Investigating Lightweight Fault Tolerance Strategies for Enterprise Distributed Real-time Embedded Systems

Jaiganesh Balasubramanian
jai@dre.vanderbilt.edu
www.dre.vanderbilt.edu/~jai

Dr. Aniruddha Gokhale
gokhale@dre.vanderbilt.edu
www.dre.vanderbilt.edu/~gokhale

Dr. Douglas C. Schmidt
schmidt@dre.vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Dr. Nanbor Wang
nanbor@txcorp.com
www.cs.wustl.edu/~nanbor

Vanderbilt University
Nashville, Tennessee

Tech-X Corporation
Boulder, Colorado

This research is sponsored by Lockheed Martin, BBN Technologies, and Raytheon under contracts to Vanderbilt University and by NSWCDD under contract to Tech-X Corporation (N65538-06-M-0029)

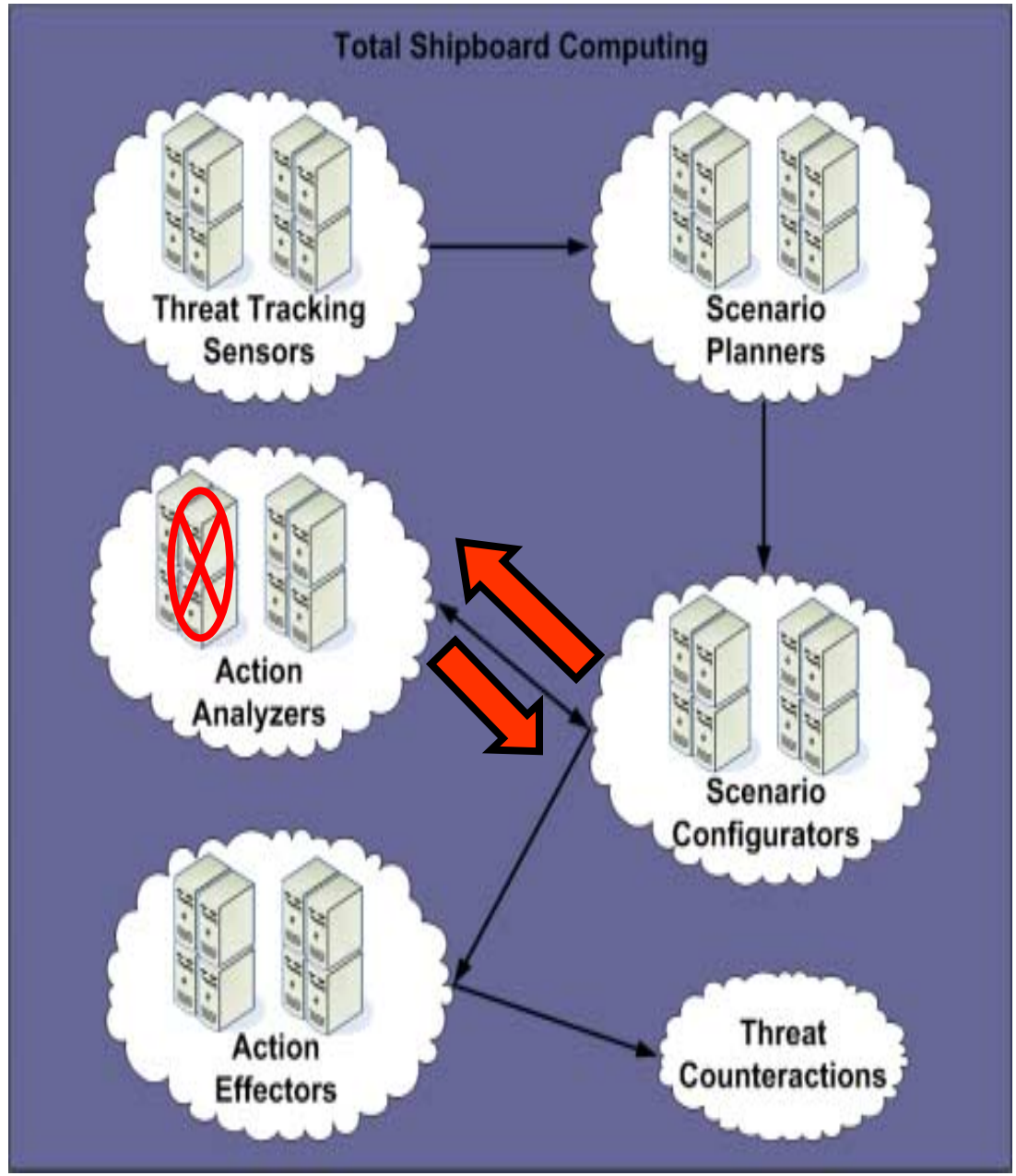
Enterprise DRE Systems Characteristics

- Network-centric, dynamic, very large-scale “systems of systems”
- Stringent *simultaneous* QoS demands, e.g., “never die,” time-critical, etc.
- Highly diverse, complex, & increasingly integrated & autonomous application domains
- Distributed Object Computing middleware used to design and develop enterprise DRE systems
  - support for highly available systems
    - e.g., FT-CORBA
  - end-to-end predictable behavior for requests
    - e.g., RT-CORBA

Goal is to provide real-time fault tolerance for enterprise DRE systems

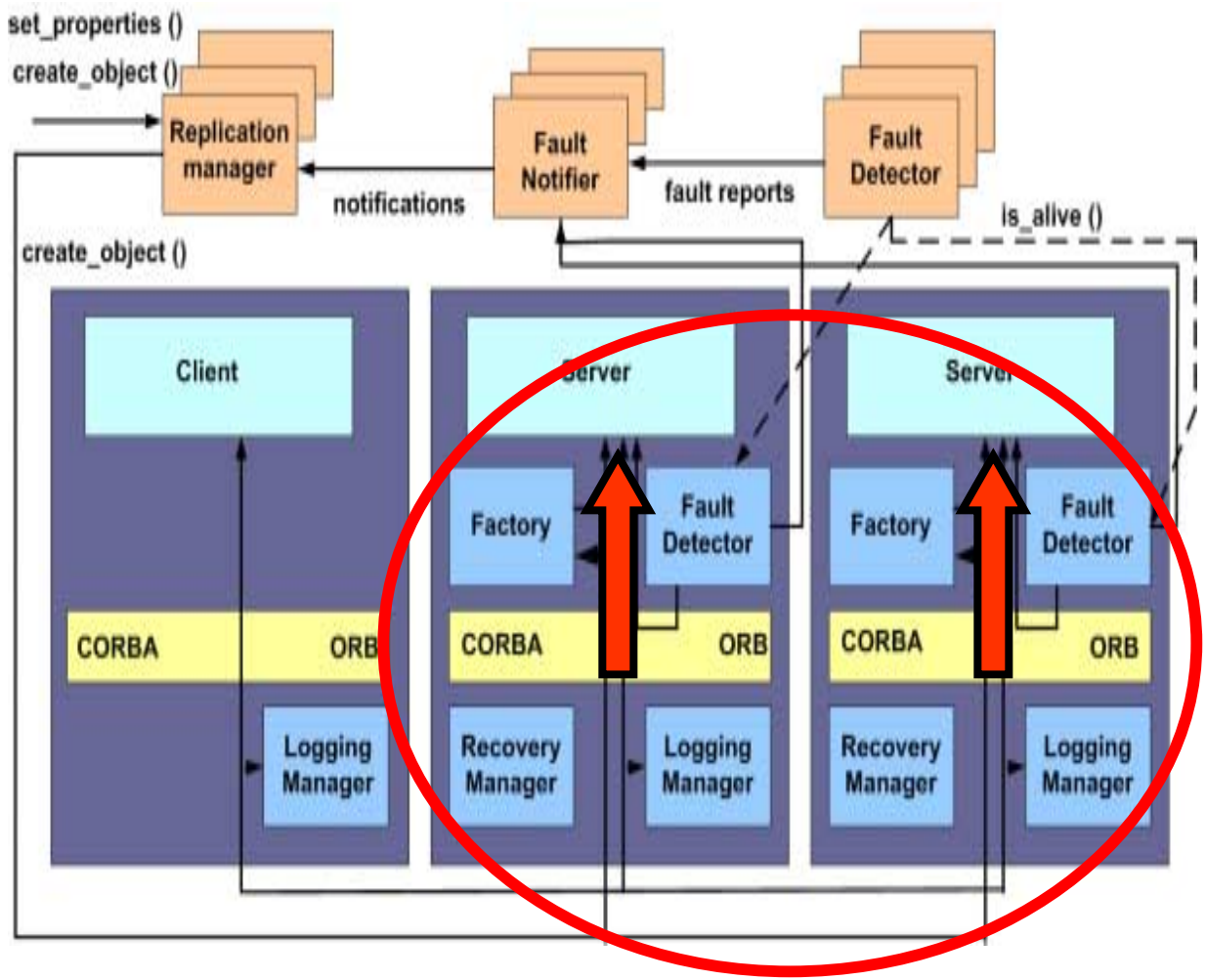
Challenges in Developing RT-FT Enterprise DRE Systems

- Meeting real-time deadlines even in the presence of failures
  - bounded times for fault detection and fault recovery
  - bounded times for dynamic reconfigurations because of hardware and software failures
- Satisfying QoS demands in the face of fluctuating and/or insufficient resources
  - degraded QoS for certain applications in the presence of hardware and software failures
  - dependable upgrades, downgrades & updates



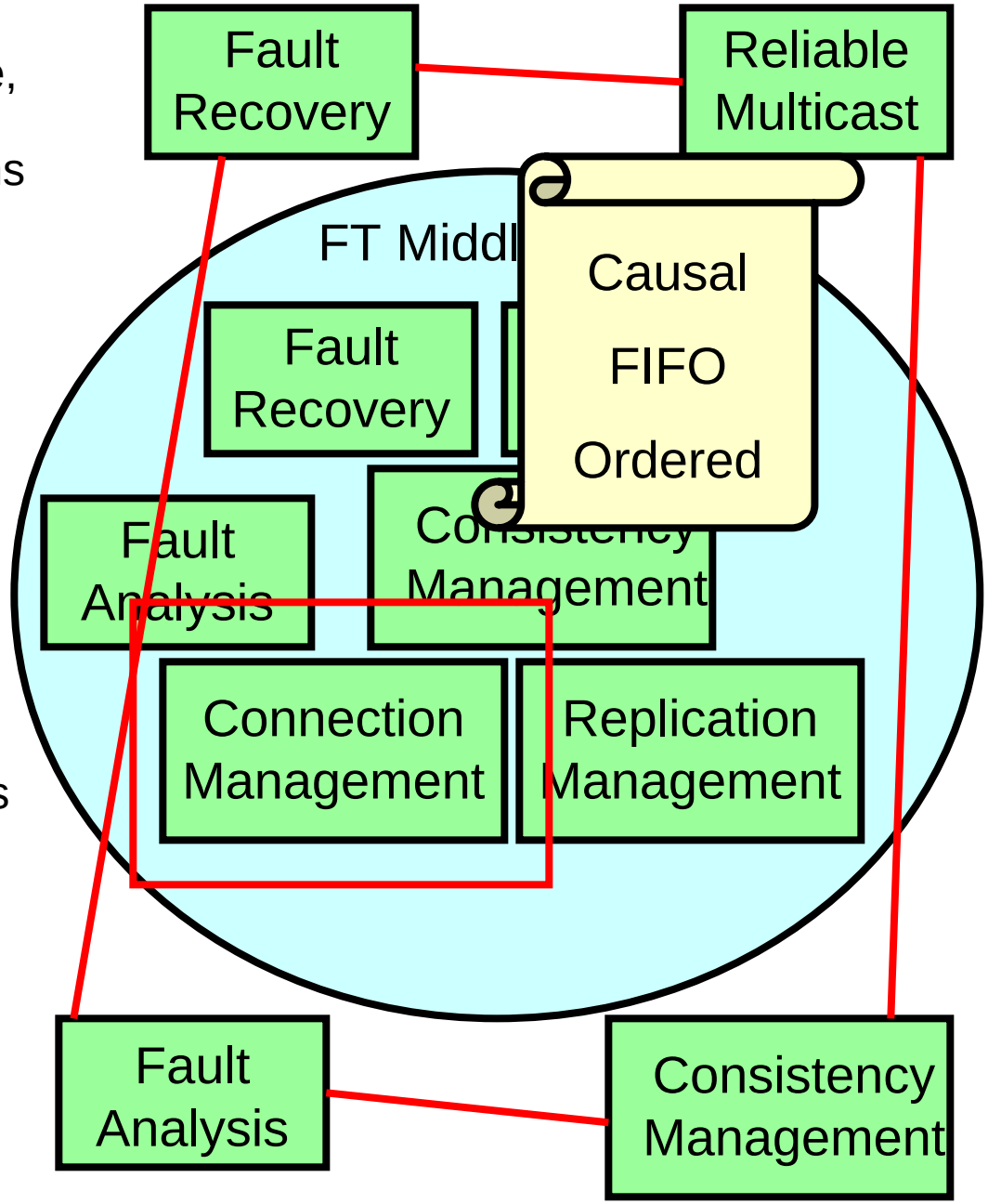
Problems in FT-CORBA Solutions for Enterprise DRE Systems

- Traditional FT-CORBA systems provide a one-size-fits-all solution
  - excessive overhead for systems with real-time requirements and resource constraints
  - overly complex, difficult to integrate
- Unpredictable and expensive replication and fault detection strategies
  - no bound on TCP connection failure detections
  - excessive overhead with active replication on totally ordered reliable group communication as well as state synchronization
- Semantic incompatibility between FT-CORBA and RT-CORBA solutions
  - unacceptable latency and performance overheads inherent in vanilla FT-CORBA solutions
  - adversely impacts predictability



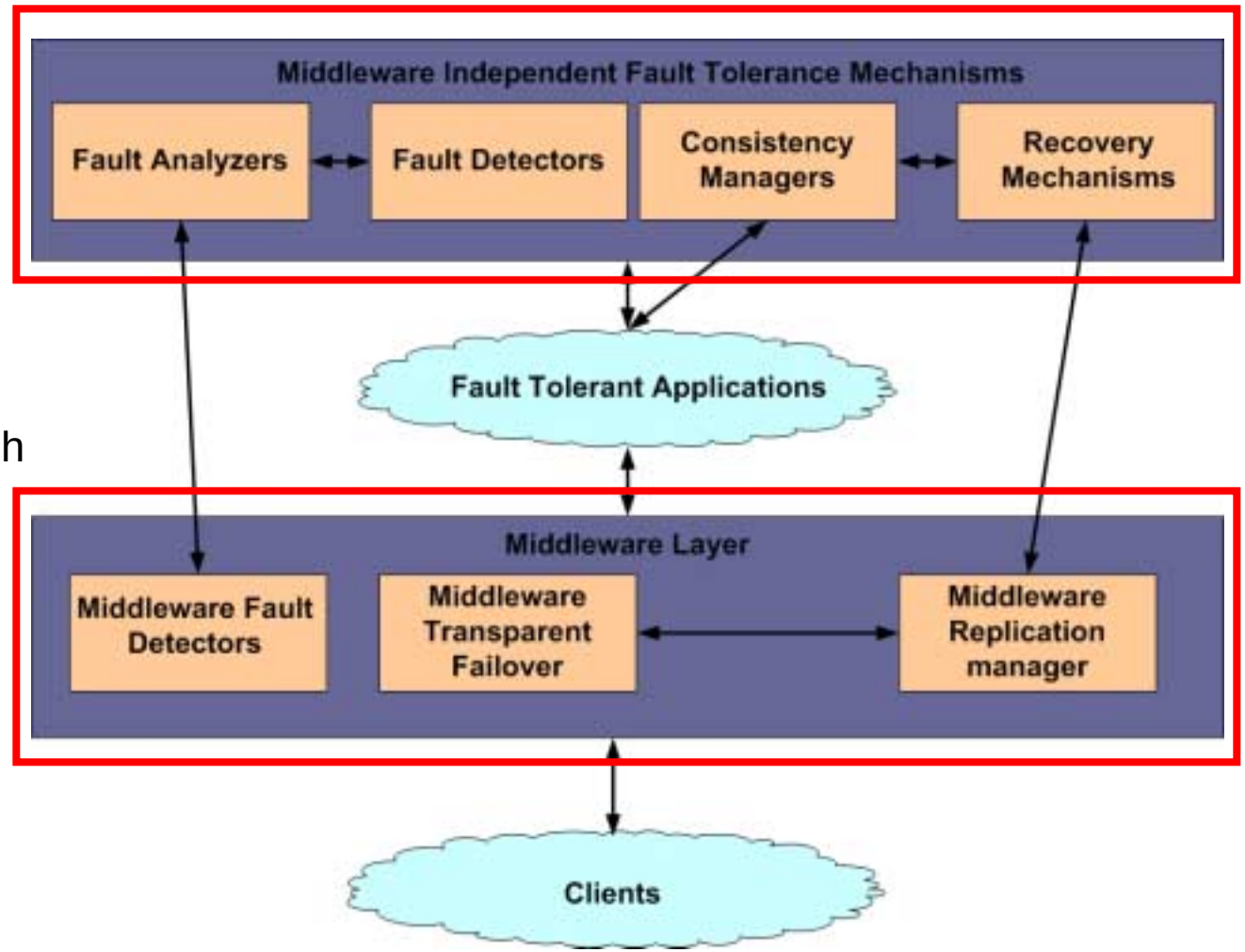
Possible Lightweight Fault Tolerance Approaches

- Decoupling of different FT-specific functionalities from the middleware, so that the middleware can be integrated easily with other systems
  - allows integrating well known fault tolerance techniques into the system
  - move away from point solutions
- Integration of the desired fault tolerance infrastructure elements with RT-CORBA mechanisms to improve predictability of fault tolerance functionalities
- Semantic configuration of fault tolerance infrastructure based on understanding of application needs
- Investigation of interoperable protocols for interactions with external systems
- Microkernel architecture applied to fault tolerant systems



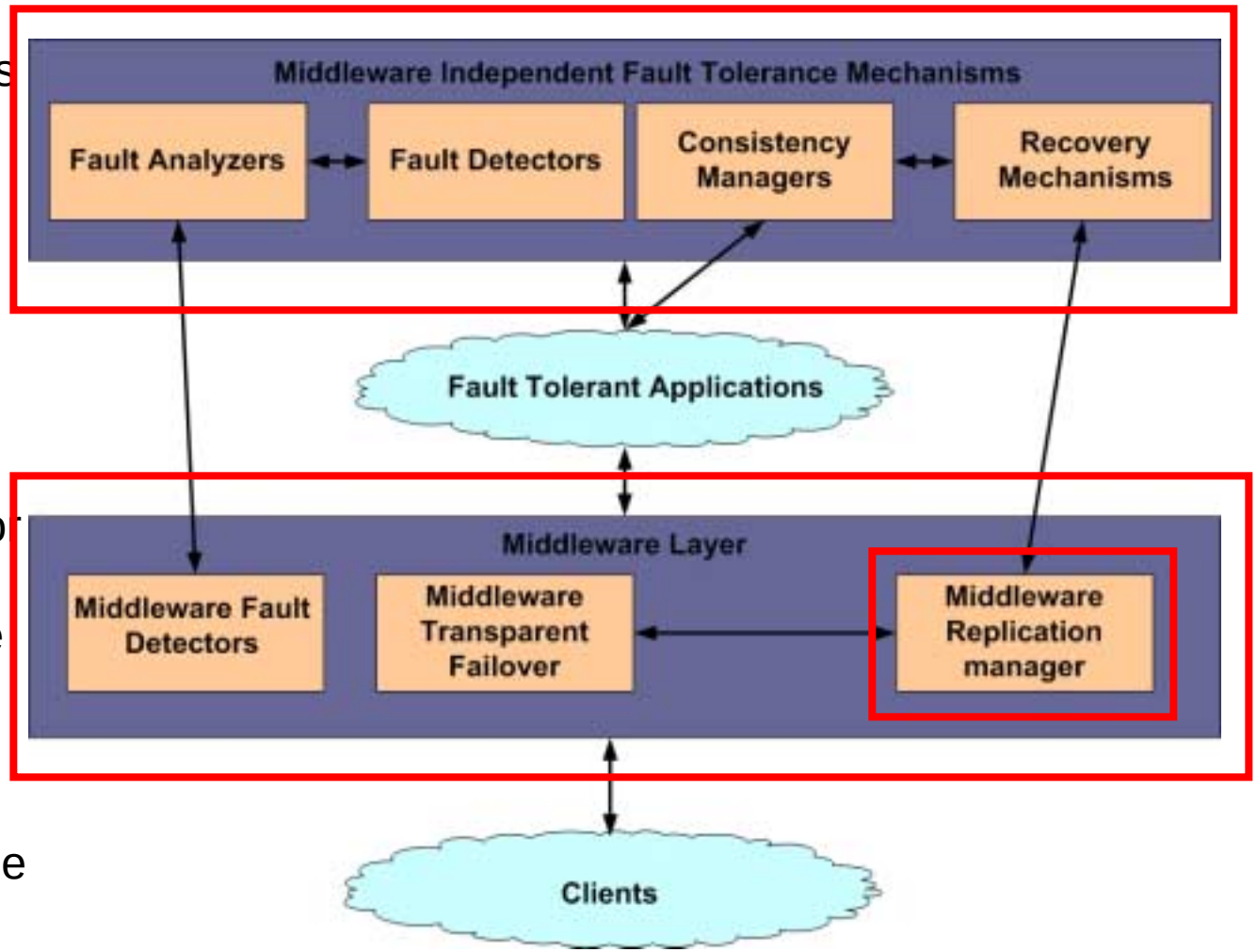
OMG RFP for Lightweight RT Fault Tolerance (1/2)

- A new RFP issued by OMG in the OMG Technical Meeting at Boston, June 2006
  - requires RT FT solutions
- Separation of functionalities from the middleware that vary depending upon the operating scenarios, such as
  - consistency management
  - fault analysis, containment and recovery
- Middleware works collaboratively with external fault tolerance mechanisms providing fault detection, replication management and transparent failover.



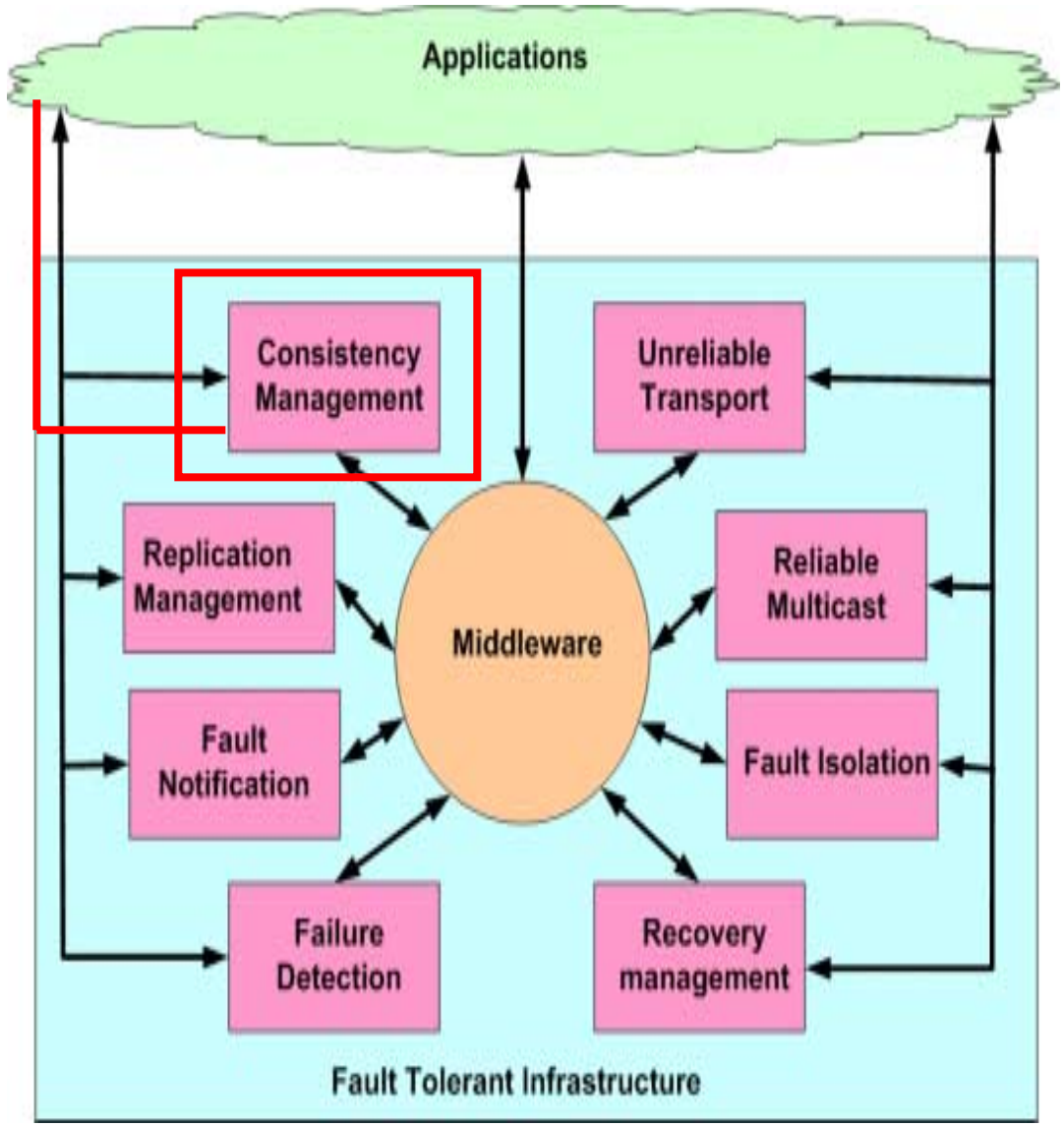
OMG RFP for Lightweight RT Fault Tolerance (2/2)

- Support for both active and passive replications
  - explore weak consistency models for active replication, thereby relaxing determinism requirements
- Support for providing a RT-CORBA mapping for the FT infrastructure elements as well as the application
  - enforce priorities on invocations during failure and failure-free situations
  - avoid priority inversions in recovery operations



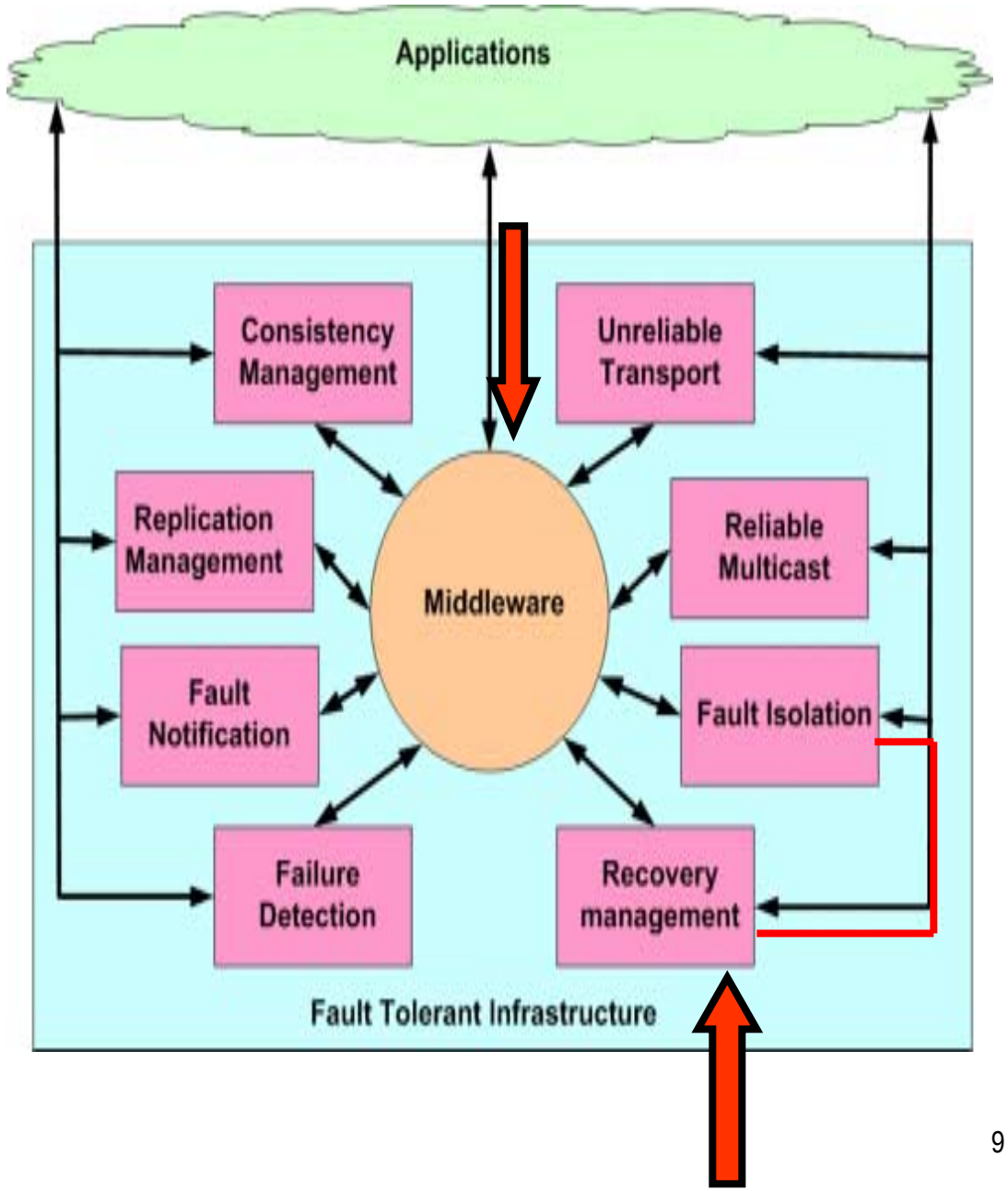
Approaches to Lightweight RT FT (1/2)

- Identification of fault tolerance functionalities that can be better provisioned by external mechanisms
  - ways to configure those mechanisms to provide different QoS to applications
  - e.g. causal, FIFO, reliable guarantees on group communication
- Identification of uniform interfaces and strategy patterns so that applications and the interfaces they access need not change but the mechanisms they access can keep changing



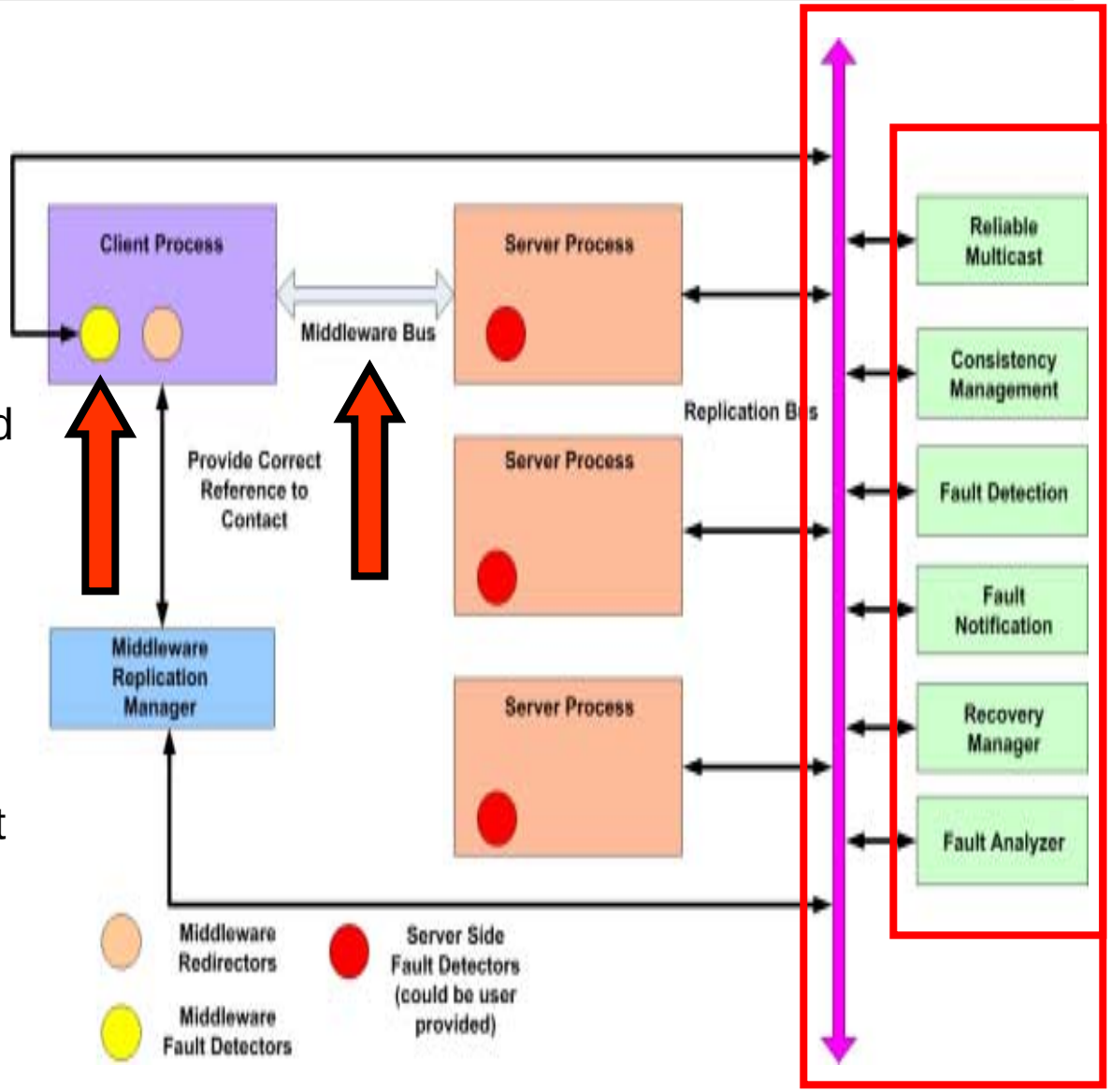
Approaches to Lightweight RT FT (2/2)

- Identification of interfaces for integrating external fault tolerance mechanisms
  - e.g. interfaces between the fault analyzers and recovery managers
- Identification of RT-CORBA mappings for consistent scheduling mechanisms for both the middleware as well as the external fault tolerance mechanisms



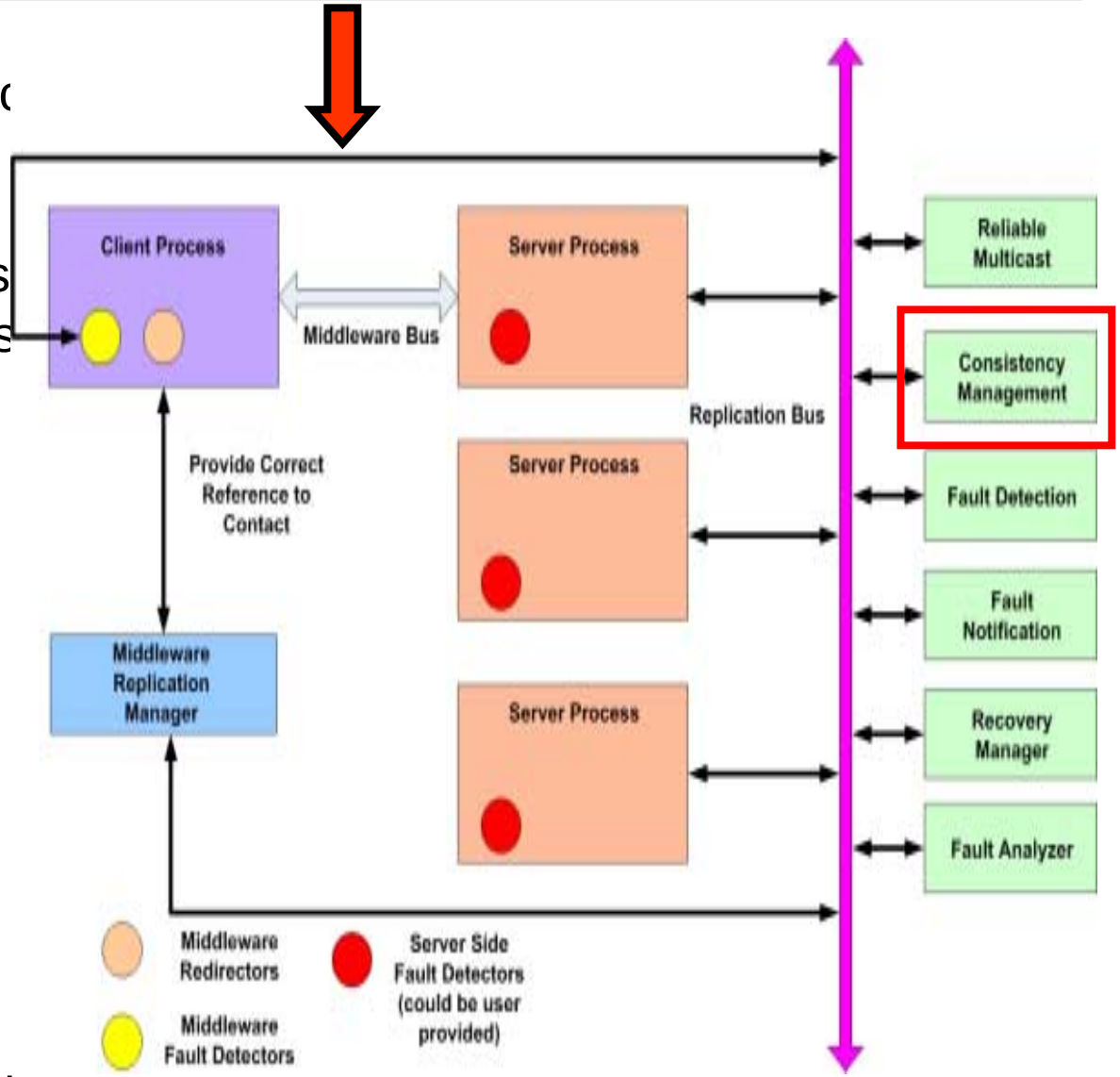
An Example Architecture (1/2)

- Middleware is responsible for handling communication between the client and the server applications
- Common fault tolerance functionalities decoupled from the middleware and configured appropriately
- Interoperable protocols like DDS can be used as a means of communication amongst external fault tolerance mechanisms
  - e.g. fault detectors and fault analyzers
- Middleware provides certain fault detectors that can help faster failure detections
  - e.g. TCP timeout detections



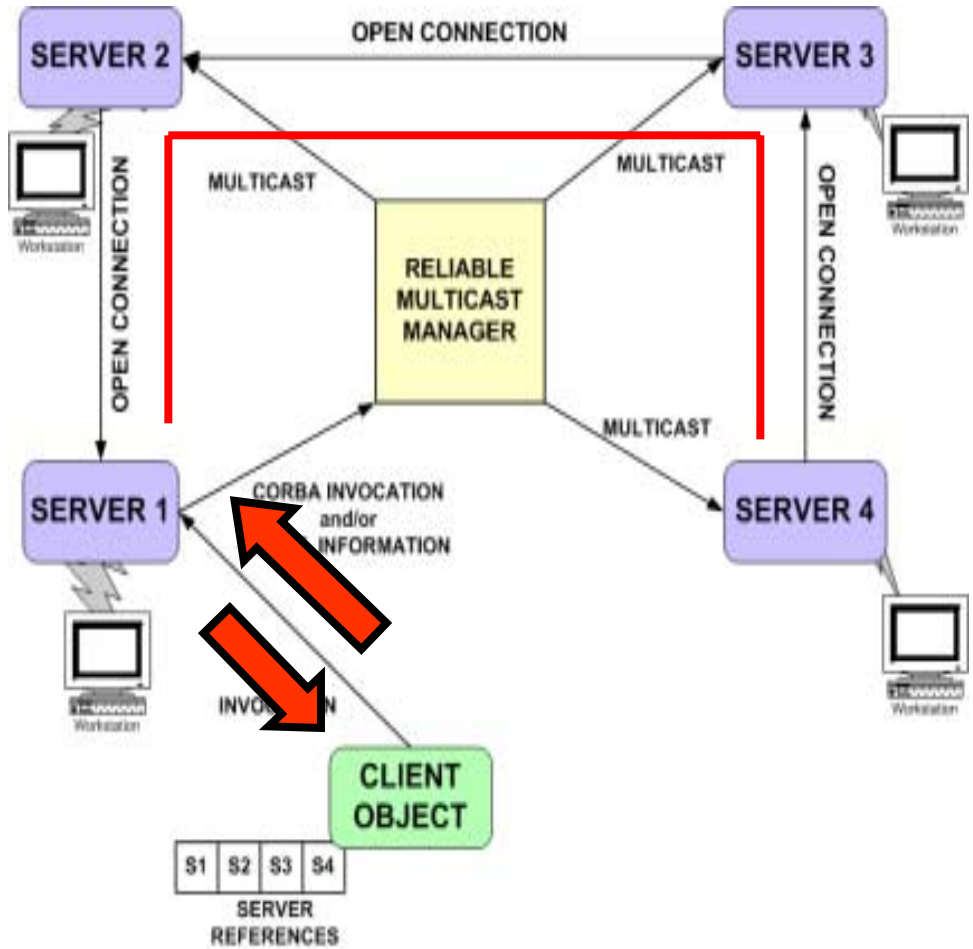
An Example Architecture (2/2)

- Middleware needs to interac with the external mechanisms to provide failure detections as well as to find out the current status of the replication group
- Application developer can configure a fault tolerance functionality that is most suited for the application needs
  - e.g. configuring a consistency mechanism that fits the application needs
  - middleware mechanisms works with the configured functionality to ensure real-time QoS



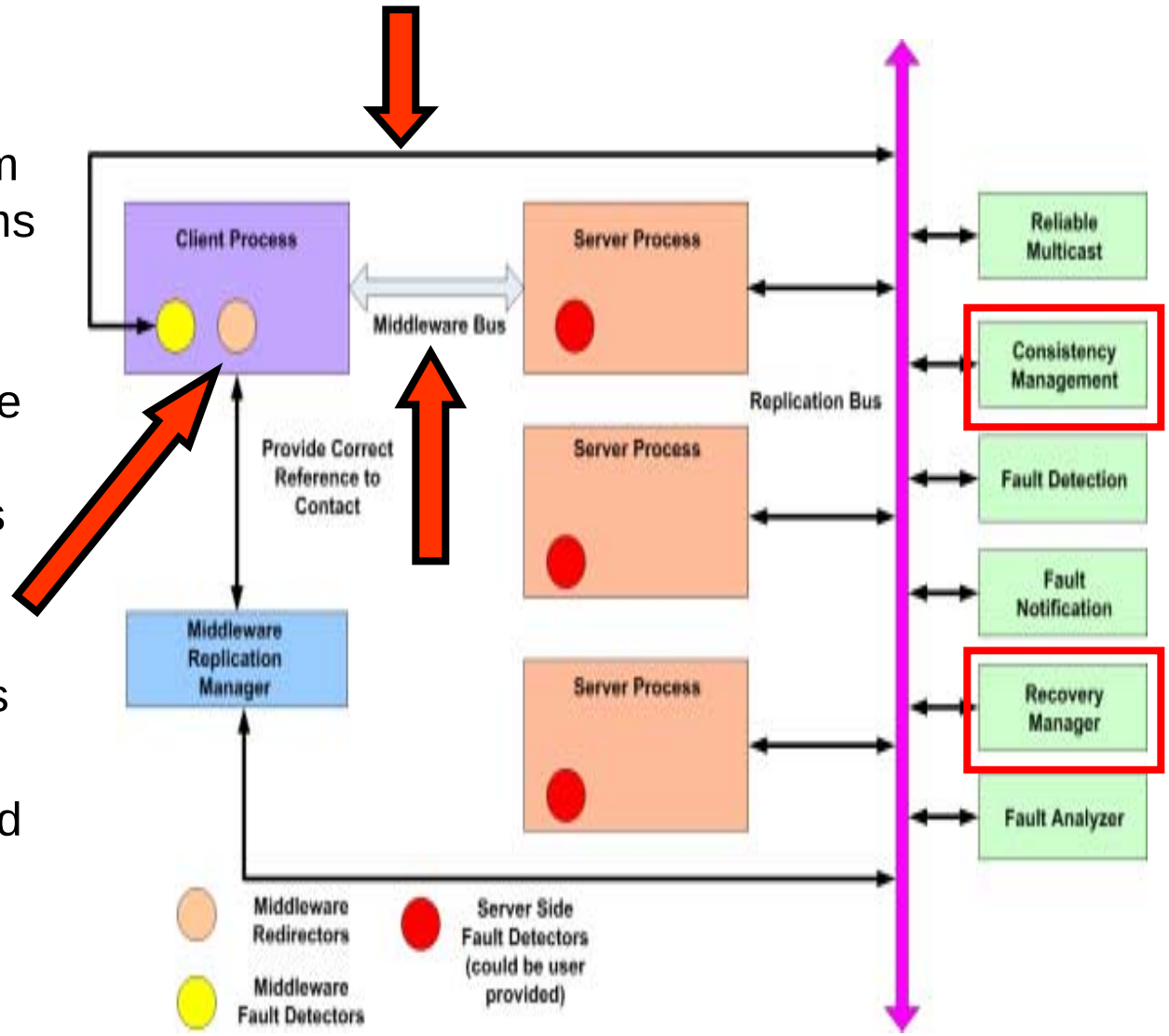
Example Replication Strategy

- Example replication strategy that can provide predictable failure detection and recovery is semi-active replication
- Replica groups are connected in a chain
  - members monitor the member in front of the chain
  - failure detections by transport connection closure or group communication timeouts
- Replica, designated as primary processes the requests and sends state updates to the other members using a reliable multicast protocol



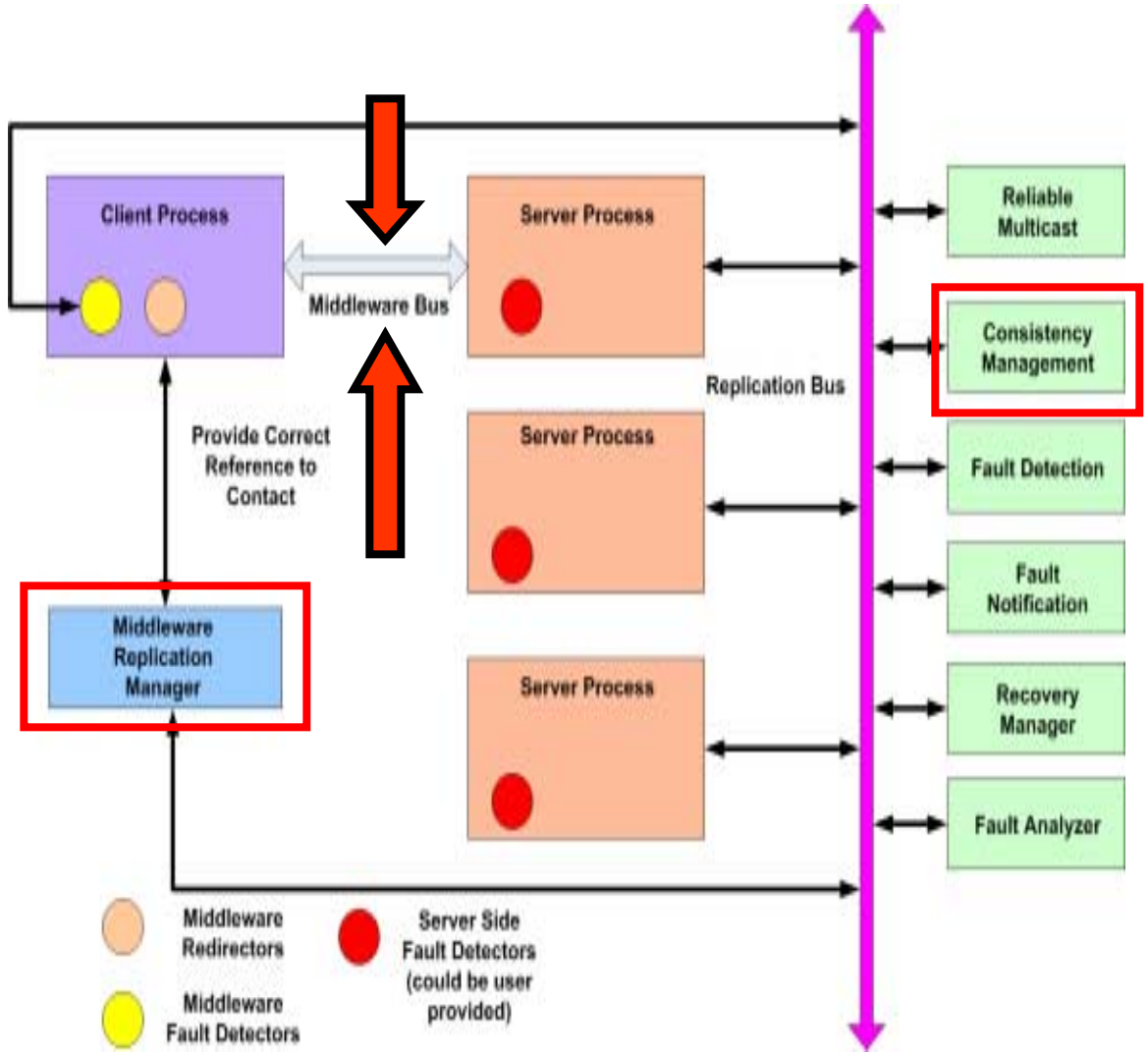
Unexplored Issues with the Lightweight RT FT Architecture (1/2)

- Issues on how to configure active replication away from the middleware mechanisms used for communication
- On the client side, portable interceptors may need to be added to catch exceptions arising from server process failures
  - assuming not using FT-CORBA compliant ORBs
- Middleware connection management timeouts need to work closely with the recovery manager
  - simultaneous failure detections



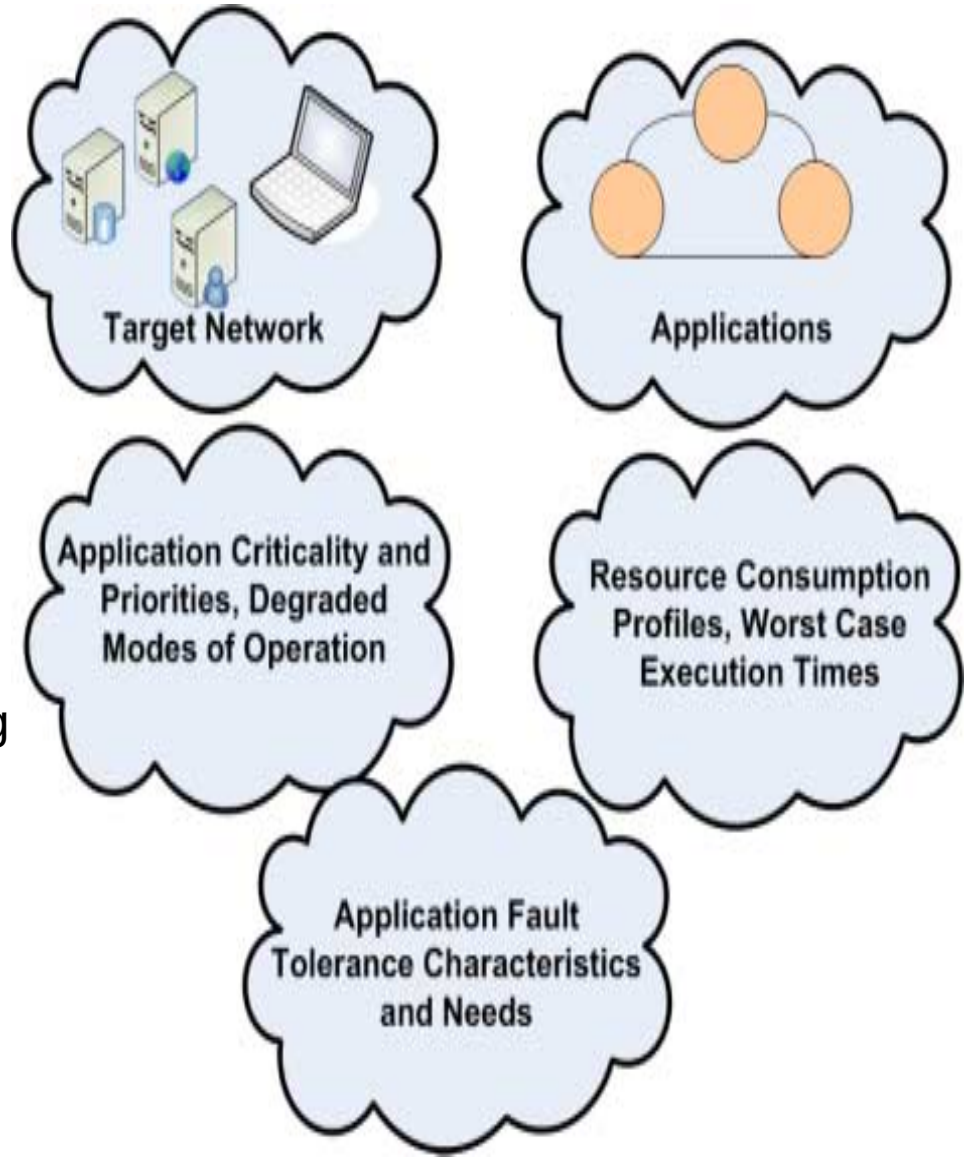
Unexplored Issues with the Lightweight RT FT Architecture (2/2)

- Externally configured consistency management mechanisms must work with the middleware connection management, in order to ensure strong consistency
  - weaker consistency models for updating states to primary as state synchronization happens
- Middleware replication manager might be a single point of failure
- Process group replication needs help from middleware for synchronizing middleware state
  - recovery granularity at the ORB/POA level



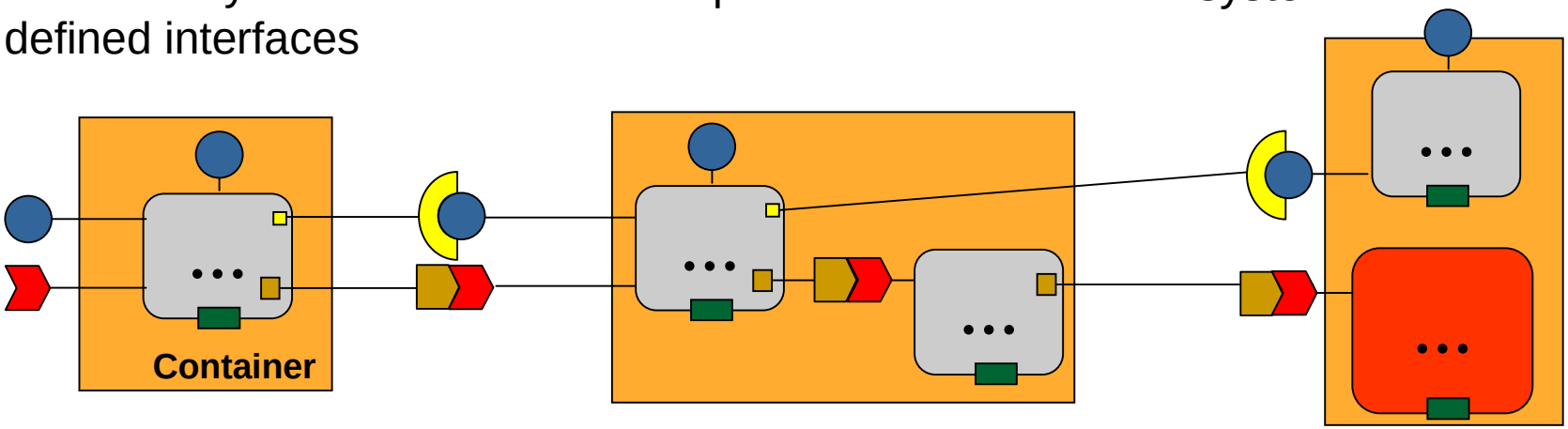
Effectively Using Lightweight RT FT

- Semantic configurability of Fault Tolerance requires collecting
  - application requirements and expected execution profiles
  - target system configurations and resource availabilities
- Efficient deployment of applications and their replicas
  - appropriate support for resource allocations to applications – application placement algorithms
  - real-time ordering of operations using well known dynamic/hybrid scheduling strategies
- Understand needs of the applications
  - allows a chance to configure the external fault tolerance mechanisms most suited for the applications
  - allows opportunities for run-time optimizations in providing fault tolerance



Lightweight CCM for Enterprise DRE Systems

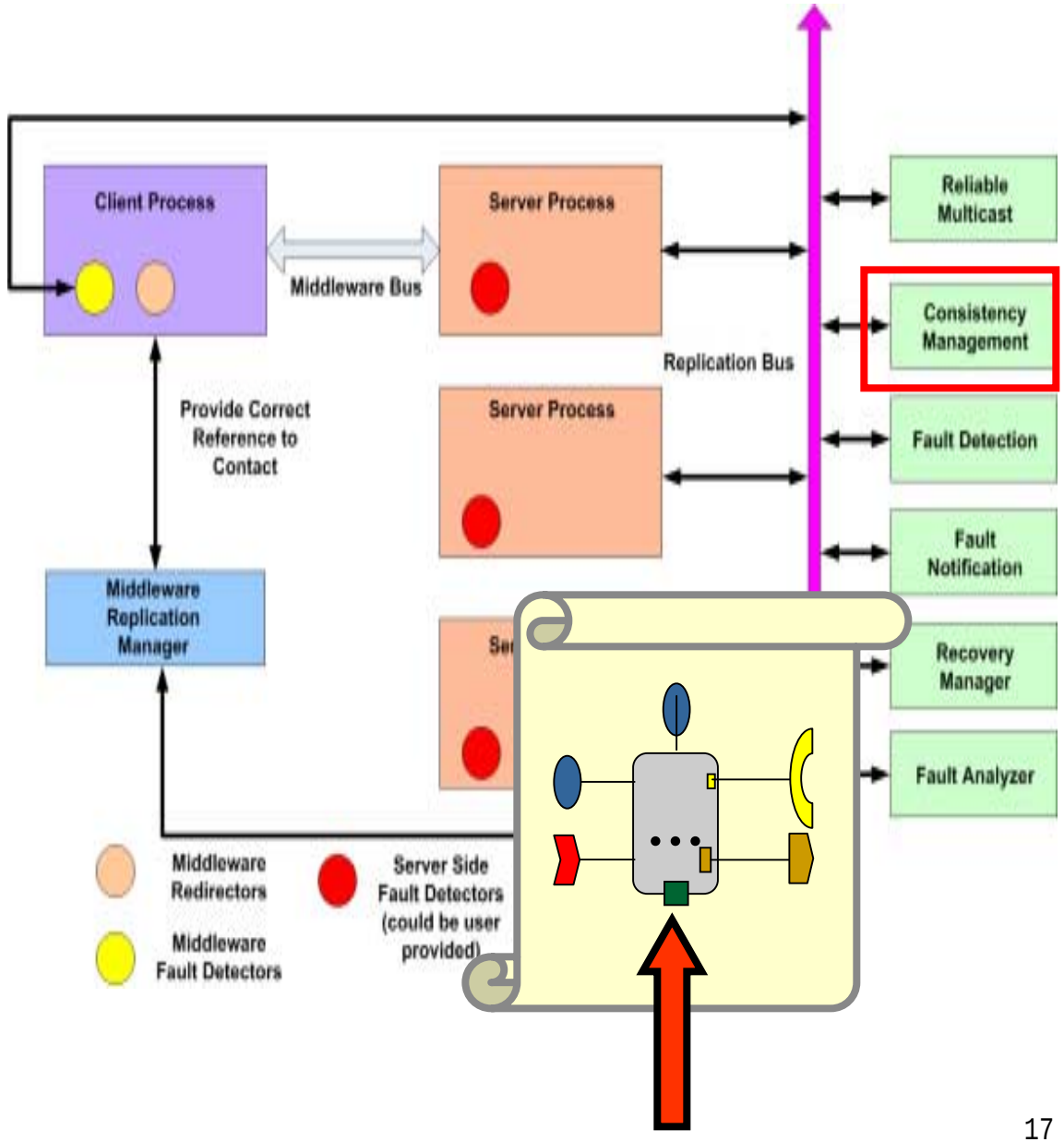
- Creates a standard "virtual boundary" around application component implementations that interact only via well-defined interfaces
- Define standard container mechanisms needed to execute components in generic component servers
- Specify the infrastructure needed to configure & deploy components throughout a distributed system



- provides capabilities for dynamic reconfiguration as well as updating of implementations, so that enterprise DRE system infrastructure can be adaptive to the needs of the applications
  - provides opportunities for reuse of core business logic
  - provides mechanisms to change system behavior at runtime and according to operating environment needs

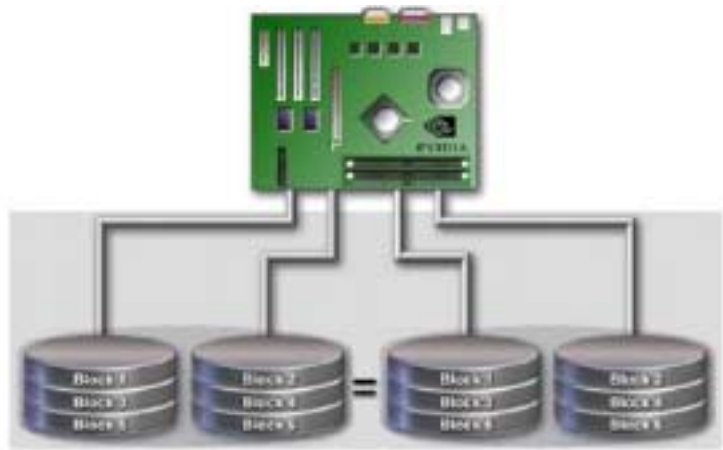
Applicability of Lightweight CCM to Lightweight RT FT

- Common fault tolerance functionalities like the fault notification can be encapsulated as Lightweight CCM components
  - attributes can be set on those Lightweight CCM components to specify the QoS they can provide
    - configure attributes using deployment and configuration tools
  - addition of helper components to interact with these Lightweight CCM components
    - interfaces to the helper methods need not change
    - Lightweight CCM implementations can be changed to get different functionality or different QoS



Concluding Remarks

- Enterprise DRE systems require various degree of deterministic behaviors even in the presence of faults or fluctuating available resources
- FT-CORBA is not suitable for enterprise DRE systems, esp.
  - very heavyweight and hard to integrate with other systems
  - hard to integrate real-time fault tolerance
  - point solutions and hard to customize



- Lightweight RT FT CORBA RFP is trying to address the needs of FT/RT applications
  - decoupling of middleware functionalities that can be customized using well known and researched fault tolerance techniques
  - separation of interfaces for easier control using RT-CORBA
  - coupled with Lightweight CCM, can be easily deployed, configured and managed according to the needs of the applications

We would like to thank Robert Kukura (Raytheon), Paul Werme (NSWC) and Andrew Foster (Prismtech) for providing useful discussions and information about the Lightweight Real-time Fault Tolerance specification and existing fault tolerance practices and solutions in the industry