Resource Demand Estimation in Distributed, Service-Oriented Applications using LibReDE

Simon Spinner

University of Würzburg – Chair of Software-Engineering

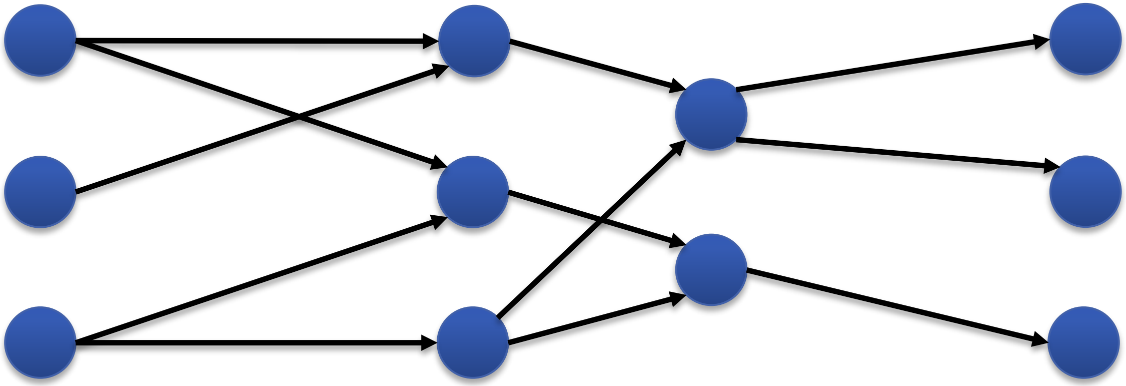
Symposium on Software Performance, Munich, Germany
05/11/2015

Context

Service-oriented applications:

- Integration of different applications (→ SOA)
- Architecture of **one** complex application (→ Microservices)

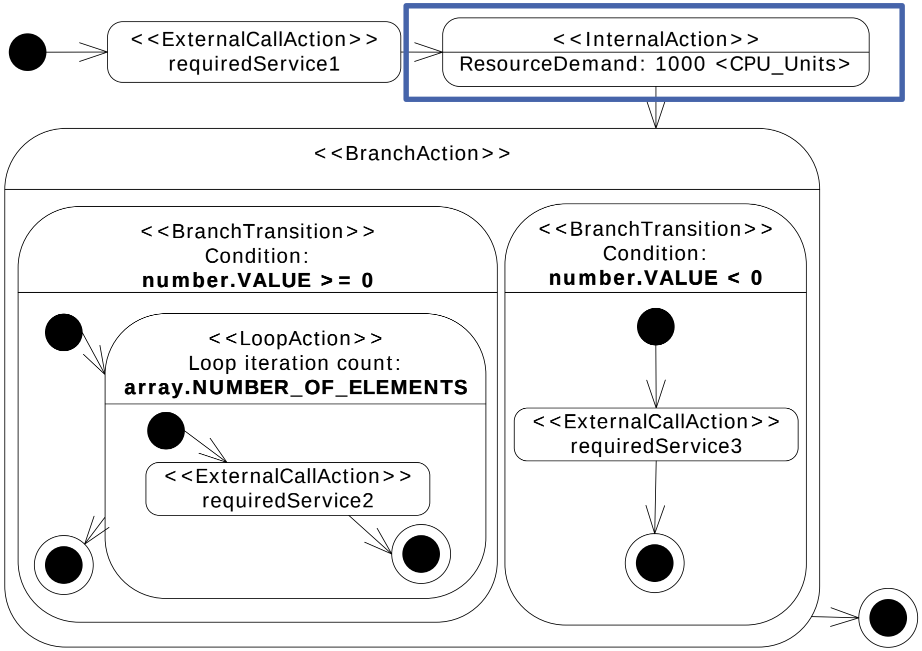
Edge Services   Business Services   Data Services



NETFLIX   facebook   LinkedIn

What are resource demands?

Example SEFF in PCM:



A ***resource demand*** is the time a unit of work (e.g., request or internal action) spends obtaining service from a resource (e.g., CPU or hard disk) in a system.

Resource Demand Estimation

Direct Measurement

Requires specialized infrastructure to monitor low-level statistics.

Examples:

- TimerMeter + ByCounter
- PMWT
- Dynatrace

Statistical Estimation

Use of statistical techniques on high-level monitoring statistics.

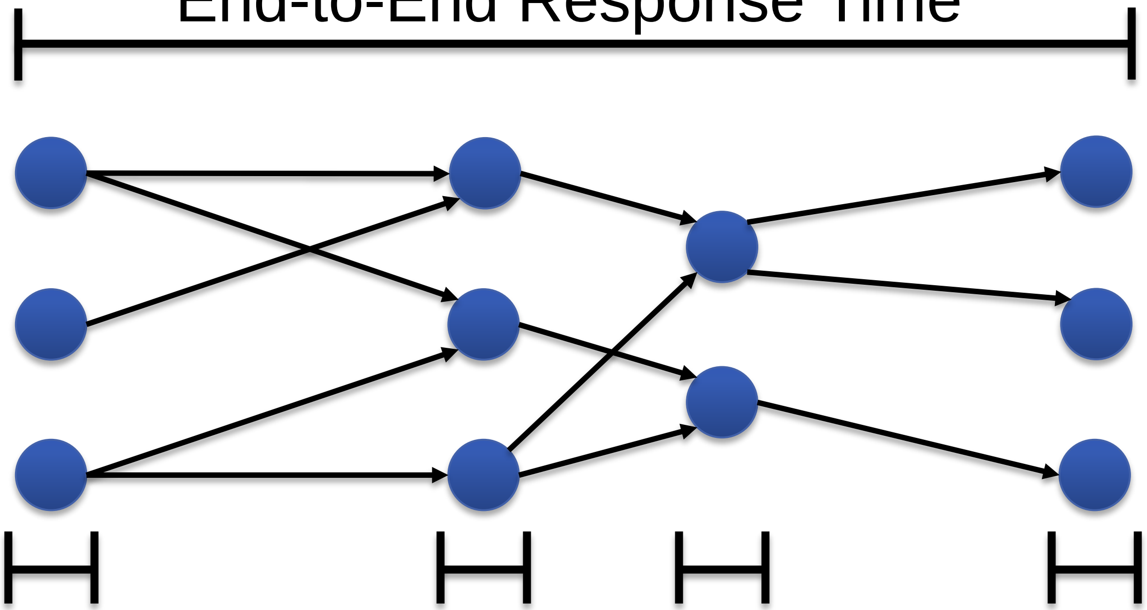
Examples:

- Linear regression
- Kalman filtering
- Nonlinear optimization
- Etc.

LibReDE

Problem

End-to-End Response Time



Residence Times

Utilization

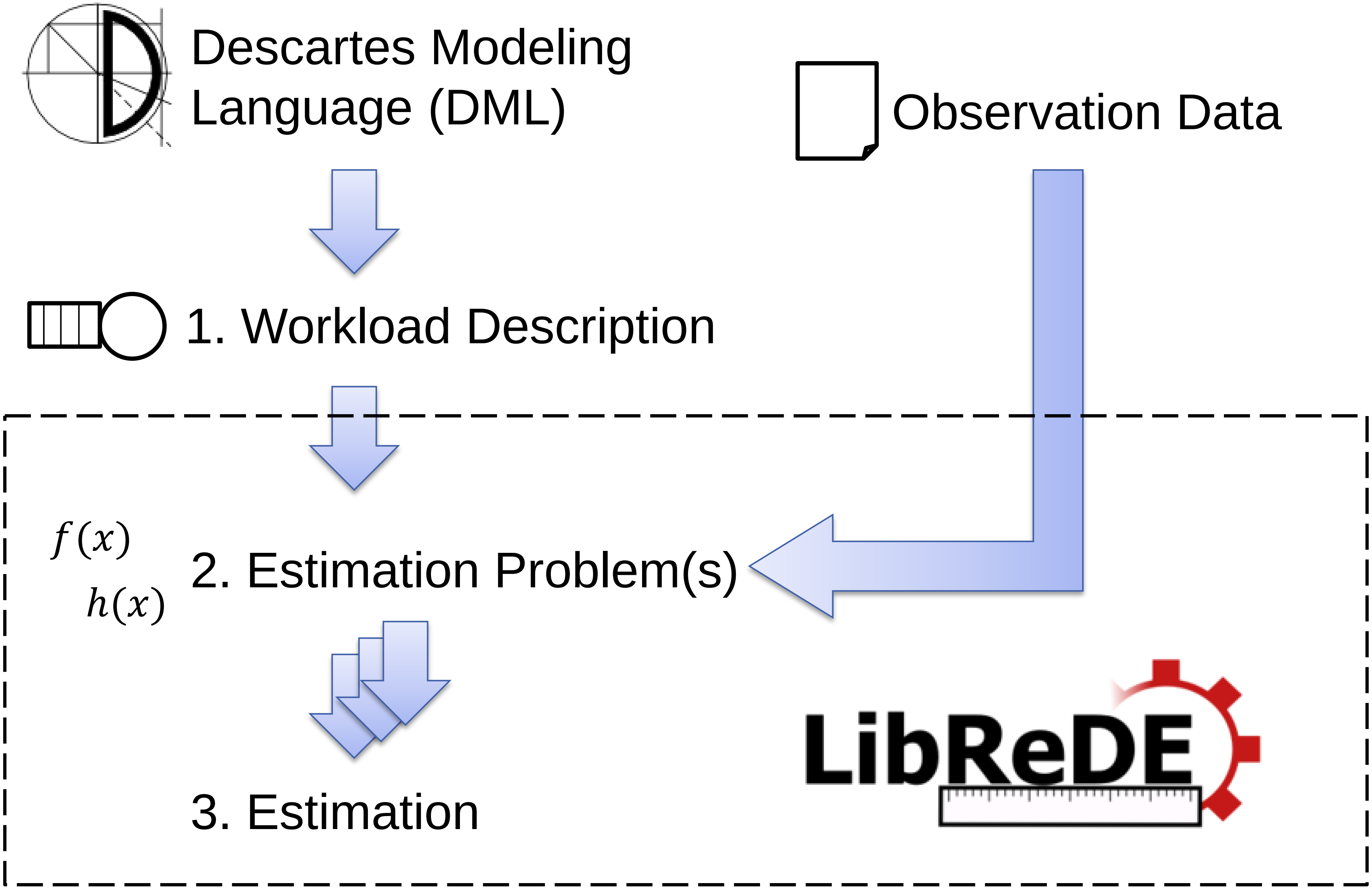
Throughput

Residence times may be **missing** or **inaccurate**

→ Use **end-to-end** response times instead?

→ Existing work limited to **3-tier applications**

Approach Overview



Descartes Modeling Language (DML)

Observation Data

1. Workload Description

$f(x)$

$h(x)$

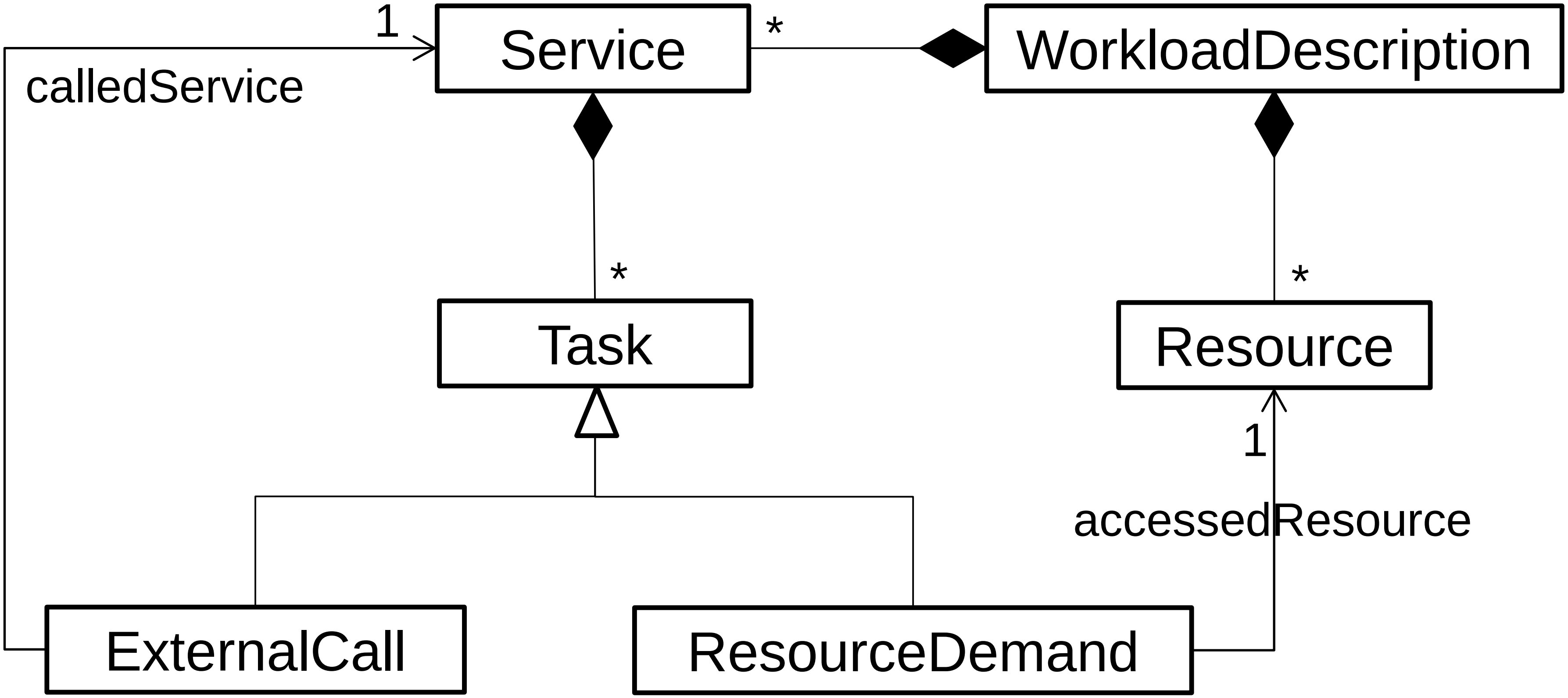
2. Estimation Problem(s)

3. Estimation

LibReDE

1. DERIVE WORKLOAD DESCRIPTION

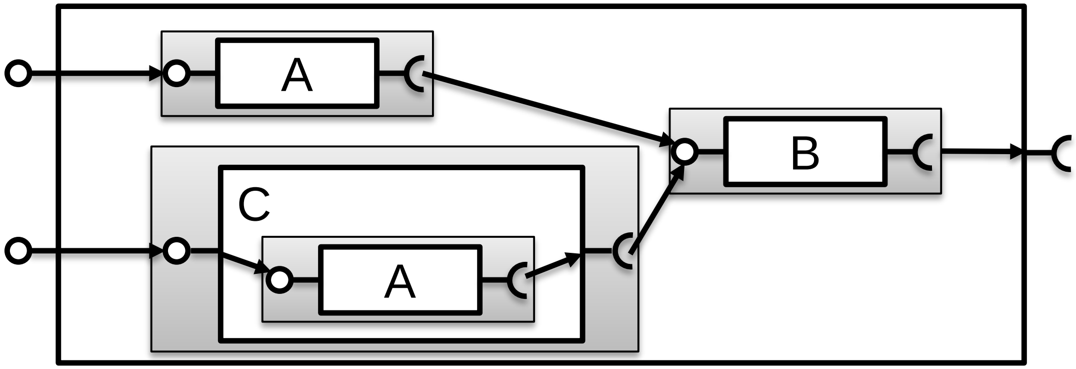
Workload Description



Assumptions

- Any parameter dependencies are solved
- Coarse-grained internal actions
  - Not more than one internal action per resource type in RDSEFF
  - Internal actions in top-level component internal behavior of RDSEFF
- Arbitrary control flow for external calls
  - Loops, branches, forks, etc.
- Product-form workload description

Mapping to DML (1/2)



- Component instance reference
  - Path of assembly contexts
  - Unique within system
- Service in workload description maps to
  - component service
  - of provided interface role
  - of a component instance reference

Mapping to DML (2/2)

- Further mappings
  - Internal action ↔ Resource demand
  - External call ↔ External call
  - Processing resource ↔ Resource
- Visit counts of external calls are derived from DML
  - Loops: average iteration count
  - Branches: weights based on branching probabilities
- Fork actions
  - Without synchronization → Ignore fork
  - With synchronization → Future work

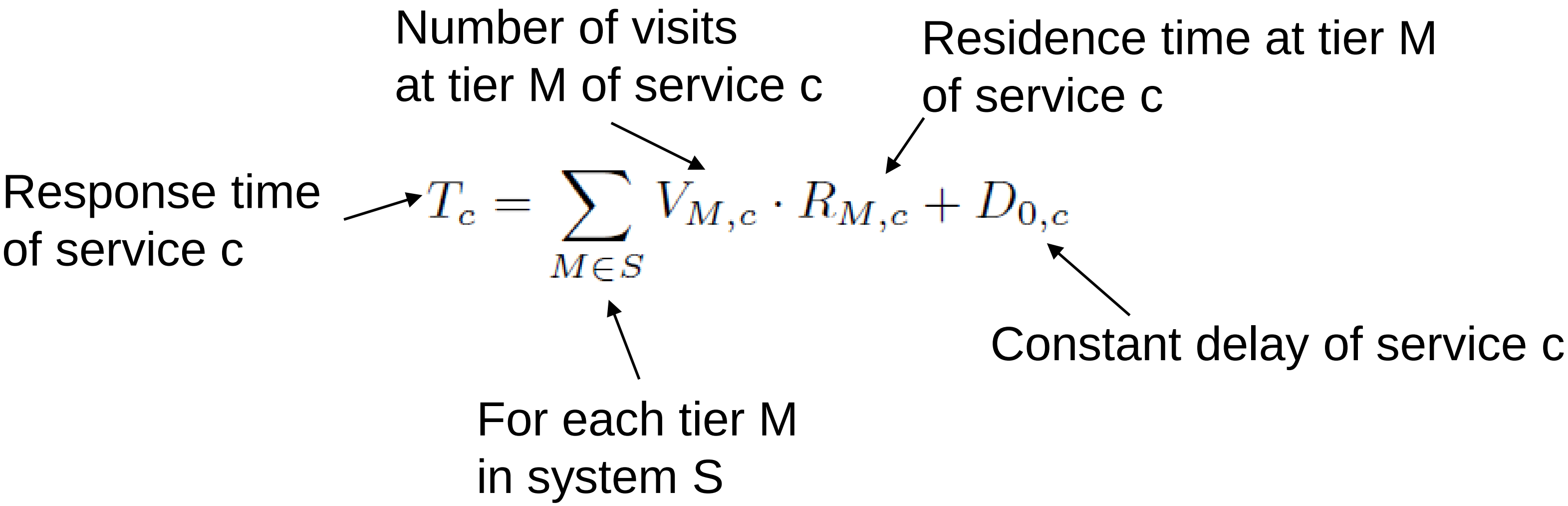
2. DERIVE ESTIMATION PROBLEM

Estimation Problem

- State model
  - Definition of state variables (i.e., resource demands)
  - Constraints on state variables
  - Initial values of state variables
- Observation model
  - Analytical function $\vec{y} = h(\vec{x})$
  - $\vec{y}$: vector of observations
  - $\vec{x}$: vector of state variables
- Estimation algorithm
  - Mathematical solution algorithm
  - E.g., non-linear constrained optimization

Strategies

- Resource level
  - Use only utilization and throughput measurements
- Tier level
  - Use residence times
- System level
  - Use end-to-end response times



Number of visits at tier M of service c

Residence time at tier M of service c

Response time of service c

$$T_c = \sum_{M \in S} V_{M,c} \cdot R_{M,c} + D_{0,c}$$

Constant delay of service c

For each tier M in system S

3. ESTIMATION

Optimization

- Non-linear, constrained optimization
  - Interior-point solver (→ Ipopt library[1])
  - Integrated in LibReDE
- Minimize:
  - Relative difference between
    - Observed and calculated response times
    - Observed and calculated utilization
  - Constant delays
- Equal weights for all parts of the objective function

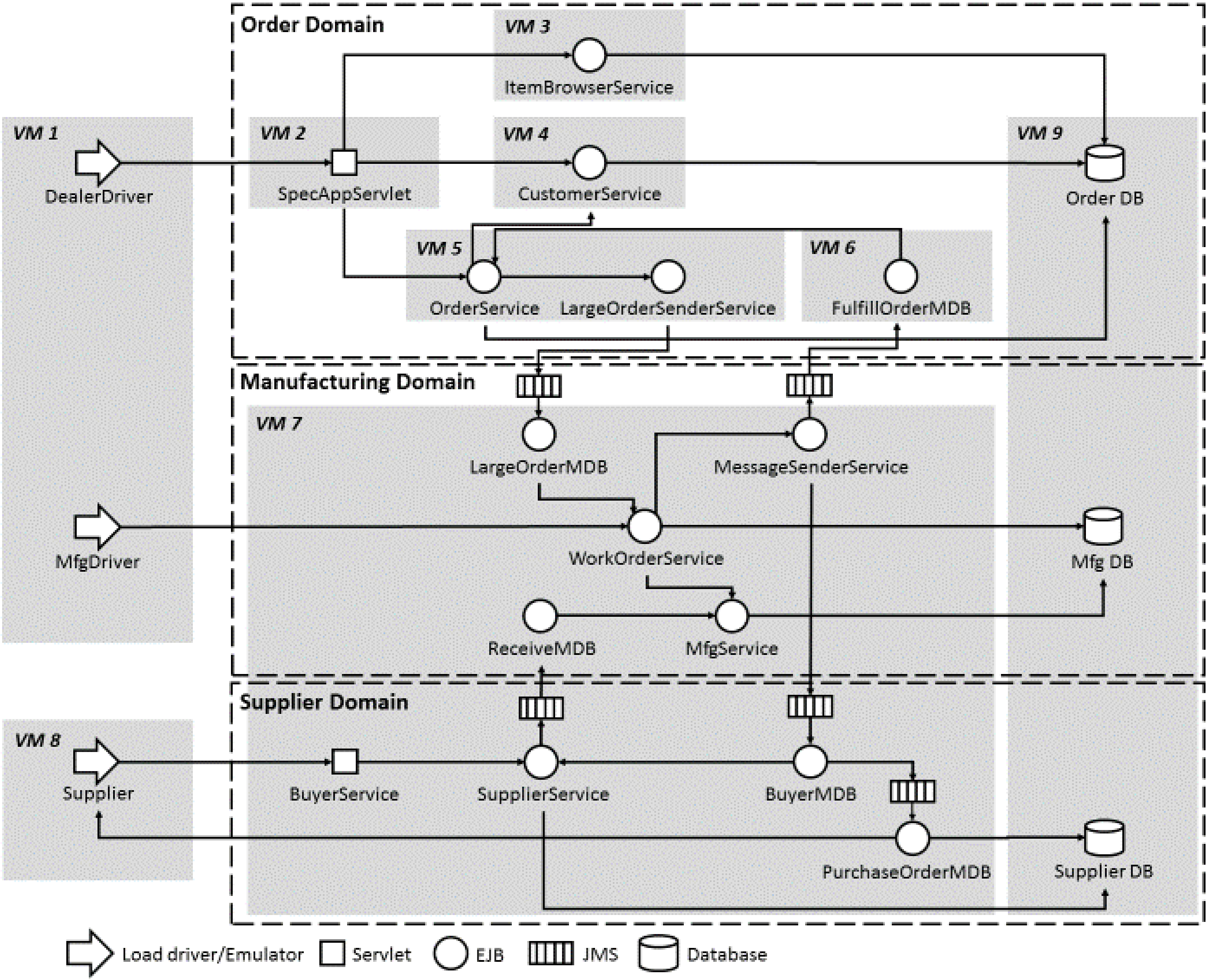
[1] https://projects.coin-or.org/Ipopt

State Space

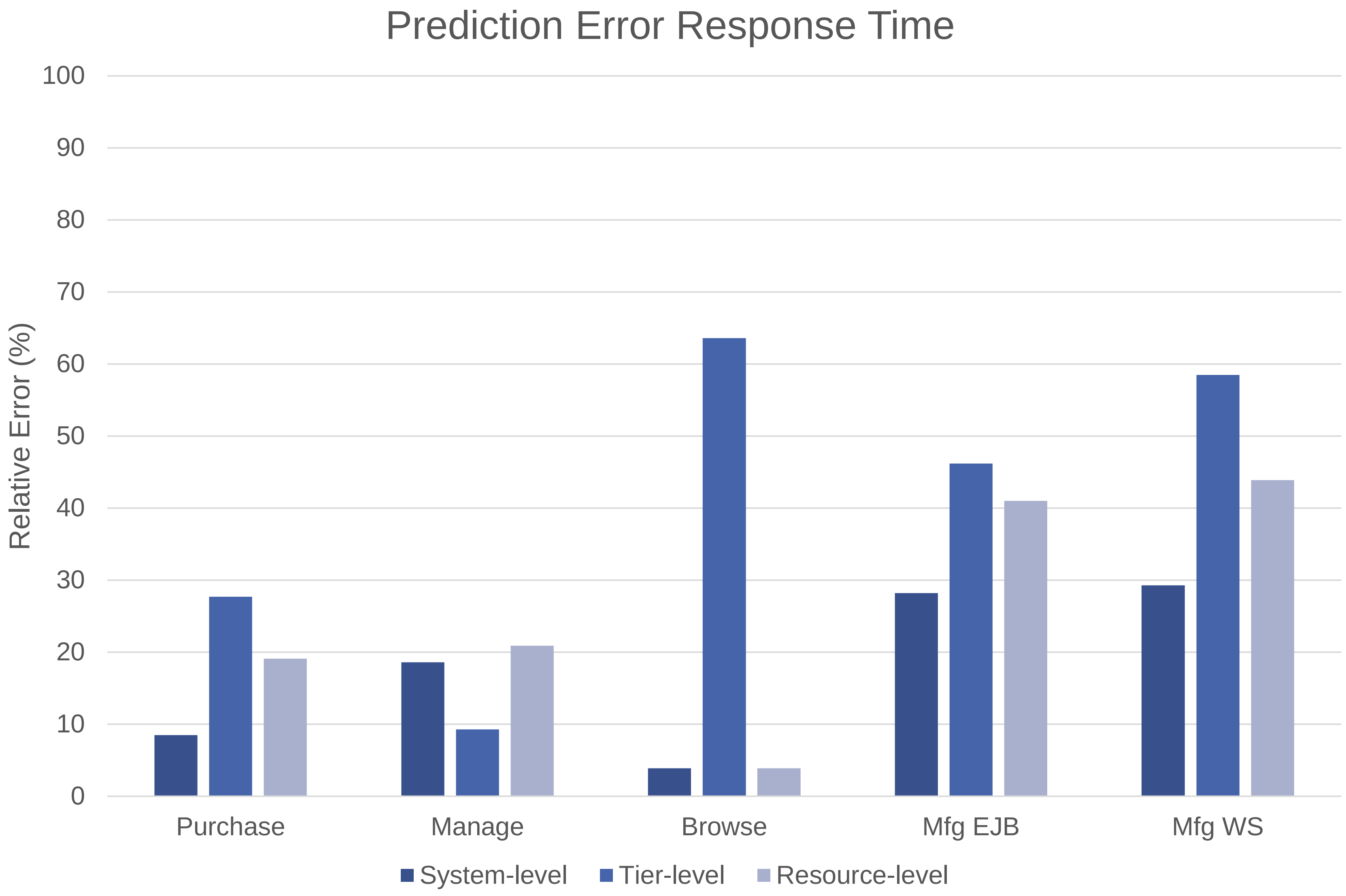
- Ipopt requires
  - Jacobi matrix
  - Hessian matrix for Lagrange multiplicators
- Use Rall’s system for automatic differentiation
  - Automatic calculation of all partial derivatives
  - Memory and computational complexity may be limiting
  - See DerivativeStructure in Apache Commons Math

CASE STUDY

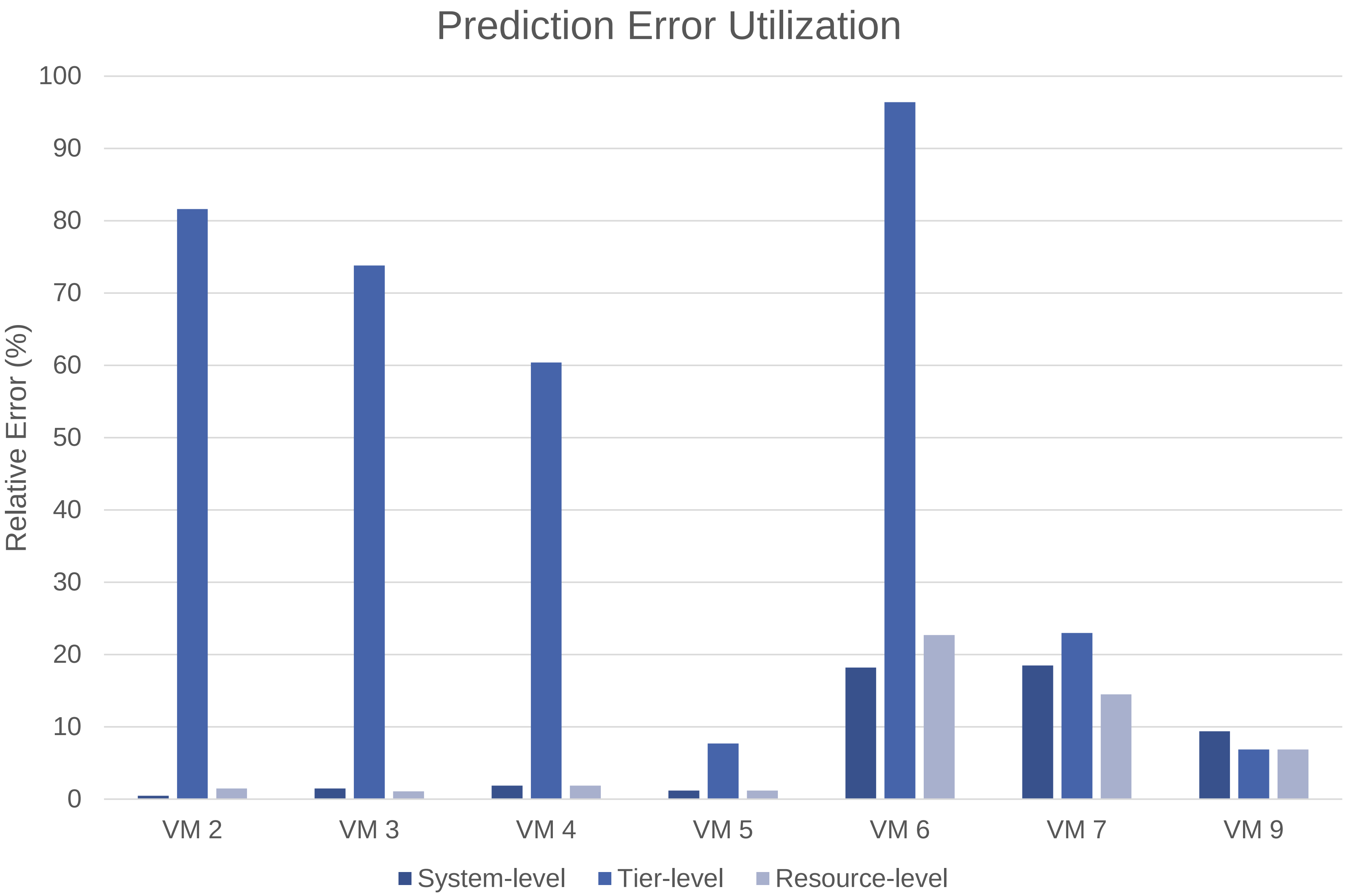
Experiment Setup



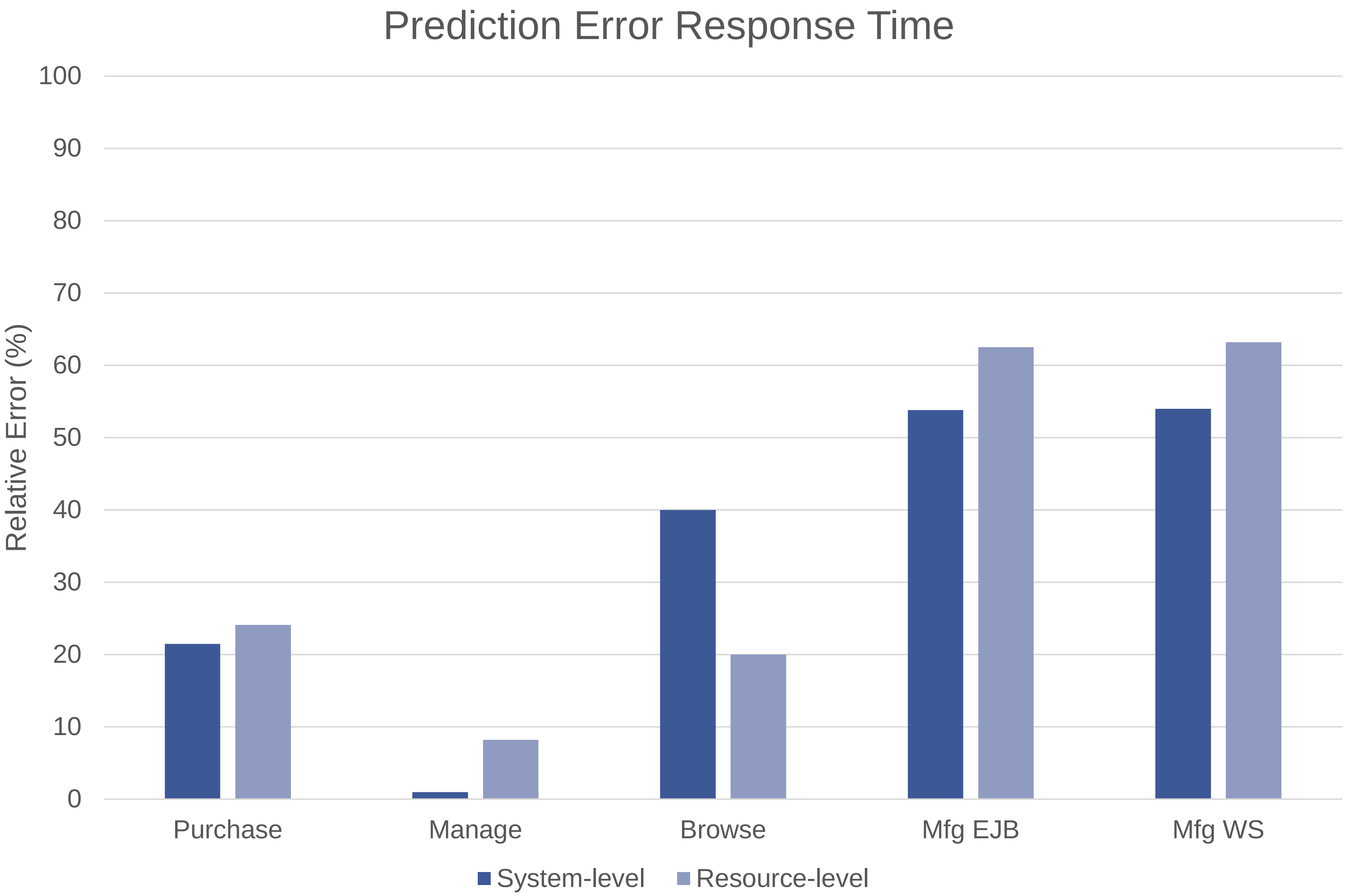
Results: Transaction Rate 60 (1/2)



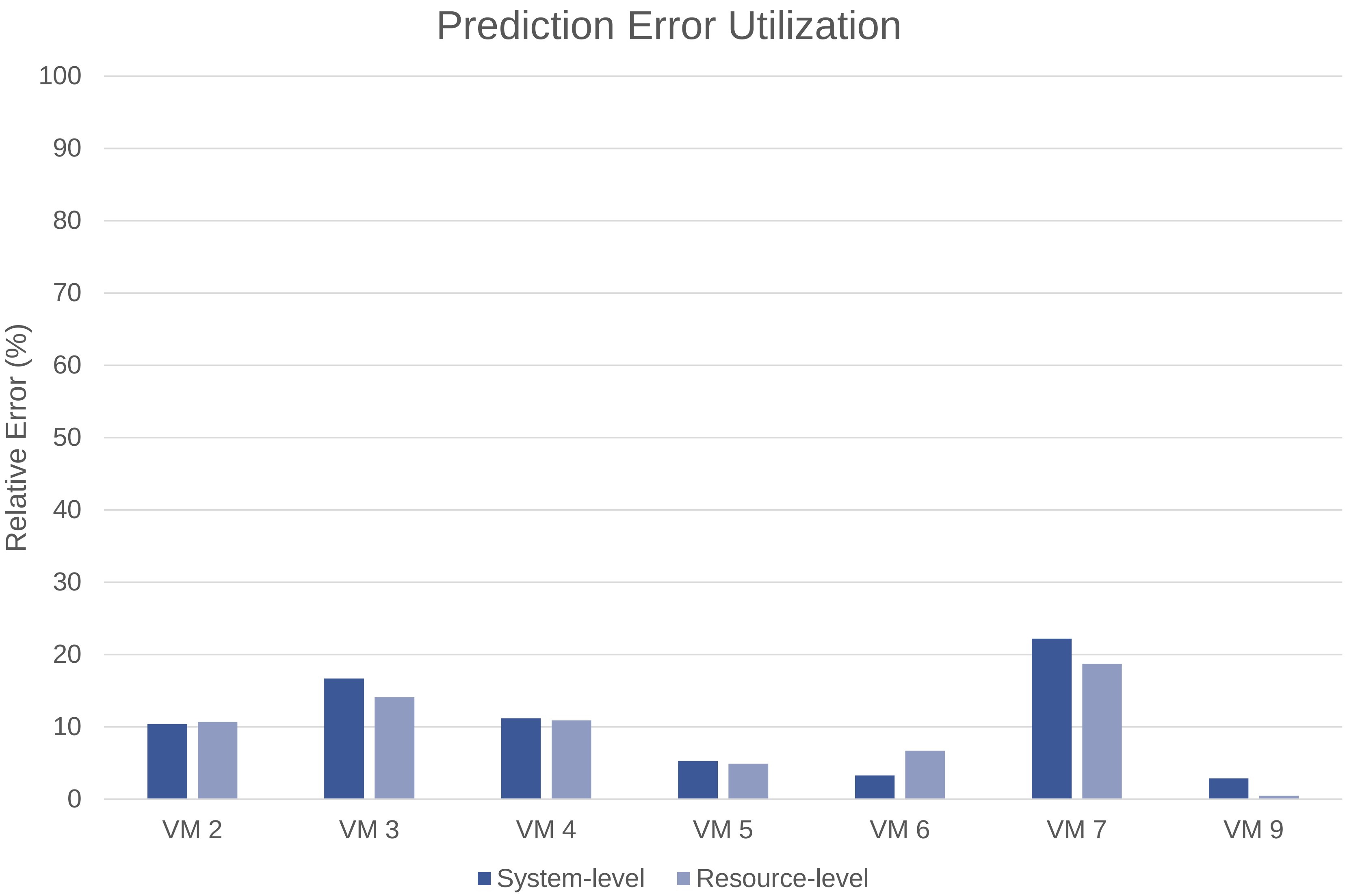
Results: Transaction Rate 60 (2/2)



Results: Transaction Rate 100 (1/2)



Results: Transaction Rate 100 (2/2)



Summary

- Extended LibReDE to support service-oriented applications
  - Control flow awareness
  - Based on end-to-end response times
- Identified different strategies for resource demand estimation
  - Resource-level
  - Tier-level
  - System-level
- Experimental results show
  - System-level is a feasible alternative
  - Tier-level highly depends on accuracy of residence times

LibReDE

http://descartes.tools/librede

Eclipse Public License (EPL)

vmware®

Google Research Award