Remote Batch Invocation—SOA with Distributed Objects

Yang Jiao, Marc Fisher II, and Eli Tilevich @ Virginia Tech

Introduction

• Most computing nowadays is distributed in nature
  - The Internet, scientific computation, mobile devices, etc.
• Programming distributed systems is difficult and labor-intensive
  - High latency, partial-failure, differences in memory access, etc.
• Latency lags bandwidth
• Goal: create intuitive programming abstractions for programming efficient distributed systems using objects
  - Ensure good performance by effectively utilizing modern networks

Challenges

• Latency lags bandwidth [1]
• Remote Procedure Calls do not compose efficiently,
  - Data Transfer and Remote Facade patterns hardcode large-granularity interfaces for particular client use cases.

Approach

• Remote Batch Invocation [2]
  - a language abstraction
  - optimize remote communication
  - leverage higher bandwidth, especially in high-latency environments.
• Batch Execution Service and Translation (BEST) [2, 3]
  - efficient middleware library for RBI

Example

How RBI works

RBI code Snippet

Java client

RMI

BEST Client Stub

BEST ServerSkeleton

Java server

Branching

Loops

Exceptions

Benchmark Result

• Emulates a common usage scenario: the client retrieves the object from the server and updates its fields.
  - The RMI version is the slowest
  - The DTO and RBI versions exhibit comparable performance

Future Work

• Employ static and dynamic program analyses to
  - Extract possible concurrency from recorded batch
  - Identify remote objects’ fields needed for local use

• Consider collections of operations as ‘documents’
  - ‘Understand ‘documents’ as batches of operations

References