Feedback-Driven Concurrency Improvement and Refinement of Performance Models

Diploma Thesis Final Talk
Misha Strittmatter | July 12, 2013
Context: Design Time Performance Evaluation

Model

Conceptual System

Specification Abstraction

Appropriate?

Introduction

Related Work

Approach

Implementation

Validation

Conclusion

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

July 12, 2013
Context: Design Time Performance Evaluation

- Model
  - Structure
  - Behavior
- Conceptual System
  - Specification
  - Abstraction
  - Appropriate?
- Simulation
- Behavior
  - Prediction
  - Accuracy?

Introduction
Related Work
Approach
Implementation
Validation
Conclusion

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement
Context: Design Time Performance Evaluation

Model

Simulation

Behavior

Conceptual System

Structure

Behavior

QoS Requirements

Accuracy

Appropriate?

Prediction

Accuracy?

Compliance

Specification Abstraction

Introduction

Related Work

Approach

Implementation

Validation

Conclusion

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

July 12, 2013 2/24
Refinement

Appropriate?
Accuracy?
Compliance?

System
Behavior
QoS Requirements

0.4 s
Refinement

Appropriate?
Accuracy?
Compliance?

System

Behavior

QoS Requirements

Refine
Adjust
Evaluate

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

Introduction
Related Work
Approach
Implementation
Validation
Conclusion

July 12, 2013 3/24
Refinement

Refine
Adjust
Evaluate

Appropriate?
Accuracy?
Compliance?

System
Behavior
QoS Requirements

0.4 s
Tradeoff: Level of Abstraction

High Abstraction (Low Detail) vs. Low Abstraction (High Detail)

System

Behavior

QoS Requirements

Introduction

Related Work

Approach

Implementation

Validation

Conclusion

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

July 12, 2013
Flaws Hindering Refinement

- Appropriate?
- Accuracy?
- Compliance?

System

Behavior

QoS Requirements

Introduction

Related Work

Approach

Implementation

Validation

Conclusion
Flaws Hindering Refinement

Appropriate?
Accuracy?
Compliance?

System
Behavior
QoS Requirements

0,4 s
Flaws Hindering Refinement

Refine
Adjust
Evaluate

Appropriate?
Accuracy?
Compliance?

System
Behavior
QoS Requirements

Introduction
Related Work
Approach
Implementation
Validation
Conclusion
Result Interpretation
Introduction

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

Related Work

Approach

Implementation

Validation

Conclusion

July 12, 2013
Contribution

Action
- Identify:
  - Concurrency Flaws & Refinement Potential
  - Ways of Automatic Detection
  - Possible Solutions
- Propose Feedback-Driven Refinement Process
- Implement a Tool to Automate the Process

Benefit
- Abstraction Tradeoff:
  Refine / Optimize Model Parts Depending on Impact
- Flaws Hindering Refactoring:
  - Automatic Flaw Detection
  - Provide Solution Catalogue
- Result Interpretation: Improved Support
Related Work

Foundations:
- SPE
- CBSE
- PCM
- MB-/MDSE

Performance Anti-Patterns

Anti-Pattern Detection

Model Analysis

Software Analysis

Monitoring

Systematic Experimenting

Bottleneck Analysis

Result Feedback & Visualization

Refactoring

Concurrency Patterns

AP Detection & Tuning

Autotuning

Concurrency Optimization

Structure

Results

AP Detection & Tuning

[Trubiani & A. Koziolek]

[Tichy, Pankratius, Schäfer]

[Zheng, Rajan]

[van Hoorn]

[Westermann, J. Happe, Hauck]

[Smith & Williams]

[Parsons, Miller, Boroday]

[Smith & Williams]

[Parsons, Miller, Boroday]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]

[Smith & Williams]
Related Work

Foundations:
- SPE
- CBSE
- PCM
- MB-/MDSE

Performance Anti-Patterns
- [Smith & Williams]

Anti-Pattern Detection
- [Parsons, Miller, Boroday]
- [Westermann, J. Happe, Hauck]

Monitoring
- [van Hoorn]

Software Analysis
- Systematic Experimenting
- [Wert]

Bottleneck Analysis
- [Wert]

Result Feedback & Visualization
- [Krogmann, Wittenburg]

Refactoring
- [Fowler, Mens]

Concurrency Patterns
- [Zheng, Rajan]

My Work

Concurrency Optimization

AP Detection & Tuning
- [Trubiani & A. Koziolek]
- [Tichy, Pankratius, Schäfer]
- [van Hoorn]

Model Analysis
- [Cortellessa, Trubiani]

Results
- [Erdogan]

Structure
- [Cortellessa, Trubiani]
Hot-Spots

Where to Refine or Optimize?

Introduction
Related Work
Approach
Implementation
Validation
Conclusion

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

July 12, 2013

9/24
Introduction
Related Work
Approach
Implementation
Validation
Conclusion

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

July 12, 2013
Hot-Spots

Find Hot-Path

Legitimate High Resource Demand vs. Unrefined
Hot-Spots

Refine and Optimize

Introduction

Related Work

Approach

Implementation

Validation

Conclusion

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

July 12, 2013
Flaw Catalogue

Flaw Template

- Flaw Description
- Detection
- List of Solutions
- Automatic Solutions?

Identified Concurrency Flaws

- Hot-Spots
- Hardware Bottleneck (Severe)
- Unbalanced Processing
- One Lane Bridge (Severe)
- Unusual High Waiting Times
Process

Start → Initial Configuration, Model → Analyse → Analysis Results, Visualizations → Model → Reaction → Done

Introduction
Related Work
Approach
Implementation
Validation
Conclusion

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement
July 12, 2013
Analysis Feature Model

Analysis Configuration

Upper Workload Intensity Bound

Flaw Treatment

Legend
- or
- exclusive or
- mandatory
- optional

List of Flaws to Ignore

Workload Intensity Scaling

Upper Workload Intensity Bound

Flaw Treatment

Utilization Margin

Individual Specification of Treatment for each Unresolvable Flaw

Utilization Increment

Utilization Range Scan

Bottleneck Search

Unresolvable Flaw Treatment

Ignore

Manual

Automatic

Terminate

Static

Dynamic

Ignore

Manual

Automatic

Terminate

Static

Dynamic

Ignore

Manual

Automatic

Terminate

Gradient Heuristic

Linear Regression Heuristic

...
Analysis Mode:
Ascending Flaw Detection

Simulation
Analysis
Scale Load
Severe Flaw Detected
Model
Results
Feedback
Intensity Increment
Workload Intensity Scaling
Flaw Treatment
Dynamic
Utilization Range Scan
 terminated
Do not Terminate
on: Unbalanced Processing, ...

Determine Increment
Utilization History
Simulation
Analysis
Model

Introduction
Related Work
Approach
Implementation
Validation
Conclusion

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement
July 12, 2013 13/24
Analysis Modes

- **Ascending Flaw Detection**: Find First in Effect Severe Flaw
- **Scan**: Explore System’s Behavior in Fixed Range
- **Binary Bottleneck Search**: Find Bottleneck, Minimize Experiments
- **Resource Scaling**: Resource Dimensioning
- **Interactive**: Refine Newly Created Models
- **Automatic**
Feedback

- Flaw Appearance (Diagnostic)
- Rankings
- Proportion Mappings
- Hot-Path
- Metric Report (Information)

1. CPU1 – 90%
2. HDD – 20%
3. CPU2 – 16%
4. LAN – 9%
...
Implemented Features

- **Core Functions:**
  - Automatic Execution of Experiment Series
  - Storing of Experiment Results & Models
  - Model Modification

- **Analysis Modes:**
  - Scan
  - Ascending Flaw Detection (Gradient Heuristic)

- **Automatic Flaw Detection:**
  - Hardware Bottleneck
  - One Lane Bridge
  - Unbalanced Processing

- **Feedback Generation:**
  - Flaw Appearance
  - Utilization Rankings
  - Series Report
  - Experiment Response Time Report

- **Extensibility:** Modes, Flaw Detection, Feedback
GUI: Overview

Introduction

Related Work

Approach

Implementation

Validation

Conclusion

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

July 12, 2013
Validation

Refine the Business Reporting System
PCM model.

- Validate: Flaw Detection, Effectiveness of Solutions
- Increased Capacity to 10 Users
- Ascending Flaw Detection, 15% Utilization Step Size
- Try to Avoid Plain Process Rate Increases
- Try Various Solutions
## Validation: Summary

<table>
<thead>
<tr>
<th>Series</th>
<th># Experiments</th>
<th>Max Users</th>
<th>Detected Flaws</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Network Bottleneck</td>
<td></td>
</tr>
</tbody>
</table>

**Diagram:**

- **WebServer**
- **SchedulerServer**
- **ApplicationServer**
- **DatabaseServer**

**Resource Environment**

**Introduction**

**Related Work**

**Approach**

**Implementation**

**Validation**

**Conclusion**
## Validation: Summary

<table>
<thead>
<tr>
<th>Series</th>
<th># Experiments</th>
<th>Max Users</th>
<th>Detected Flaws</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Network Bottleneck</td>
<td>Network Split</td>
</tr>
</tbody>
</table>

### Resource Environment

- **WebServer**
  - CPU
- **SchedulerServer**
  - CPU
- **ApplicationServer**
  - CPU
- **DatabaseServer**
  - CPU
  - HDD

---

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

*July 12, 2013*
# Validation: Summary

<table>
<thead>
<tr>
<th>Series</th>
<th># Experiments</th>
<th>Max Users</th>
<th>Detected Flaws</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Network Bottleneck</td>
<td>Network Split</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>Network Bottleneck</td>
<td></td>
</tr>
</tbody>
</table>

## Resource Environment

- **WebServer**: CPU
- **SchedulerServer**: CPU
- **ApplicationServer**: CPU
- **DatabaseServer**: CPU, HDD
## Validation: Summary

<table>
<thead>
<tr>
<th>Series</th>
<th># Experiments</th>
<th>Max Users</th>
<th>Detected Flaws</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Network Bottleneck</td>
<td>Network Split</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>Network Bottleneck</td>
<td>Server Consolidation, TP Increase</td>
</tr>
</tbody>
</table>

### Resource Environment

- **WebServer**
  - CPU

- **SchedulerServer**
  - CPU

- **Application- & DatabaseServer**
  - CPU
  - HDD

---

**Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement**

July 12, 2013

22/24
## Validation: Summary

<table>
<thead>
<tr>
<th>Series</th>
<th># Experiments</th>
<th>Max Users</th>
<th>Detected Flaws</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Network Bottleneck</td>
<td>Network Split</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>Network Bottleneck</td>
<td>Server Consolidation, TP Increase</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>6</td>
<td>One Lane Bridge</td>
<td></td>
</tr>
</tbody>
</table>

### Resource Environment

```
Resource Environment

WebServer
  CPU

SchedulerServer
  CPU

Application- & DatabaseServer
  CPU
  HDD
```

---

**Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement**

**Introduction**

**Related Work**

**Approach**

**Implementation**

**Validation**

**Conclusion**

July 12, 2013

22/24
## Validation: Summary

<table>
<thead>
<tr>
<th>Series</th>
<th># Experiments</th>
<th>Max Users</th>
<th>Detected Flaws</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Network Bottleneck</td>
<td>Network Split</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>Network Bottleneck</td>
<td>Server Consolidation, TP Increase</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>6</td>
<td>One Lane Bridge</td>
<td>Capacity Increase</td>
</tr>
</tbody>
</table>

### Resource Environment

- WebServer
- SchedulerServer
- Application- & DatabaseServer

---

**Introduction**

**Related Work**

**Approach**

**Implementation**

**Validation**

**Conclusion**

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

Jul 12, 2013 22/24
## Validation: Summary

<table>
<thead>
<tr>
<th>Series</th>
<th># Experiments</th>
<th>Max Users</th>
<th>Detected Flaws</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Network Bottleneck</td>
<td>Network Split</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>Network Bottleneck</td>
<td>Server Consolidation, TP Increase</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>6</td>
<td>One Lane Bridge</td>
<td>Capacity Increase</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>9</td>
<td>CPU Bottleneck</td>
<td></td>
</tr>
</tbody>
</table>

### Resource Environment

- **WebServer**: CPU
- **SchedulerServer**: CPU
- **Application- & DatabaseServer**: CPU, HDD

---

**Introduction**

- Related Work
- Approach

**Implementation**

- Validation
- Conclusion

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement
## Validation: Summary

<table>
<thead>
<tr>
<th>Series</th>
<th># Experiments</th>
<th>Max Users</th>
<th>Detected Flaws</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Network Bottleneck</td>
<td>Network Split</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>Network Bottleneck</td>
<td>Server Consolidation, TP Increase</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>6</td>
<td>One Lane Bridge</td>
<td>Capacity Increase</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>9</td>
<td>CPU Bottleneck</td>
<td>Core Added</td>
</tr>
</tbody>
</table>

### Resource Environment

- **WebServer**
  - CPU

- **SchedulerServer**
  - CPU

- **Application- & DatabaseServer**
  - CPU
  - HDD
  - CPU

---

### Introduction

- Related Work

### Approach

- Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

### Validation

- July 12, 2013

### Conclusion

- ○○○○○
**Validation: Summary**

<table>
<thead>
<tr>
<th>Series</th>
<th># Experiments</th>
<th>Max Users</th>
<th>Detected Flaws</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Network Bottleneck</td>
<td>Network Split</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>Network Bottleneck</td>
<td>Server Consolidation, TP Increase</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>6</td>
<td>One Lane Bridge</td>
<td>Capacity Increase</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>9</td>
<td>CPU Bottleneck</td>
<td>Core Added</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>15</td>
<td>CPU &amp; Network Bottleneck</td>
<td></td>
</tr>
</tbody>
</table>

**Resource Environment**

- WebServer
- SchedulerServer
- Application- & DatabaseServer

---

**Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement**

July 12, 2013
Conclusions

Everything my approach / tool does can also be done manually.

But...
Conclusions

Everything my approach / tool does can also be done manually.

But...

- Requires Much Time and Effort
- Is Prone to Careless Mistakes
- Requires Skill (Flaw Detection & Solution)
- Requires Knowledge about the System
Conclusions

Everything my approach / tool does can also be done manually.

But...

- Requires Much Time and Effort
- Is Prone to Careless Mistakes
- Requires Skill (Flaw Detection & Solution)
- Requires Knowledge about the System
- May Lead to Ad-hoc, Unsystematic Refinement & Optimization
Future Work

- **Conceptual:**
  - General Performance Flaws
  - ProtoCom Prototype Generation

- **Implement:**
  - Extension: Flaws, Analysis Modes, Visualizations
  - Improvement: Performance, Usability
  - Give Actual Advice
  - Visual Architectural Feedback
  - Drill-Down Navigation for Feedback

- **Further Validation:** Empirical Study
Summary

Problem:
- Concurrency Flaws Impair Performance, Hinder Refinement
- Result Interpretation Difficult
- Trial & Error Refinement Inefficient
- Manual Scalability Assessment Cumbersome

Idea:
- Guide Developer During Refinement
- Put Intelligence in Tooling

Benefit:
- Goal-Driven Flaw Detection
- Flaws Mitigated within Design Time
- Model Refined where Relevant
- Refinement Improves Flaw Detection

Action:
- Identify Flaws, their Detection and Solutions
- Develop Feedback-Driven Refinement Process
- Implement Automatization Tool
References
References I


F. Brosch et al. „Architecture-Based Reliability Prediction with the Palladio Component Model“. In: Software Engineering, IEEE Transactions on 38.6 (2012), pp. 1319–1339. ISSN: 0098-5589. DOI: 10.1109/TSE.2011.94.


Appendix
<table>
<thead>
<tr>
<th>Series</th>
<th>Average Response Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>332.8</td>
</tr>
<tr>
<td>1</td>
<td>305.3</td>
</tr>
<tr>
<td>2</td>
<td>20.4</td>
</tr>
<tr>
<td>3</td>
<td>15.4</td>
</tr>
<tr>
<td>4</td>
<td>8.4</td>
</tr>
</tbody>
</table>

Table: Usage Scenario Response Time Comparison (10 Users)
Mode: Binary Bottleneck Search

1. Scale Load
2. Intensity Change
3. Analysis
4. Results
5. Model
6. Model
7. Simulation

Feedback
[Bottleneck Detected]
Mode: Resource Scaling

Scale Load

[Max Workload Reached]
[Other Problem or non-Solvable Bottleneck]
[No Bottleneck]
[Solvable Hardware Bottleneck]

Model

Simulation

Scale Resources

Analysis

Model

Results

Feedback

References

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

July 12, 2013

38/24
Mode: Automatic

Simulation

Scale Load

Results

Analysis

Fix Problems

Model

Fix Problems

Model

Results

Model

Simulation

Scale Load

Model

Analysis

[No Problems]

[Solvable Problems]

[Max Workload Reached]

[Non Solvable Problem]

Feedback

References

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

July 12, 2013 40/24
GUI: Tree View

References

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement
GUI: Settings

Validation
- Validate

Analysis Mode
- Scan
- Ascending Problem Detection
- Binary Bottleneck Search
- Resource Scaling
- Interactive
- Automatic

Workload Specification
- Start Population: 1
- Max Population:
- Step Size:
- Type: closed
- UsageScenario ID: _qhidVnpBE6yqgfK4Hy_BFA

Simulation Stop Conditions
- Max Simulation Time: 150000
- Max Measurement Count: 10000

Initial PCM Instance
- Allocation Model
  - C:\Models\BRS_CaseStudy\brs.allocation
  - Workspace...
  - File System...
  - Default

- Usage Model
  - C:\Models\BRS_CaseStudy\brs.usagemodel
  - Workspace...
  - File System...
  - Default

- Middleware Repository
  - pathmap://PCM_MODELS/Glassfish.repository
  - Workspace...
  - File System...
  - Default

- Event Middleware Repository
  - pathmap://PCM_MODELS/default_event_middleware.repository
  - Workspace...
  - File System...
  - Default

Temp Data
- Delete Temp Data after Analysis

References

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement

July 12, 2013
RW Cites for the Venn Diagramm

- Anti-Patterns Smith and Williams 2000; Smith and Williams 2002; Smith and Williams 2003a; Smith and Williams 2003b
- Patterns Zheng and Harper 2010d; Rajan, Kautz, and Rowcliffe 2010a
- Autotuning Schaefer 2010b; Schaefer, Pankratius, and Tichy 2010; Pankratius, Jannesari, and Tichy 2009b
- Monitoring Parsons 2005b; Parsons 2007; Miller et al. 1995; Boroday et al. 2005a
- Anne & Catia Trubiani and Koziolek 2011b
- Structure Cortellessa, Di Marco, et al. 2010; Cortellessa, Martens, et al. 2010
- Results: Erdogan 2012b
- Catia AP Feedback Diss: Trubiani 2011a
- Feedback: Krogmann et al. 2009a
- Alex: Wert 2012d
- SoPeCo: Westermann et al. 2010c
- Kieker: Hoorn, Waller, and Hasselbring 2012c
- Refactoring: Mens and Tourwe 2004

References

Misha Strittmatter – Feedback-Driven Concurrency Improvement and Refinement
Ownership and Copyright to the title image ("Stairs") is owned by the user andrewcs and http://www.sxc.hu. Its license permits me the use within this presentation. However, duplicating the image from this presentation is prohibited by its license. Instead you can download it there: http://www.sxc.hu/photo/875056 (if you adhere to its license)

Cliparts from:
http://office.microsoft.com/de-de/images/

All other images by me.