HARD-RoCS:
Goals and Approaches

Kane Kim
UCI DREAM Lab
khkim@uci.edu, http://dream.eng.uci.edu/

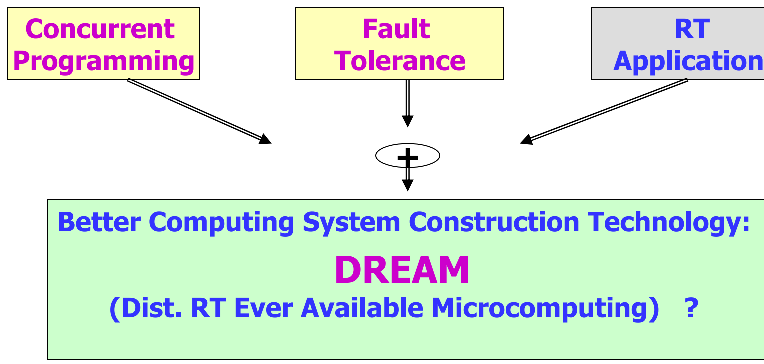
For presentation at
the WG 10.4 Meeting, Sal
January 2003

Why am I working on Middleware ?

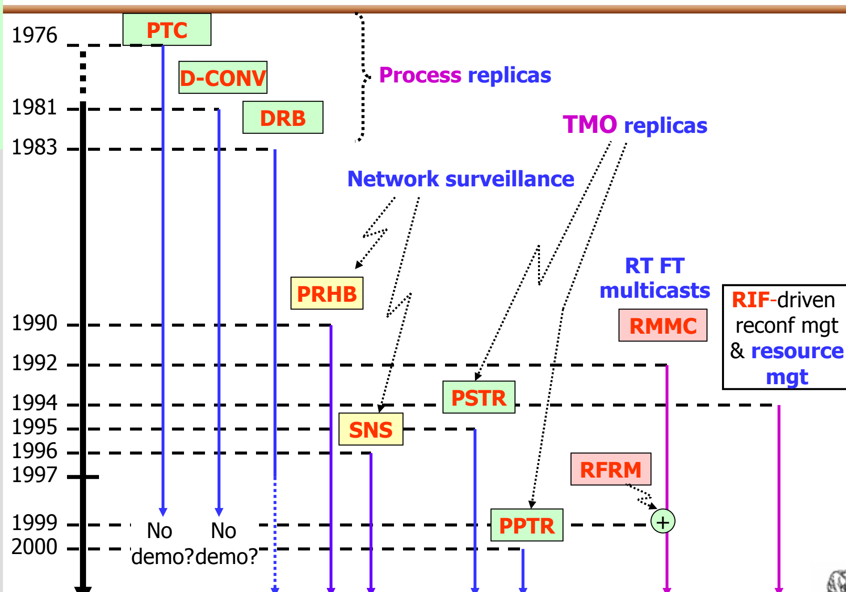
- Even the RT OS kernel field was invaded by industry giants some years ago
  - I had to run away
- There is hope that good ideas / techniques / mechanisms demonstrated in middleware can be picked up by the OS industry and migrate them into future Dist. OS products.

My Research Paths of Last 25 Years

- Main attempts

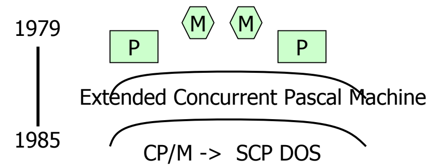


Major Dependability Techniques Pursued

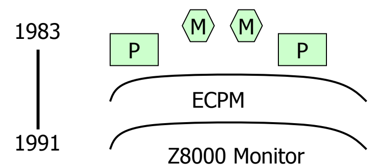


Gadgets Used

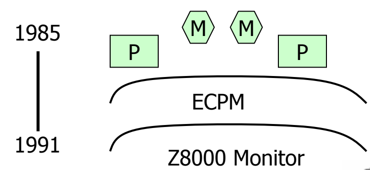
- Cromemco Z2 (Z80 based) nodes connected via RS 232serial comm cables



- Rack-mounted OEM Z8000 single-board computer nodes + Central Data cabinet Z8000 nodes connected via dual-port shared memories

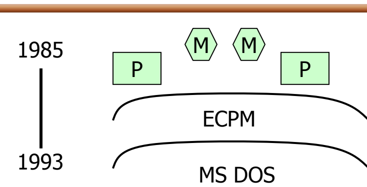


- Rack-mounted OEM Z8000 single-board computer nodes connected via Crossbar switches



Gadgets Used (cont)

- PC nodes connected via RS 232 serial comm cables
- M68000 box nodes connected via RS232 serial comm cables



- PC nodes connected via Ethernet

1988 – Present

- iPaq network
- Network of MEMS prototype nodes
- PC nodes connected via TTP monitors which are in turn connected via TTP Bus

- OptIPuter : Optically connected wide area distributed comp systems

Research Objectives

- A middleware architecture enabling easy composition and analyzable QoS
  - Tentatively named HARD-RoCS (Highly Autonomous RT Distributed Robust Computing Support)
- Pluggable, analyzable, and parameterized (PAP) components of middleware supporting real-time objects, fault tolerance, and security enforcement
  - Including their prototype implementations
- Demonstration of middleware (HARD-RoCS) and its components with real application scenarios

Target Figures of Merit

- Ease of developing networked embedded computing systems yielding tight service time bounds and recovery time bounds. (=> Development time and End-to-end timing analyzability)
- Values of service time bounds and recovery time bounds achieved for a specified range of possible type-frequencies of faults
- Ability to replace and adapt component versions (=> PAP (Pluggable, Analyzable, Parameterized) Components)

Target Applications

- Distributed multi-media processing
  - Next-generation RT VR (Virtual Reality)
  - Multi-party multi-media conferencing and collaboration
  - Distributed orchestra (?)
- Transportation automation
- Military command-control
- Non-stop web servers
- Time-sensitive health care
  - Monitoring of out-patients and patients under intensive care
  - Remotely controlled surgery

Powerful Building-Block Needed

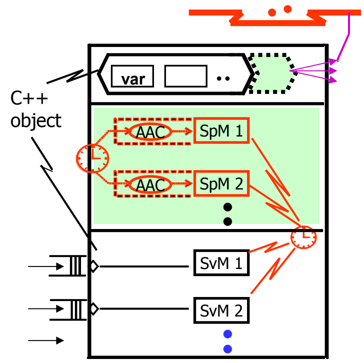
- **Without concrete QoS requirement specifications, it is difficult to avoid blind/fuzzy search for dependable computing approaches.**
  - **Spec must be concrete in both output accuracy requirements and risks caused by QoS losses.**
- **Top-level services and associated QoS requirement specifications must be specified and each decomposition of such specifications must be facilitated**
  - **Such decomposition is essentially a design based on stepwise refinement**
- **A powerful building-block is needed**

=> TMO in my view

High-Level RT Object: TMO

The Time-triggered Message-triggered Object (TMO) programming and specification scheme

- Meant to be a natural easy-to-use extension of the C++/Java technology into an RT distributed software component programming technology
- Supports design of distributable HRT objects and distributable non-RT objects
  within one general structure
- Contains only high-level intuitive and yet precise expressions of timing requirements
- Formulated from the beginning with the objective of enabling design-time guaranteeing of timely actions

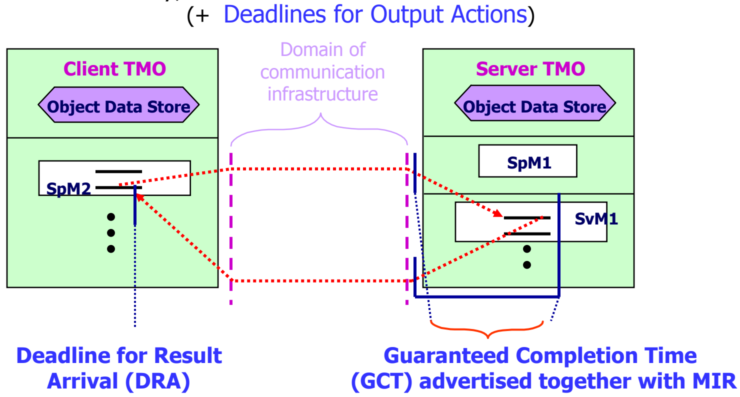


TMO (Time-triggered Message-triggered Object) Scheme

- A natural easy-to-use extension of the **C++ / Java** technology into an **RT distributed software component programming** technology

- **High-Level Specification of Timing Requirements**

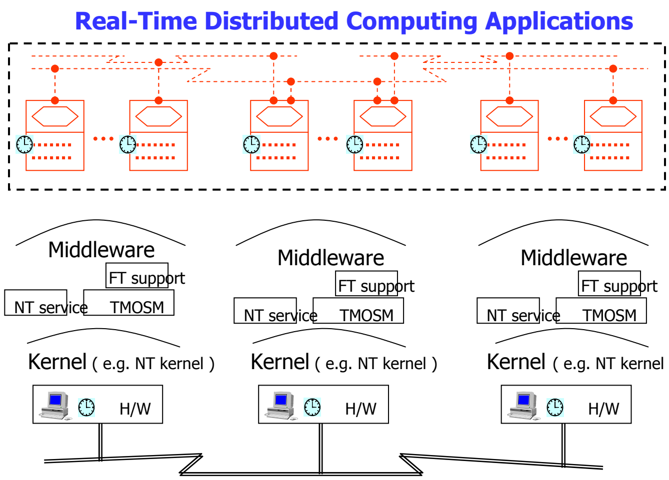
Basically, **Clients' DRAs** and **Servers' GCTs**
(+ Deadlines for Output Actions)



- **No priority, No thread**

Making Applic Programmers' Life Easier

Structure as TMO networks
relying on intelligent execution facilities



No concerns with

- **Processes & Threads**
- **Object locations** (except in avoiding overloaded nodes)
- **Low-level comm protocols**

No specification of timing reqts in indirect terms (e.g., priorities)

- **Only start-windows and completion deadlines for object methods**
- **time-windows for output actions**

TMO Programming Scheme (cont)

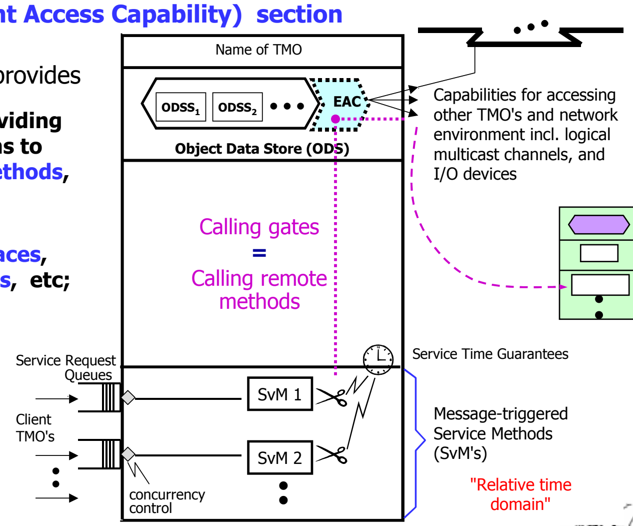
Abstract-style Remote Method Calls

L0

EAC (Environment Access Capability) section

(an ODS extension) provides

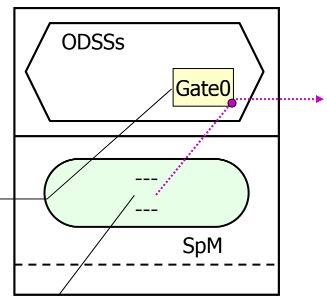
- **Gate objects providing efficient call-paths to remote object methods,**
- **I/O device interfaces, channel interfaces, etc;**



TMO Programming Scheme (cont)

Ease of Creation of a Remote Service Call

- Just two statements are needed to create a remote service call to an SvM in another TMO !!
  - Gate declaration
  - Service call thru the Gate




```
TMOGateClass  Gate0 ("TMO2", "SvM2",
   tm4_DCS_age, (7*1000*1000));
```

```
Gate0->NonBlockingSR (&SvM2Para,
   sizeof(SvM2Para),  Timestamp);
```


TMO Programming Scheme (cont)

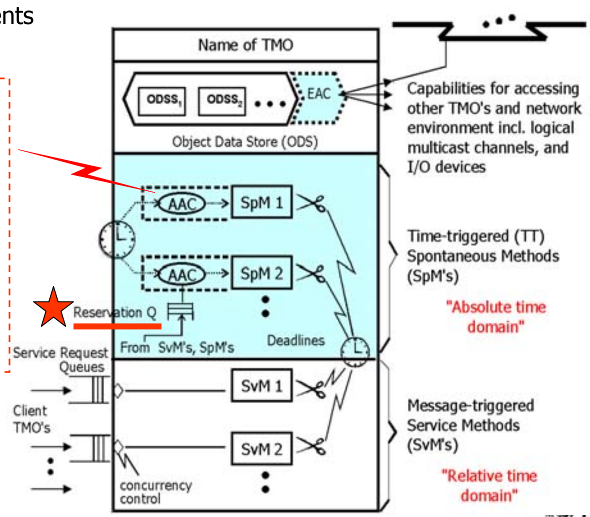
Time-Triggered Spontaneous Methods (SpM's)

- Clearly separated from the service methods (SvM's) triggered by messages from clients

Example of AAC:

{"<u>start-during</u>
(10am, 10:05am)
<u>finish-by</u> 10:10am",

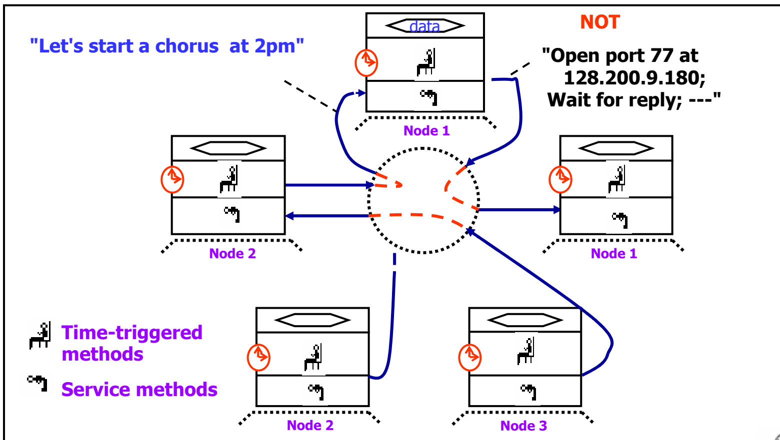
"<u>for</u> t = <u>from</u> 10am
<u>to</u> 10:50am
<u>every</u> 30min
<u>start-during</u>
(t, t+5 min)
<u>finish-by</u> t+10min" }.



Actions to be taken at real times determined at the design time appear only in SpM's

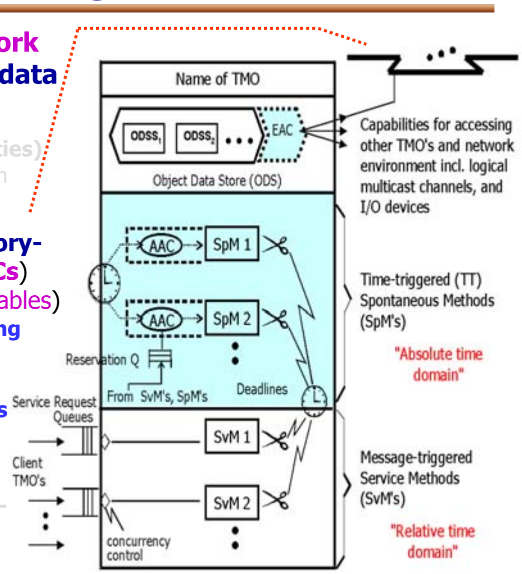
Time-Based Coordination of Distributed Actions

Imagine the Advantages of
A group of cooperating people with wrist-watches
over
A group of people not using the globally referenced time

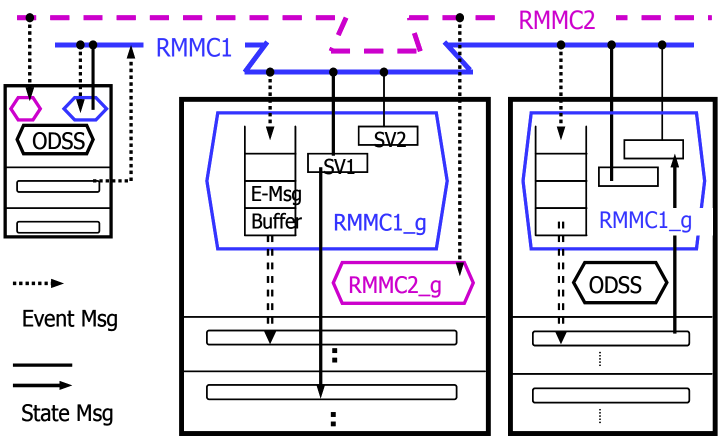


TMO Programming Scheme (cont)

- **Connections to the network environment** as **possible data members:**
  - **TMO gates** (= access capabilities) (opening the services available from other, possibly remote, TMO's)
  - **Real-time Multicast & Memory-replication Channels (RMMCs)** (incl. distributed replicated variables)
    - **Location-transparent grouping of cooperating TMOs**
    - **An alternate mechanism for inter-connecting TMO groups**
    - **Yields a tight time bound for a fault-tolerant multicast**
  - **Based on the RFRM (Release-time based Real-Time Fault-Tolerant Multicast) scheme**



RMMCs Connecting TMOs



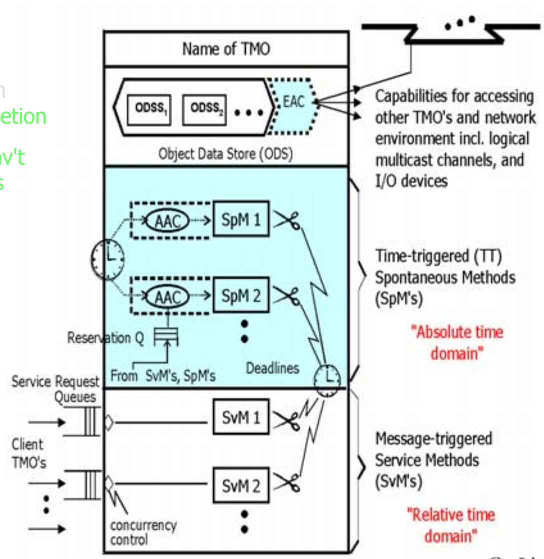
UCI
DREAM Lab

TMO Programming Scheme (cont)

L0

- Time-triggered (TT-) or spontaneous methods (SpM's)
- Time-window imposed on each output action & method completion
- Connections to the network env't (e.g., RMMC's and TMO access capabilities) as data members

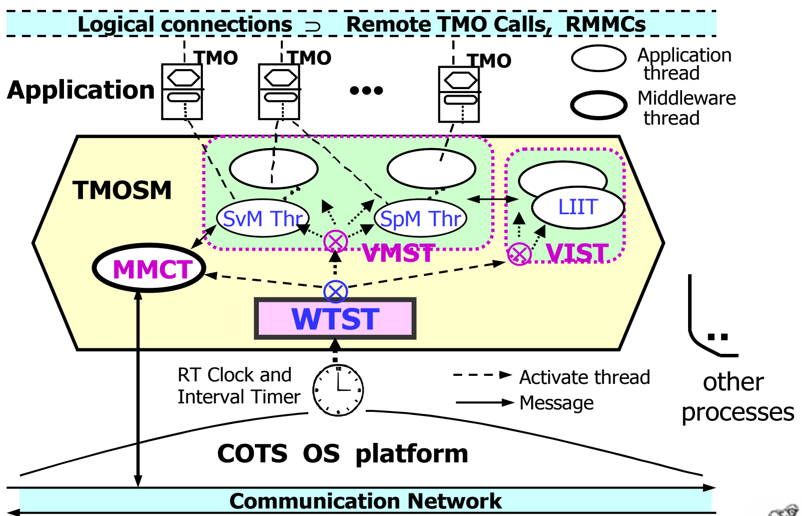
- **Basic concurrency constraint (BCC):**
  - SpM executions not disturbed by SvM executions.
  - Eases design-time guaranteeing of timely services of TMO's



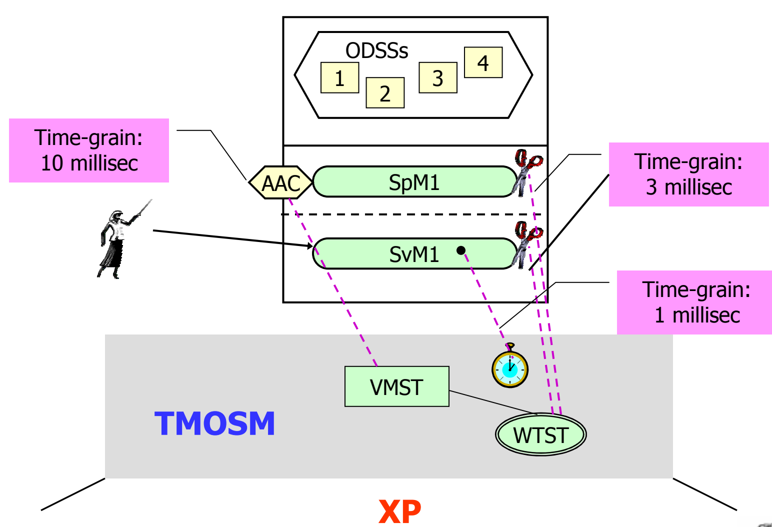
- Pipelining of SvM executions are supported.

UCI
DREAM Lab

TMO Support Middleware on Windows NT/XP:
TMOSM / XP / Socket

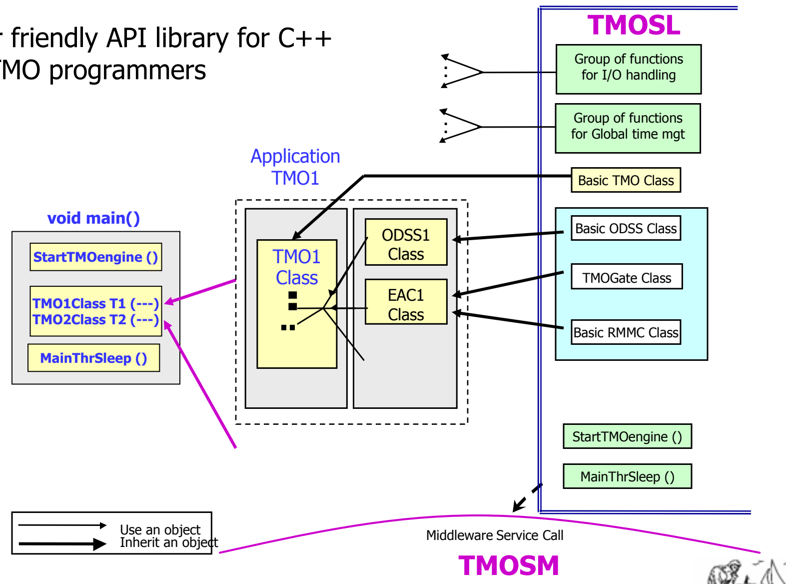


Time-Grains to be used in TMO Programming with TMOSM/XP



TMOSM Support Library (TMOSL)

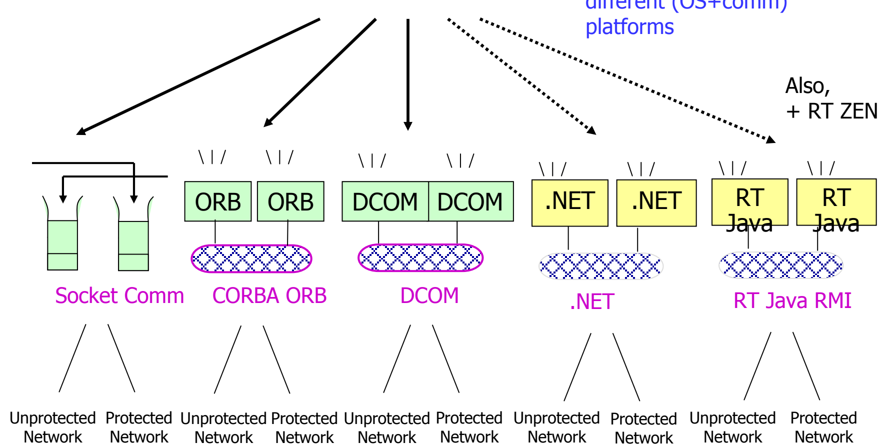
User friendly API library for C++
TMO programmers



Also, formal
education started
at UCI !!

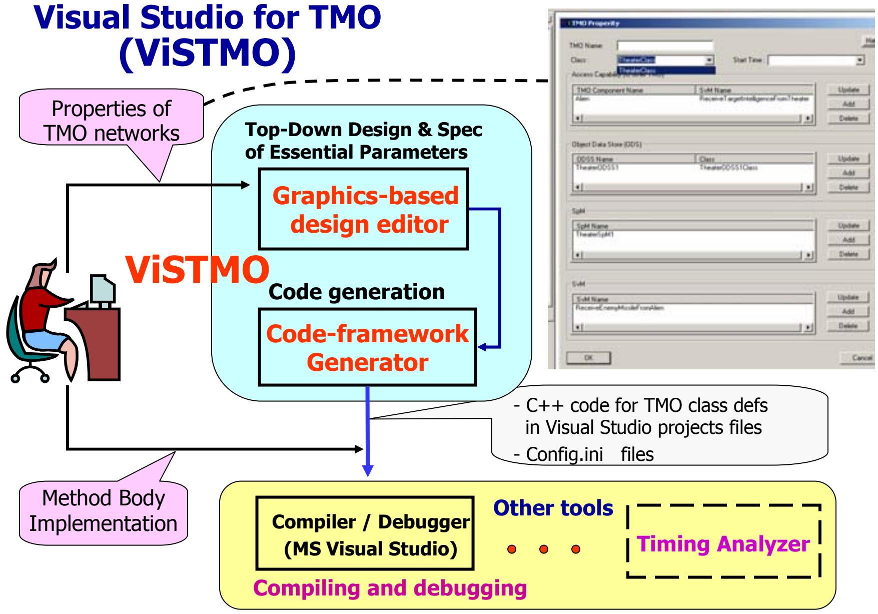
TMOSM
(& TMOSL)

Middleware architecture
easily adaptable to
different (OS+comm)
platforms

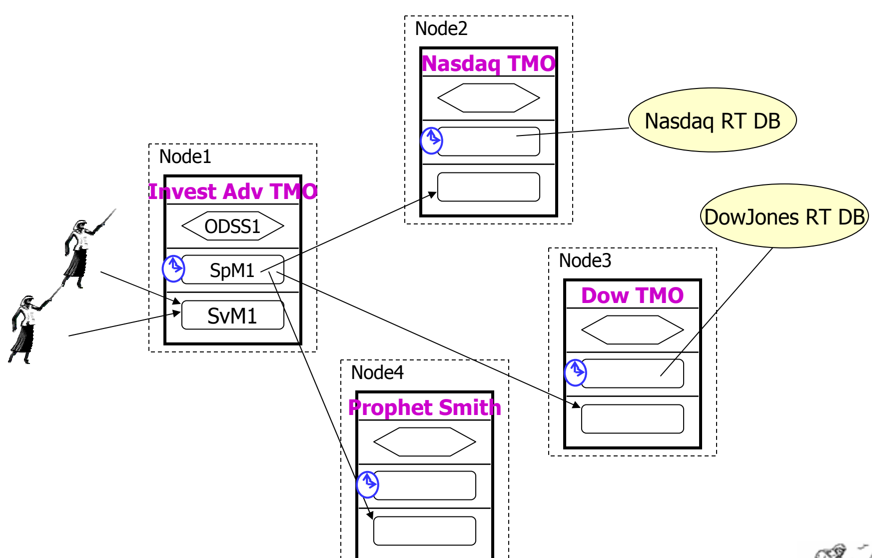


Also, NT / XP --> WinCE, .NET, LInux, and RT Java platforms

Visual Studio for TMO (ViSTMO)



An example of a TMO network



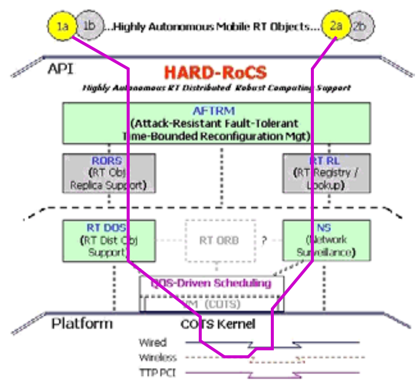
Technical Approach
Exploration of New Design Paradigms

- Time-based coordination of distributed actions (TCoDA)
- High-level programming of time-sensitive distributed actions
- **End-to-end timing analyzability**

= f {

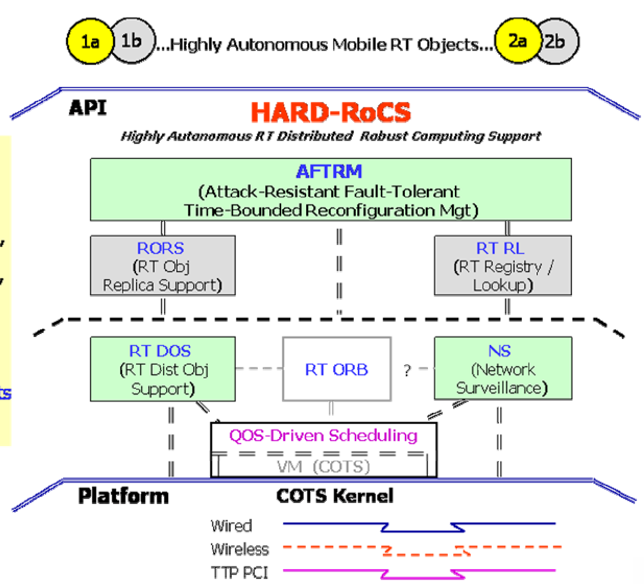
Timing analyzability
at every layer and
in every component

}



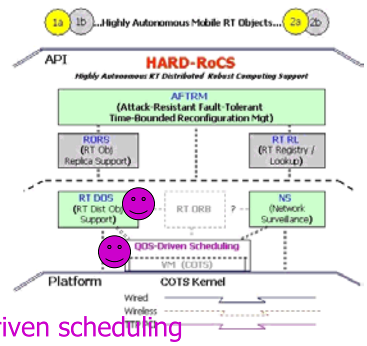
A Middleware Architecture Skeleton - HARD-RoCS

PAP
(Pluggable,
Analyzable,
Parame-
terized)
Components



Technical Approach & Deliverable -
RT DOS (Distributed Object Support)

- Will support design and execution of highly autonomous and yet timeliness-guaranteed distributed computing objects such as TMOs.
- With RT DOS available, time-based coordination of distributed actions becomes an easy job for application developers.



- Will enable QoS (quality-of-service)-driven scheduling & management of distributed resources, thereby making it easier for application developers to produce implementations meeting application timing requirements.
- An API model approximating an idealistic real-time distributed object language which does not require a new compiler, will be developed.

Progress -
RT DOS (Distributed Object Support) (cont)

- QoS-driven multi-level resource allocation
  - Reflects both timing specifications and risk incursion functions (risk value specifications for insufficient QoS's) associated with applications
  - Time budgets and RIFs are both statically and dynamically partitioned in multiple steps/levels.
  - A version of a low-level scheduler was developed and an experimental evaluation produced positive results.
- Clock resynchronization
  - A version that is compatible with the scheduling structure of RT DOS and yields known bounds on the overhead of periodic resynchronization has been developed.
- The RT DOS architecture (TMOSM) has been extended to enable incorporation of new peripheral devices (& drivers) into the underlying platform and use of them in RT objects without requiring modification of the RT DOS each time. A prototype implementation on XP has been upgraded, TMOSM 3.0.
- Another running on iPaq/WinCE has also been produced.

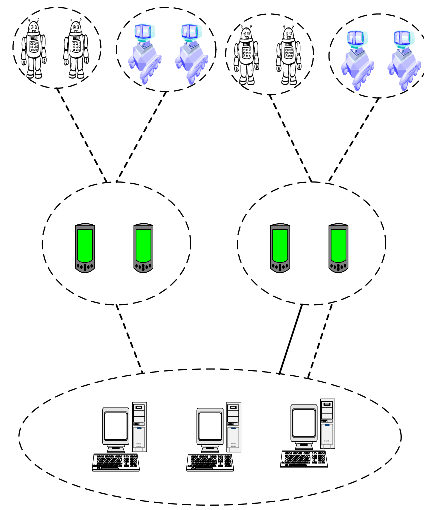
Progress -
RT DOS (Distributed Object Support) (cont)

A small subset of Functionality will be kept.
E.G., TT-function, deadline violation alarm, etc

Functionality is mostly kept

RT DOS - TMOSM

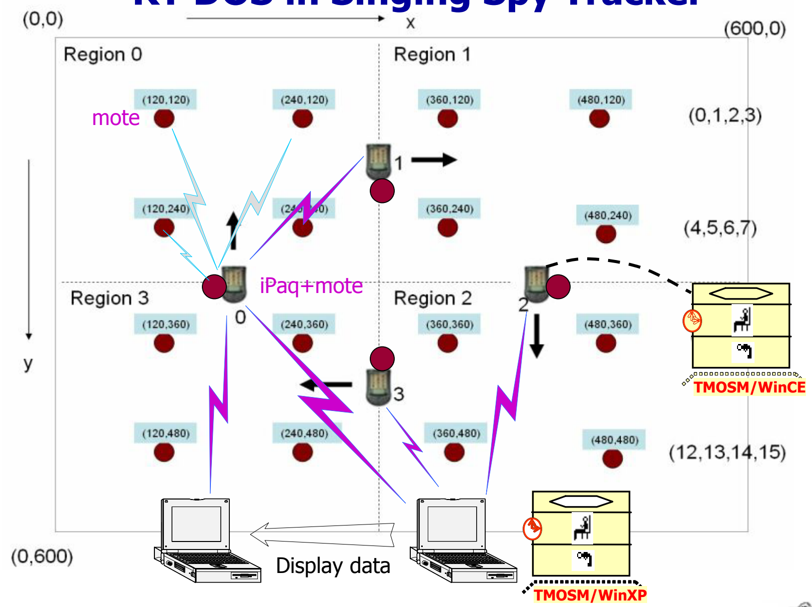
RT DOS
QOS-Driven Scheduling
VM (COTS)



Progress -
RT DOS (Distributed Object Support) (cont)

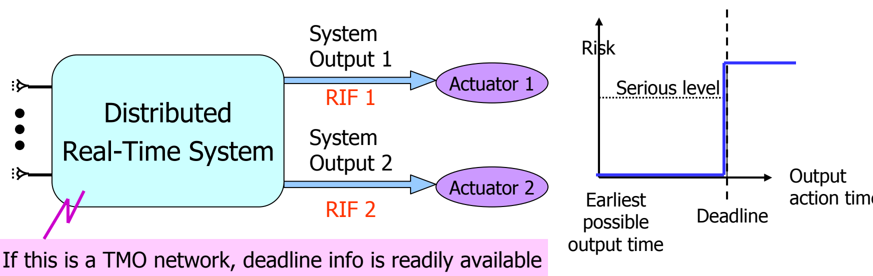
- The API which wraps the services of RT DOS and is an approximation of an idealistic RT distributed object language, TMOSL 3.0, has been established.
  - An earlier version, TMOSL 2.2, along with the corresponding RT DOS, TMOSM 2.2, was used effectively in an undergraduate senior-year course on "Introduction to RT Distributed Programming" at UCI a year ago.
  - The lecture notes and that earlier version (2.2), now called the α **version**, have been tested at several R&D organizations.
  - The β **version** (3.0) is currently used in the course at UCI.
    Plan to distribute this version to many more organizations during the winter quarter (Jan – March) of 2003.
- Plan to port this onto Linux.

RT DOS in Singing Spy Tracker



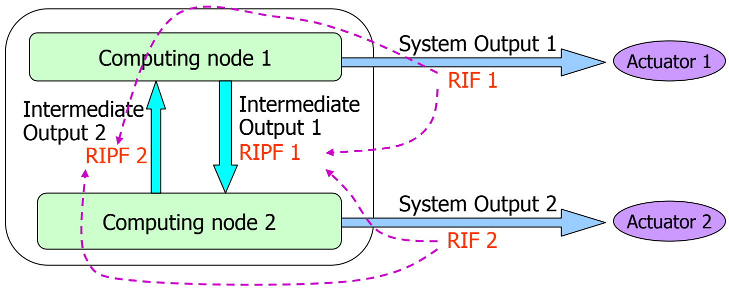
- **Sensor values are collected and sensors and relays are coordinated in a timely manner in spite of various resource failures.**

Progress - RT DOS & QoS-Driven Scheduling
Risk Incursion Function (RIF)



- Risks: Damaging impacts of deadline violations and/or other QoS losses on the application mission.
- **RIF** (Risk Incursion Function, a.k.a., Benefit Loss Function)
  := Relation (Loss in timed value accuracy of each output action,
  Potential application damage)
  := Relation (QoS loss, Risk)

Risk Incursion Potential Function (RIPF)

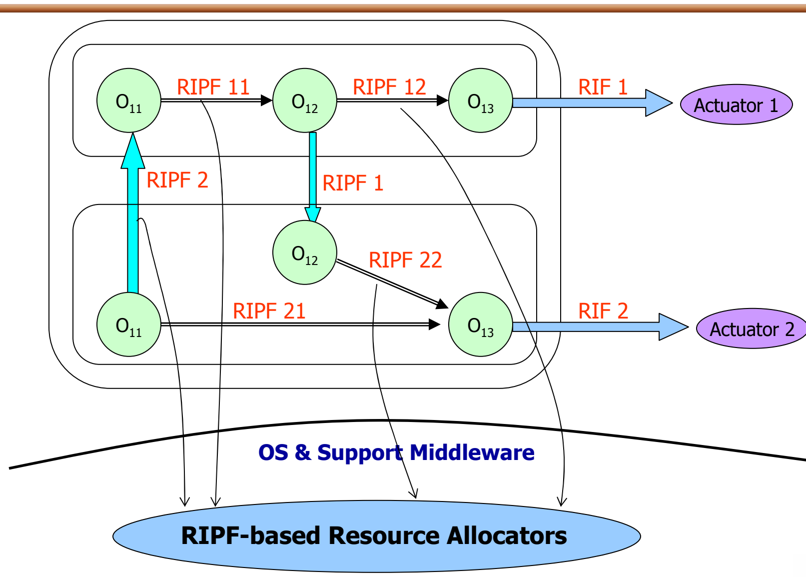


System-level RIF and Derived RIF (= RIPF)

RIPF (Risk Incursion Potential Function) = Derived RIF
= Relation (Accuracy loss in intermediate output, Potential risk)

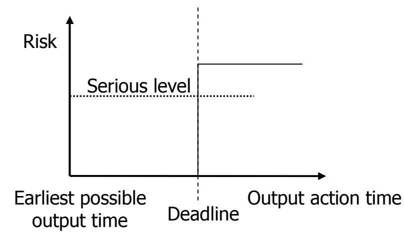
- Derivation of RIPFs :=
Allocation of Time Budgets & Budget-Overrun Penalties

Risk Incursion Potential Function (RIPF)

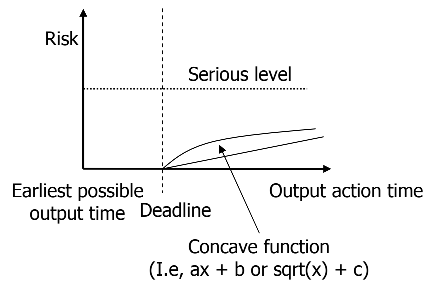


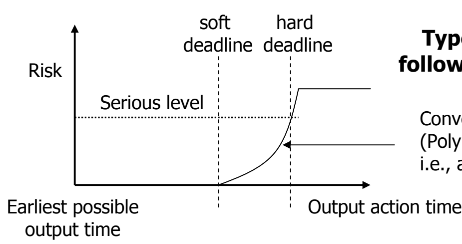
RIF (RIPF) Examples

Type I: Hard Deadline



Type II: Soft Deadline



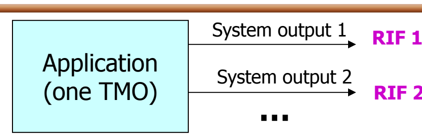


Type III: Soft deadline followed by a hard deadline

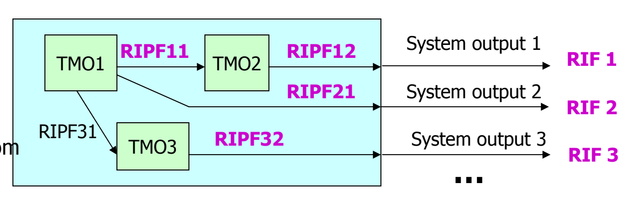
Convex function
(Polynomial function,
i.e., $ax^3 + bx^2 + cx + d$)

Top-down Multi-step Allocation of Time Budgets & Budget-Overrun Penalties

Entire Application viewed as one TMO

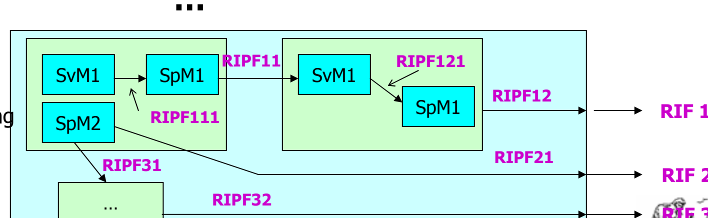


The TMO is divided into multiple TMOs, The RIPFs are also derived from the RIFs



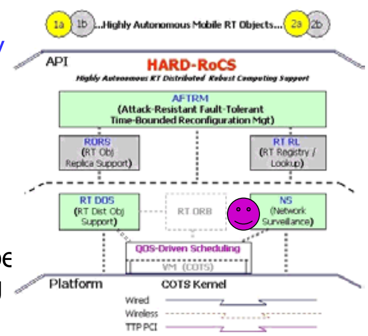
...

Elaborated as a TMO network (basic scheduling unit is SxM supported by a thread)



Technical Approach & Deliverable -
NS (Network Surveillance)

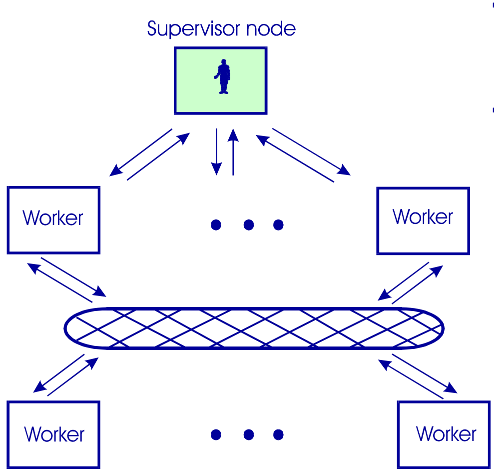
- Basically a (partially or fully) decentralized mode of detecting faulty and repaired status of distributed computing & comm components.
- An important metric: **Detection latency bound**
- Basic techniques that exploit time-based synchronization principles will be used as initial cornerstones in building up this technology base :



1. The supervisor-based network surveillance (SNS) scheme (by Kim) for network surveillance in point-to-point network based systems;
2. The time-triggered protocol (TTP) scheme (by Kopetz) for network surveillance in bus-LAN based systems.

Progress -
NS (Network Surveillance)

Supervisor-based NS (SNS) scheme



Duties of the elected supervisor

1. Performs basic duties of workers
2. Judges the health conditions of worker nodes and links among them
3. Informs relevant nodes of detected faults

Basic duties of workers

1. Exchanges heartbeats with neighbors at frequency f_1
2. Sends a suspicion report (on the health of a neighbor) to the supervisor

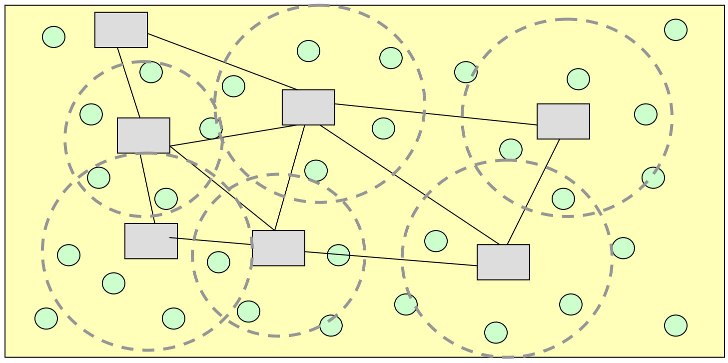
Special duties of the supervisor's neighbors

1. Makes a group decision on the health of the supervisor
2. Participate in a new supervisor election

- One prototype implementation on XP has been produced.

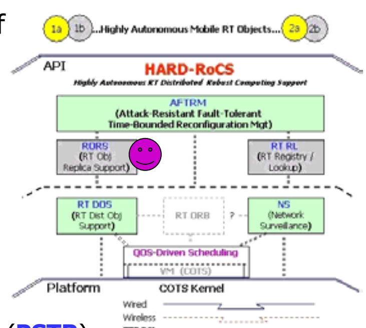
Progress -
NS (Network Surveillance)

- A hierarchical extension of SNS has been developed.
  - A system may use many desktops, pocket PCs, motes, etc.



- A version is being incorporated into the Singing Spy Tracker demo involving notebooks, iPaqs, and motes.

Technical Approach & Deliverable -
RORS (RT Object Replica Support)



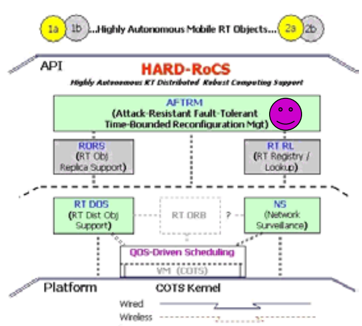
- Will support fault-tolerant execution of RT objects with tightly bounded time costs.
- Important metric: **Recovery time bound**
- Techniques exploiting **synchronized active replica** construction principles
  - Will be used as initial cornerstones in building up this technology base :
    1. The Primary-Shadow TMO Replication (**PSTR**) scheme (by Kim) which is an extension of the DRB (distributed recovery block) scheme for fault tolerance in RT object-based systems. ;
    2. The Time-Triggered Architecture (**TTA**) scheme (by Kopetz) for replication with voting in bus-LAN based systems.
- Will also support passive replicas: **PPTR** (Primary-Passive TMO Replication).
- Location-aware replicas will be supported.

Progress -
RORS (RT Object Replica Support)

- A (quick-and-dirty) prototype implementation of a subset of PSTR has been implemented.
- A C3 distributed computing testbed, which is called CAMIN (Coordinated Anti-Missile Interceptor Network) and based on a LAN of 5 PCs, has been used to experiment with the prototype implementation.
- Full PAP versions of the PSTR support component and the passive replica support component are under development.

Technical Approach & Deliverable -
FTRM
(Fault-tolerant Time-bounded Reconfig Mgt)

- Will be layered on layered on RT RL / RT DOS / RT platform.
- Will support fault-tolerant time-bounded reconfiguration of mobile servers.
- Temporary and permanent failures of server objects (incl. both application objects and middleware objects, e.g, RT RL), server stations, and comm infrastructure components will be handled.



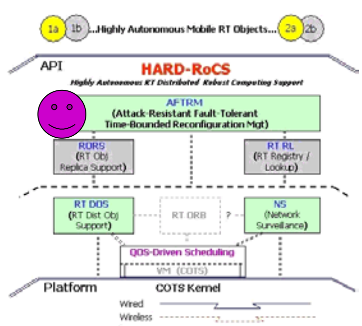
- A glue among NS, RORS, and RT RL
- **Adaptive configuration mgt**:
  - Aimed for maintaining the minimum required quality of services (QoS's) of critical components for longest possible periods.

Progress - FTRM (Fault-tolerant Time-bounded Reconfig Mgt)

- A high-level framework of an RIF-driven FTRM has been established.

Technical Approach & Deliverable - AFTRM (Attack-resistant Fault-tolerant time-bounded reconfig mgt)

- An extension of the FTRM middleware
- Will possess the additional capability & mechanisms for enabling applications to resist
  - not only against random failures of computing and communication components
  - but also against security attacks
  - via authentication and time-bounded reconfiguration of mobile servers.
- The security attacks to be considered include denial-of-service attacks.



- Will be capable of fault-tolerant authentication.

Progress - AFTRM

- A high-level framework for Hierarchical Authentication has been established.
  - Contribution of Bharat Bhargava's research group at Purdue
- Incorporation of this scheme into the RT RL (Registry-Lookup) support component is under way.
- Resource allocation schemes which disable denial-of-service attacks are under study.

But

- **I should perhaps forget this.**

  Instead, focus on completing ROAFTS (= HARD-RoCS – Security Mechanisms) & writing a monograph.

Summary

- I have had a chronic disease of underestimating how long it takes to demonstrate a small idea !