode for the FTU Layer is about 25% of the size of the TTP/C protocol.

# Conclusion: The Bridge across the Gap?

---

