MiSim — A Lightweight and Extensible Simulator for a Scenario-based Resilience Evaluation of Microservice Architectures @SSP2021

André van Hoorn
Alireza Hakamian
Sebastian Frank
Lion Wagner
Scientific Background and Motivation
Scientific Background and Motivation

12 Resilience Scenario[1]

ChaosToolkit (CTK) Experiments

Lower execution costs
Allows Parallel Running
Can Run on dedicated hardware

predict/prioritize

24 Requirements for the Simulator
Requirements Evaluation

<table>
<thead>
<tr>
<th></th>
<th>SimuLizar</th>
<th>DRACeo</th>
<th>MiSim</th>
<th>µqsim</th>
<th>MuSim</th>
</tr>
</thead>
<tbody>
<tr>
<td>R9 CTK faultloads</td>
<td>N</td>
<td>~4</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>R10 LIMBO support</td>
<td>Y¹</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y³</td>
</tr>
<tr>
<td>R11 Resilience Features</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R11.1 Self-healing (restart)</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y³</td>
</tr>
<tr>
<td>R11.2 Auto Scaling</td>
<td>Y¹</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y³</td>
</tr>
<tr>
<td>R11.3 Load Balancing</td>
<td>Y¹</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>R11.4 Retry</td>
<td>Y¹</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>R11.5 Circuit Breaker</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>R11.6 Rate Limiter</td>
<td>Y¹</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>R11.7 Caching</td>
<td>Y¹</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>R12 Compatibility</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>R12.1 Cambio Scenarios</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

MiSim — A Lightweight and Extensible Simulator for a Scenario-based Resilience Evaluation of Microservice Architectures
What does MiSim do?
What does MiSim do?

Service Oriented Architectures
- Message Traveling (Failures/Timeouts/Dependencies)
- Separation of Service Instances

Resilience Pattern Behavior
- Circuit Breaker
- Retry
- ...

Chaos Injections
- Chaos Monkey (Instance/Service Failure)
- Network Delay
How does MiSim work?
How does MiSim work?

Architecture Description

```json
{
    "network_latency": ".02+.002-0.001",
    "microservices": [
        {
            "name": "gateway",
            "instances": 1,
            "capacity": 10000,
            "loadbalancer_strategy": "even",
            "operations": [
                {
                    "name": "API_Endpoint",
                    "demand": 1,
                }
            ]
        }
    ]
}
```
How does MiSim work? 

Architecture Description

```
{
  "network_latency": ".02+.002-0.001",
  "microservices": [
    {
      "name": "gateway",
      "instances": 1,
      "capacity": 10000,
      "loadbalancer_strategy": "even",
      "operations": ["..."],
      "patterns": [
        {
          "type": "retry",
          "strategy": {
            "type": "linear"
          }
        },
        {
          "name": "Service_A" ...
        },
        {
          "name": "Service_B" ...
        }
      ]
    }
  ]
}
```
How does MiSim work?

Experiment Description

Architecture Tradeoff Analysis Method (ATAM) Scenario\(^3\)

Resilience Test
How does MiSim work?

Experiment Description

Architecture Tradeoff Analysis Method (ATAM)

Scenario

```json
{
    "name": "SSP Demonstration",
    "description": "Shuts down 2 instances of Service_A at 45 STU.",
    "report_location": "SSP_Report/",
    "duration": 130,
    "seed": 42,
    "artifact": "Service_A",
    "component": "API_Endpoint",
    "stimulus": "LOAD src/test/resources/SSPExample/SSP_Limbo.csv AND KILL @45",
    ...
}
```
MiSim — A Lightweight and Extensible Simulator for a Scenario-based Resilience Evaluation of Microservice Architectures
Extensibility of MiSim

Extend one of these:
- ServiceOwnedPattern
- InstanceOwnedPattern
- ExperimentAction
- ...

and Annotate.
- @JsonTypeName("<pattern_type_name>")

Listen. (optional)
- IRequestUpdateListener

Implement.

Profit.

```
{  
  "type": "<pattern_type_name>",
  "config": {
    "field1": 5,
    "field2": "5.3+2"
  },
  "strategy": {
    "type": "<strategy_type_name>",
    "config": {
      "strategy_field3": "42",
      "strategy_field4": "C:\"  
  }
}
```

Architecture Parser instantiates Objects automatically.
Evaluation and Conclusion
Evaluation Plan

MiSim — A Lightweight and Extensible Simulator for a Scenario-based Resilience Evaluation of Microservice Architectures

10/11/2021
Evaluation Results Excerpt

Delay Injection

![Graph showing response times over simulation time](image-url)
Ongoing Work

- Performance Improvements
- Container Orchestration
- Formalization of Scenarios
Links and References

Where to find us:

• MiSim 3.x onwards https://github.com/Cambio-Project/resilience-simulator
• Cambio-Project https://github.com/Cambio-Project

References:


Additional Information
Extensibility in MiSim

Retry Example

```java
@JsonTypeName("retry")
public class Retry extends StrategicInstanceOwnedPattern<IRetryStrategy> implements IRequestUpdateListener {
    @Expose
    private int maxTries = 5;
}
```

Expects Strategy Definition

```java
@JsonTypeName("jittering")
public class JitteringExponentialBackoffRetryStrategy extends ExponentialBackoffRetryStrategy {
}
```

Is registered as network listener.
What does MiSim do?

Input

Simulation

Output

Architecture Description

Service Orientated Architectures
• Message Traveling (Failure/Timeouts/Dependencies)
• Separation of Service Instances

Resilience Pattern Behavior
• Circuit Breaker
• Retry

Chaos Injections
• Chaos Monkey (Instance/Service Failure)
• Network Delay

(Multicore) CPU Scheduling

LIMBO Load Models

Experiment Description

Simulation Trace

Raw Metrics
• Instance Counts
• Response Times
• CPU Usage
• …

Analysis Scripts

Dependency Graph (soon)
Scientific Background and Motivation
Roots of MiSim

ORCAS Project:
Leverage Simulation for Efficient Resilience Evaluation

Cambio-Project:
Using ATAM Scenarios to describe Resilience requirements.

Requirements Specification of 24 Requirements for the Simulator
What does MiSim NOT do?

- Storage Medium/Hardware Simulation
- Hypothesis/Model Checking
- Preprocess Data
Architecture MiSim 3.0
(excluding parsing and data collection)
Software Scenarios — Example

Source: Wage Clerks
Stimulus: Triggers a lot of Calculations
Artifact: Microservices A-D
Response: 1. Handle the calculations in time
2. Autoscale if necessary
Response Measure: Response-Time <= 1s for 99% of Requests
Environment: Regular Service Hours

Target: Calculation Request Endpoint
Method: Normal workload that increases linear over a short time period
Hypothesis: Load Profil
Steady-State

MiSim — A Lightweight and Extensible Simulator for a Scenario-based Resilience Evaluation of Microservice Architectures
Modification of MiSim 2.0

Architecture Improvements
- Separate unrelated Code
- Enforce Strategy Patterns
- Enforce Observer Patterns
- Network Simulation

Architecture Reengineering

Requirements Implementation
- Resilience Patterns
- Chaos Injections
- Data/Metric Collector
- Input/Scenario Parser

MiSim — A Lightweight and Extensible Simulator for a Scenario-based Resilience Evaluation of Microservice Architectures
## Feature-Set Comparison of MiSim 2.0 and 3.0

<table>
<thead>
<tr>
<th>Requirement</th>
<th>MiSim 2.0</th>
<th>MiSim 3.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1 (uses DES)</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>R2 (headless mode)</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>R3 (lightweight)</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>R4 (parallel runs)</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>R5 (output metrics)</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>R5.1 Response Times</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R5.2 Error Rates</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>R5.3 Throughput</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>R5.4 Queue lengths</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>R5.5 Execution Traces</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>R6 (raw output)</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>R7 (common architecture desc.)</td>
<td>~¹</td>
<td>Y</td>
</tr>
<tr>
<td>R8 (case study faultloads)</td>
<td>~²</td>
<td>Y</td>
</tr>
<tr>
<td>R9 (CTK faultloads)</td>
<td>~²</td>
<td>Y</td>
</tr>
<tr>
<td>R10 (LIMBO support)</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>R11 (resilience features)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R11.1 Self-healing (restart)</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>R11.2 Auto Scaling</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>R11.3 Load Balancing</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>R11.4 Retry</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>R11.5 Circuit Breaker</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>R11.6 Rate Limiter</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>R11.7 Caching</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>R12 (compatibility)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R12.1 Cambio Scenarios</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>R12.2 Resirio [Zor21]</td>
<td>~</td>
<td>Y</td>
</tr>
<tr>
<td>R12.3 TransVis [Bec21]</td>
<td>~</td>
<td>~</td>
</tr>
</tbody>
</table>

- Most requirements are now satisfied
- R5.2, R5.3 and R11.1 are achievable with little additional manual effort
- R11.7 Caching was not implemented as simulated request currently don’t contain content
How does MiSim work?

Experiment Description

Resilience Test

```json
{
  "simulation_metadata": {
    "name": "SSP Demonstration",
    "description": "Shuts down 2 instances of Service_A at 60 STU.",
    "report_location": "SSP_Report/",
    "duration": 120,
    "seed": 42
  },
  "generators": [
    {
      "type": "limbo",
      "config": {
        "arrival_time": 0,
        "model": "src/test/resources/SSPExample/SSP_Limbo.csv",
        "target_operation": "gateway.PING"
      }
    },
    {
      "named_monkey": {
        "type": "chaos_monkey",
        "config": {
          "microservice": "gateway",
          "instances": "2"
        }
      }
    }
  ]
}
```
MiSim — A Lightweight and Extensible Simulator for a Scenario-based Resilience Evaluation of Microservice Architectures

DiSpel

STAKEHOLDERS

Verification

Specification

SOFTWARE SYSTEM

ARCHITECTURAL RECONFIGURATION

TRANSIENT BEHAVIOR

ATAM-BASED SCENARIOS

Visualization & Interaction

Natural Language

Formal Language

PSP

PCTL

STRUCTURAL REQUIREMENTS

ARCHITECTURAL RECONFIGURATION

TRANSIENT BEHAVIOR

EVENT DETECTION

MONITORING

SUGGESTIONS

MODEL CHECKING

REASONING MODEL

SOFTWARE SYSTEM

ARCHITECTURAL RECONFIGURATION

TRANSIENT BEHAVIOR

EVENT DETECTION

MONITORING

SUGGESTIONS

MODEL CHECKING

REASONING MODEL