Tunable Replica Consistency for Primary-Backup Replication in Distributed Soft Real-time and Embedded Systems

Jaiganesh Balasubramanian & Aniruddha Gokhale
EECS, Vanderbilt University, Nashville, TN

Contact: a.gokhale@vanderbilt.edu

Presented at OMG Workshop on Real-time, Embedded and Enterprise-scale Time-critical Systems
May 24-26, 2010

Research supported in part by NSF CAREER and NSF SHF/CNS Awards
Focus: Distributed Real-time Embedded (DRE) Systems

- Heterogeneous soft real-time applications
- Operation in dynamic & resource-constrained environments
  - changing system loads
  - process/processor failures
- Stringent *simultaneous* QoS demands
  - high availability, satisfactory average response times, etc.
  - efficient resource utilization
- Examples include
  - NASA’s Magnetospheric Multi-scale (MMS) mission
  - Total shipboard computing environment (TSCE)
  - Modern office environments

DRE systems need both high availability & soft real-time performance in resource-constrained environments
Standards-based middleware mechanisms available for QoS management

- end-to-end predictable behavior for requests (e.g., RT-CORBA)
  - priority bands
  - thread pools with lanes
  - eliminate priority inversion
- support for highly available systems (e.g., FT-CORBA)
  - replication management
  - failure detection
  - multiple replication styles

Standards-based middleware provide support for different QoS – but only one QoS at a time
Problem – Providing Real-time Fault-tolerance

- Standards-based FT middleware provide only the mechanisms
  - e.g., checkpoint frequency
- No information on how to tune those mechanisms
  - e.g., what checkpoint frequency to use? Can it be changed?
- Ad-hoc ways to provide fault-tolerance do not consider resource usage of the object & the resource availability of the system
  - affects real-time performance

Fault-tolerant middleware needs to manage available resources efficiently to simultaneously provide & maintain soft real-time performance of applications
Prevalent Schemes for Fault-tolerance in DRE Systems

• Active replication?
  • client requests multicast & executed at all the replicas
  • faster recovery – as long as any one replica is alive
  • high communication/processing overhead

• Passive replication?
  • low resource/execution overhead – only primary executes requests
    • primary makes the state of the backup replicas consistent with itself
  • slower recovery time – clients redirected to one of the backups

Passive replication – better suited for resource-constrained DRE systems
Our Prior Contributions: Deployment Phase

• Allocate CPU & network resources efficiently at deployment-time
  • Applications & their replicas are deployed in their appropriate physical hosts => meets high availability requirements
  • CPU & network resource needs of applications are provisioned => meets response time requirements
  • Overcomes inefficient allocations – for both applications & replicas => conserves resources

DeCoRAM D&C Engine: Appeared in RTAS 2010
Our Prior Contributions: Runtime Phase

- Provide both high availability & soft real-time performance at runtime
  - Provide bounded-time failure detection & failure recovery => maintains soft real-time performance even in the presence of failures
  - Resource-aware failure/overload recovery => maintains soft real-time performance after recovering from failures/overloads
  - Overcomes need for ad-hoc mechanisms to detect & recover from failures/overloads that affect soft real-time performance of clients
• Performance versus Fault-tolerance – optimize resource usage
  • Need for configurable application consistency management
    • support for range of consistency assurances – weak to strong
  • Need for analyzing & selecting trade-offs among FT & performance
    • resource usage for FT versus resource usage for performance
  • Need for multi-modal operations – degraded levels of FT & performance
    • dynamic adaptations to system loads & failures

Current Work: Resource-aware Replica Consistency Management
Replica & State Management in Passive Replication

• Replica Management
  • synchronizing the state of the primary replicas with the state of the backup replicas

• Resource consumption trade-offs
  • performance (response times) versus fault-tolerance
  • e.g., if goal is better performance => lesser resources for state management => lesser levels of FT
  • e.g., if goal is better fault-tolerance => response time suffers until all replicas are made consistent

Resource consumption for FT affects performance assurances provided to applications & vice versa
Replica & State Management in Passive Replication

- Diverse application QoS requirements
  - for some applications, FT important
  - for others, performance important

- Need tunable adaptive fault-tolerance
  - cater to the needs of variety of applications
    - no point solutions
  - configurable per-application fault-tolerance properties
    - optimized for desired performance
  - monitor available system resources
    - auto-configure fault-tolerance levels provided for applications

Focus on operating region for FT as opposed to an operating point
• Diverse application QoS requirements
  • for some applications, FT important
  • for others, performance important

• Need tunable adaptive fault-tolerance
  • input → available system resources
  • control → per-application fault-tolerance properties
  • output → desired application performance/reliability
  • fairness → optimize resource consumption to provide minimum QoS
  • trade-offs needed in resource-constrained environments
    • goal → maximize both performance and fault-tolerance
    • degrade QoS – either of FT or performance – as resource levels decrease

Focus on operating region as opposed to an operating point
Resource Optimizations in Fault-tolerant Systems

- Different applications have different requirements
  - e.g., FT more important than performance and vice-versa
- Configurable resource consumption needed on per-application basis
- Under resource constraints
  - trade-offs need to be made to balance the use of available resources for
    - fault-tolerance
    - response times

Need mechanisms that can focus on an operating region rather than an operating point to tune state management
Solution Approach: TACOMA

- Tunable Adaptive COnsistency Management middlewAre (TACOMA)
  - built on top of the FLAReRe middleware
  - configurable consistency management middleware
    - resource-aware tuning of application consistency – i.e., number of replicas made consistent with the primary replica
    - use of different transports to manage consistency – e.g., CORBA AMI, DDS
  - Local Resource Manager – TACOMA agent
    - added on each processor hosting primary replicas
    - application informs the agent when state changes
    - agents synchronize the state of the backup replicas
      - works with FLARe replication manager to obtain object references
TACOMA: Configurable Consistency Management (1/2)

- Determine configurable consistency for each application
  - to respond to a client within a certain deadline, the state of how many backup replicas can be made consistent with the primary replica by the TACOMA agent?
- Time taken to make one backup replica consistent equals
  - the worst case execution time of an update task initiated by the TACOMA agent in the primary replica
  - Sum of worst case execution times of update tasks at all backup replicas + processing time at primary replica = client response time
TACOMA: Configurable Consistency Management (2/2)

• Determine worst case execution times of update tasks
  • use time-demand analysis
• Tunable consistency management
  • input → available system resources
  • control → per-application consistency depth
  • output → desired application performance/reliability
  • fairness → provide minimum QoS assurances
• Configure TACOMA agents with the consistency depth determined
• **Hypotheses:** TACOMA
  
  • is customizable & can be applied to a wide range of DRE systems
    • consistency depth range (1 to number of replicas)
  
  • utilizes available CPU & network resources in the system efficiently, & provides applications with the required QoS (performance or high availability)
    • response times are always met – no deadline misses
  
  • tunes application replication consistency depth at runtime, as resource availability fluctuates
    • consistency depth decreases from MAX (number of replicas) to MIN (1)
Concluding Remarks

- Passive Replication is a promising replication scheme for DRE systems
  - Crucial for resource-constrained environments

- Problems using passive replication for DRE systems
  - Lack of resource-aware FT decisions
    - potential to cause cascading failures
    - potential to cause clients not meet their performance requirements
  - Lack of Adaptive Fault-tolerant Middleware
    - no opportunities for customization & (re)configuration to provide both RT & FT capabilities

- TACOMA is work-in-progress implemented on top of TAO & FLARe

- Our FT-RT contributions include:
  - Resource-aware adaptive fault-tolerant middleware
    - deployment & allocation of applications driven by model-driven QoS allocation engine
    - adaptive fault-tolerant middleware reacts to failures & overloads, & maintains soft real-time requirements simultaneously
  - Replica consistency management middleware that optimizes trade-offs between resource usage, performance, & fault-tolerance
    - Exploit results from RT scheduling theory to understand the implications of FT on performance & resource usage