Evolving the CORBA standard to support new distributed real-time and embedded systems

Tom Bracewell
Senior Principal Software Engineer
Raytheon Integrated Defense Systems
Sudbury, MA. / (978) 440-2539
bracewell@raytheon.com

Priya Narasimhan
Asst. Professor of ECE and CS
Carnegie Mellon University
Pittsburgh, PA / (412) 268-8801
priya@cs.cmu.edu

Raytheon

What's Needed?

- **How does the CORBA standard need to evolve to meet the needs of *large* distributed real-time and embedded systems ?**
- **Examples**
  - total ship computing – a system of systems
  - surface radar systems – a large DRE system
- **Will the CORBA standard address the needs of real time and embedded systems of the near future ?**
- **We need to be able to "pick standards, not products"**

Will CORBA support embedded systems?

- **A computer (and its SW) is considered embedded if it is an integral component of a larger system and is used to control and/or directly monitor that system, using special hardware devices." - IEEE**

- **Is it embedded?**
  - program that understands physics, a PDA - no
  - disk controller, a microprocessor controlling cell phone - yes
  - radar computing system – yes, nowadays

- **Where's the boundary - focus on intent**
  - If it's embedded, it's controlling special hardware
  - If I shoot this component, will the special hardware stop working?

- **New, large DRE systems create new challenges for CORBA**

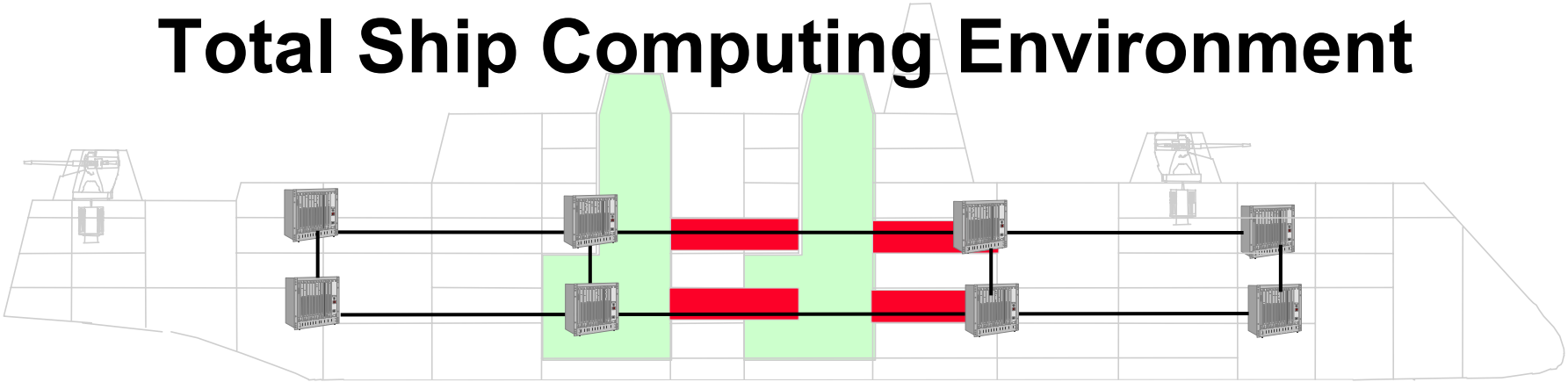
Issues driving new systems

- **End-to-end QoS, predictability, scalability**
- **Dependability, survivability, vulnerability, recoverability**
  - vulnerability 1-P(maintain mission readiness after *n* hits)
  - recoverability P(restore mission capability in timeframe T)
- **Resource management**
  - application needs (e.g real-time)
  - mission needs (e.g. fault tolerance)
  - component-level redundancy in both HW and SW
  - high performance in less weight, volume and cost

More Issues driving new DRE systems

- **Work with multi-level security (MLS)**
- **Must lower the cost of developing, verifying and fielding SW**
- **"Pick standards not products"**
  - Standards bodies important
    - define standards
    - conformance tests
    - performance metrics

Total Ship Computing Environment

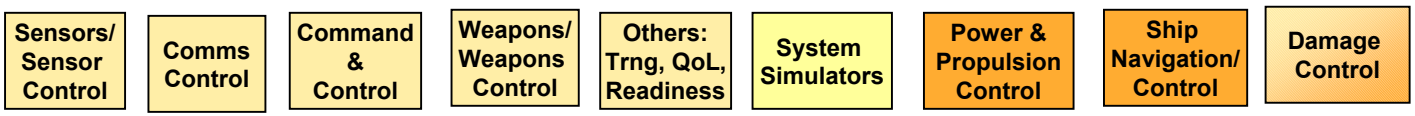


- **Distributed ship-wide for both tactical and non-tactical uses**
- **COTS-based open-system network of multiprocessor systems**
- **Total ship approach to survivability and performance**
  - Manage vulnerability and recoverability
  - Monitor resource configuration and performance
  - Provide continuous reconfiguration recommendations
  - Adaptive, fault tolerant, self-reconfiguring - and real-time
- **Supports network-centric warfare**
- **MLS environment, intrusion tolerant**
- **Supports rapid, cost-effective application development**

TSCE Open System Architecture

Each layer uses commercially defined common open standards

- **Diverse applications**



- **Domain services, e.g.**

- **<u>Standards-based middleware, e.g.</u>**

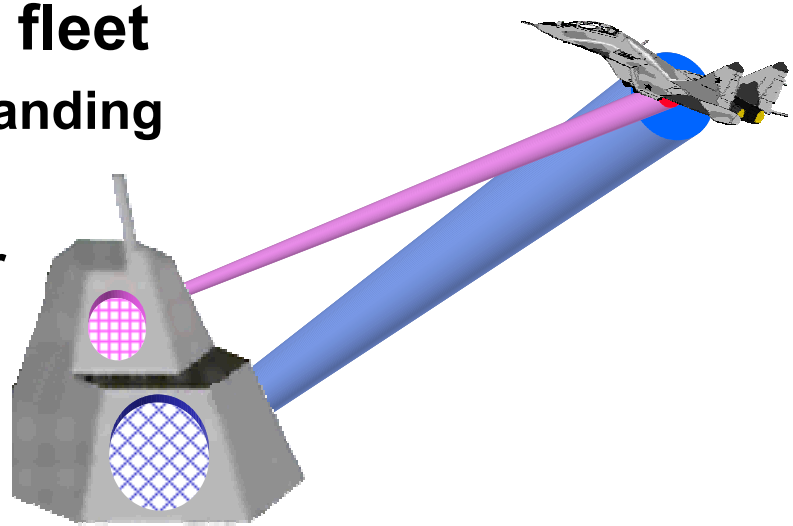
- **COTS real-time operating systems and computers**

Raytheon

Surface Radar Product Line

- Multifunction Radar (MFR)
- Volume Search Radar (VSR)

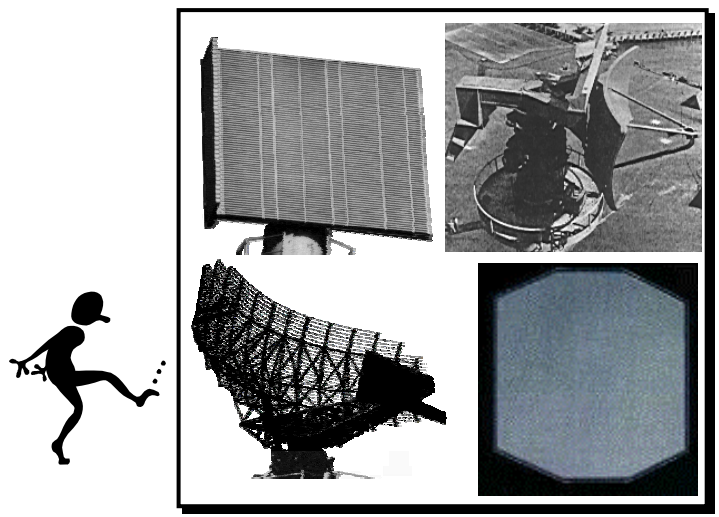
- MFR meets all horizon search and fire control requirements of 21st-century fleet
  - support precision approach & landing
  - support missile engagements
  - detect stealthy targets in clutter
    - cruise missiles, periscopes
    - horizon aircraft, ships



Raytheon

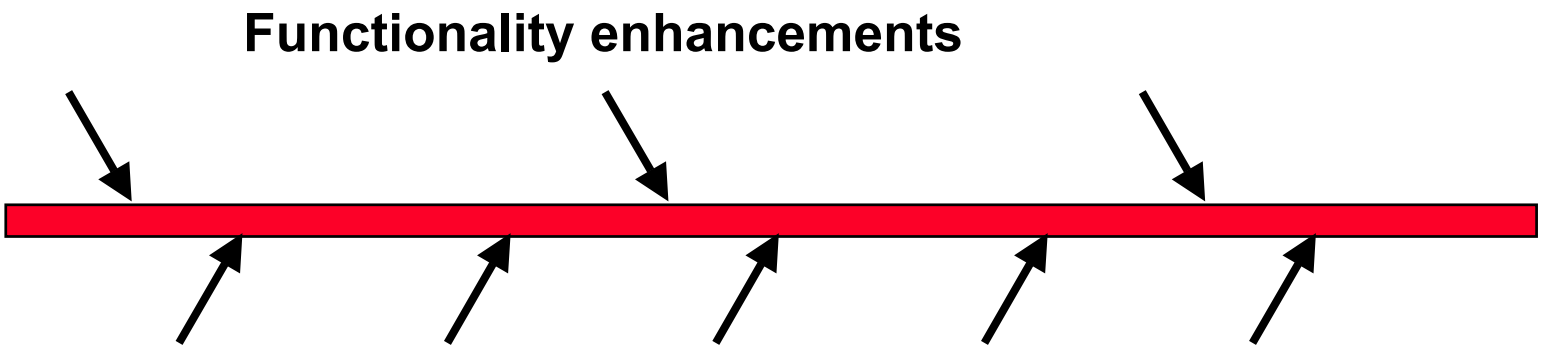
Surface Radar Systems

- **VSR supports aircraft, long range target tracking**
- **DBR includes MFR and VSR**
- **Bottom line:**
  - 160+ high-end processors
  - DRE system uses COTS HW, OS
  - SW fault tolerance important
  - less cost per functionality
  - eliminates special equipment



Raytheon

Managed Product Evolution



Integrate with CORBA standard roadmap

CORBA: Support Two Worlds

Issue	Real-time System	Fault-Tolerant System
Events	*a priori* knowledge of events is required	faults might occur at any time
Ordering of Operations	meet task deadlines	preserve replica consistency
Determinism	"bounded predictable temporal behavior"	"coherent state across replicas for every input" (FT-Determinism)
Multithreading	used for concurrency and efficient task scheduling	FT-Determinism prohibits multithreading
Timers	local timeouts and timer-based mechanisms used	FT-Determinism prohibits using local processor time
Faults	faults include timeouts	fault recovery takes time

RT FT CORBA Issues

- End-to-end predictability (latency, priority)
- Replica consistency (determinism, message guarantees)
- Determinism (RT predictability, FT reproducibility)
- Multithreading (deadlines, nondeterminism)
- Clock synchronization and global time
- Ordering of events (deadlines, nondeterminism)
- Fast fault detection and isolation
- Predictable recovery times
- Application-transparent fault tolerance

- This is not for the application layer or the OS to handle!

What do we need RT FT CORBA to do

- Tolerate common faults
  - **crash, communication, timing, partitioning faults**
- Order tasks to meet replica consistency and deadlines
- RT FT Scheduler and resource manager
- Policy-driven fault recovery
- Bounded fault detection and recovery times
  - **Plan for worst-case fault recovery**
- Support proactive dependability, software rejuvenation

We Need Offline Tools Too

- Offline tools complement runtime capabilities
- Advisor takes guesswork out of configuring for reliability
- Hazard oracle to detect and reduce non-determinism

OMG Actions

- Evolve CORBA standard to address system needs
- Establish compliance metrics
- Develop/own compliance tests