Adaptive Failover for Real-time Middleware with Passive Replication

Chenyang Lu
CSE 520S

Distributed Real-time Embedded Systems

- Heterogeneous soft real-time applications
- Operation in dynamic environments
- Changing system loads
- Process/processor failures
- Stringent simultaneous QoS demands
- High availability
- Soft real-time performance

Examples include
- NASA's Magnetospheric Multi-scale mission
- Total shipboard computing environment
- Modern office environments

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
  - Suitable for hard real-time
- Passive replication?
  - Low resource utilization – only primary executes requests
  - Primary makes the state of the backup replicas consistent
  - Slower recovery time – clients redirected to one of the backups
  - Suitable for soft real-time

Passive Replication: Challenges

- Failure recovery – selecting a new primary and subsequent failover
- Static failover target selection will not work because
  - Workload fluctuations and multiple failures change load distributions
  - If clients are redirected to heavily loaded replicas upon failure, their response time requirements will not be satisfied

Solution Approach: FLARe: Fault-tolerant Middleware with adaptive failover target selection & overload management support

Contributions

- LAAF: Load-Aware Adaptive Failover Target Selection
  - Load-aware → maintain soft real-time performance after recovery
  - Adaptive → handle dynamic load changes and multiple failures
- ROME: Resource Overload Management and Redirection
  - Maintain soft real-time performance during overloads
- FLARe: Fault-Tolerant Load-Aware and Adaptive Middleware

System Model

- Failure model
  - Multiple processor/process failures
  - Fail-stop
- Replication Model
  - Passive replication
  - Asynchronous state updates
- Implemented on top of TAO Real-time CORBA Middleware
Middleware Architecture

- Client Failover Manager
  - catches processor/process failure exceptions
  - redirects clients to failover targets
- Monitors
  - periodically monitors liveness and CPU utilization of each processor
- Replication Manager
  - collects CPU utilizations from monitors
  - calculates ranked list of failover targets using LAAF
  - updates client-side with ranked list of targets
  - manages overloads using ROME

LAFF: Load-Aware Adaptive Failover

- Monitor CPU utilization of each processor
- Rank backup processors based on load
- Distribute failover targets of objects on a same processor to avoid hotspot after recovery
- Proactively update clients

ROME: Overload Management

- Overload can occur due to multiple processor failures
- Real-time treat overload as failure
- Redirect clients of high utilization objects to backups on lightly loaded processors
- Distributes overloads across multiple processors

Experiment Setup

- Linux clusters at ISISLab
- 6 clients – 2 clients CL-5 & CL-6 are dynamic clients (start after 50 seconds)
- 6 different servers – each have 2 replicas
- Experiment ran for 300 seconds – each server consumes some CPU load
- Rate Monotonic scheduling on each processor

LAAF Algorithm Results

- At 50 secs, dynamic loads are introduced
- At 150 seconds failures are introduced
LAAF Algorithm Results

Concluding Remarks
- Passive replication is an effective low-overhead strategy for distributed soft real-time applications
- Adaptive resource-aware failover target selection
  - Maintain soft real-time performance after failure recovery
  - Handle dynamic workload
  - Handle multiple failures
- FLARe: RT-FT middleware based on passive replication.

Reference
- Slides based on Jaiganesh Balasubramanian’s RTAS’09 presentation.

FLARe available open-source as part of ACE/TAO repository:
http://www.dre.vanderbilt.edu

Washington University in St. Louis