DS-Bench Toolset:
Tools for Dependability Benchmarking with Simulation and Assurance

Dependable Systems and Networks (DSN 2012), Boston, MA, USA

Hajime Fujita (U. of Tokyo, Japan)[*1]
Yutaka Matsuno (U. of Tokyo, Japan) [*2]
Toshihiro Hanawa (U. of Tsukuba, Japan)
Mitsuhisa Sato (U. of Tsukuba, Japan)
Shinpei Kato (UC Santa Cruz, USA) [*2]
Yutaka Ishikawa (U. of Tokyo, Japan)

[*1] Currently in the University of Chicago, USA
[*2] Currently in Nagoya University, Japan

This work has been conducted as a part of the "Dependable Embedded Operating System for Practical Use (DEOS)" project, funded by JST (Japan Science and Technology Agency).

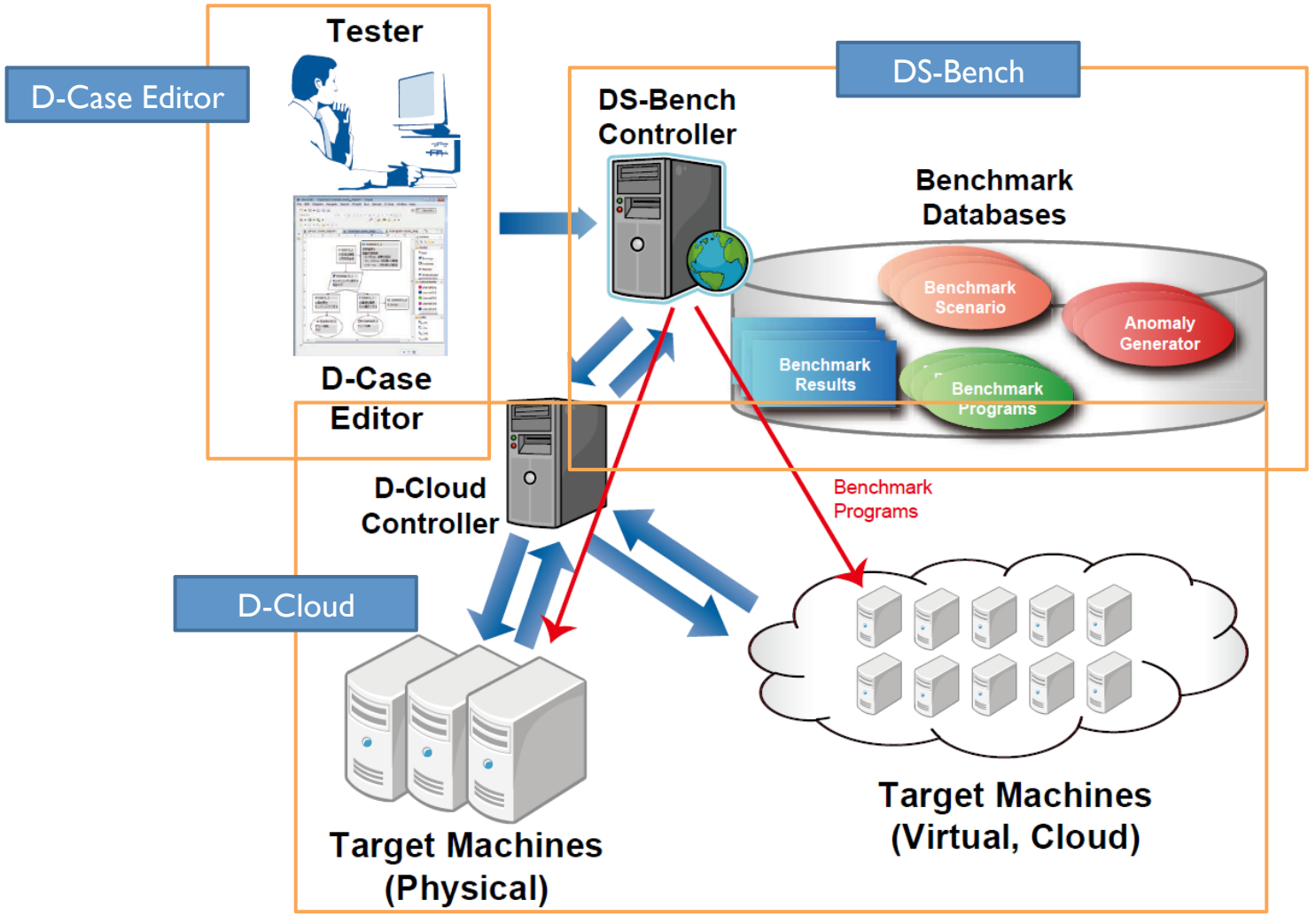
Introduction (1)

- Increasing demands for highly dependable system
  - Today's society totally depends on information systems, and suspension of services cost a lot
- A complex system involves a lot of stakeholders (e.g. developer, supplier, user, etc…)
- All of them must agree on what is the "dependability" for their system, e.g. minimum throughput or maximum latency
  - The dependability of the system should be expressed clearly and supported by clear evidences so that every stakeholder agrees that the system is in fact dependable

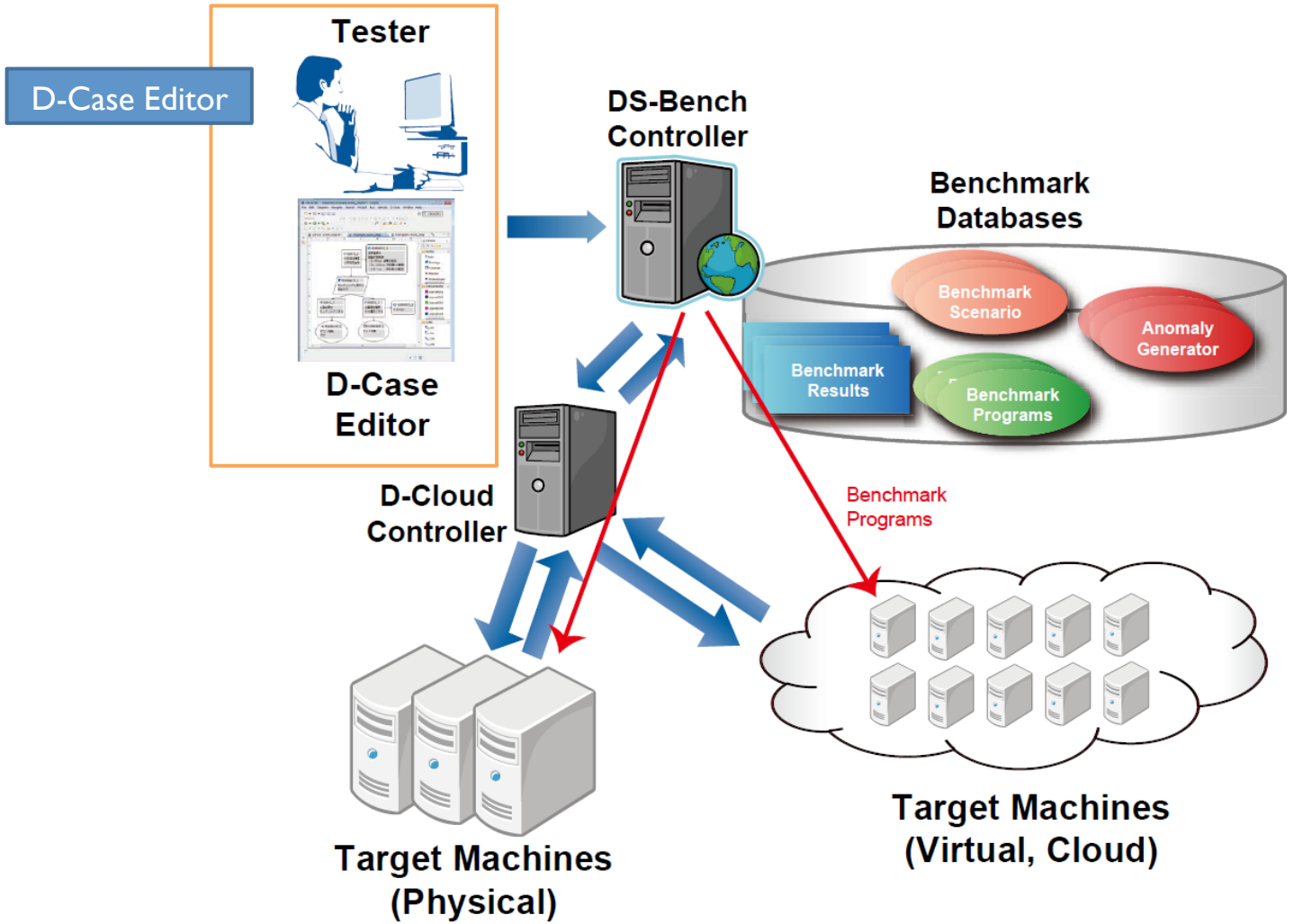
Introduction (2)

- Some argument on system's dependability may require quantitative evaluation of the system
- An automated benchmark testing tool is needed
  - Systems are getting more parallel and distributed
  - Testing takes much time and cost
- Results of the tests should be automatically collected as evidences of dependability

DS-Bench Toolset: Overview

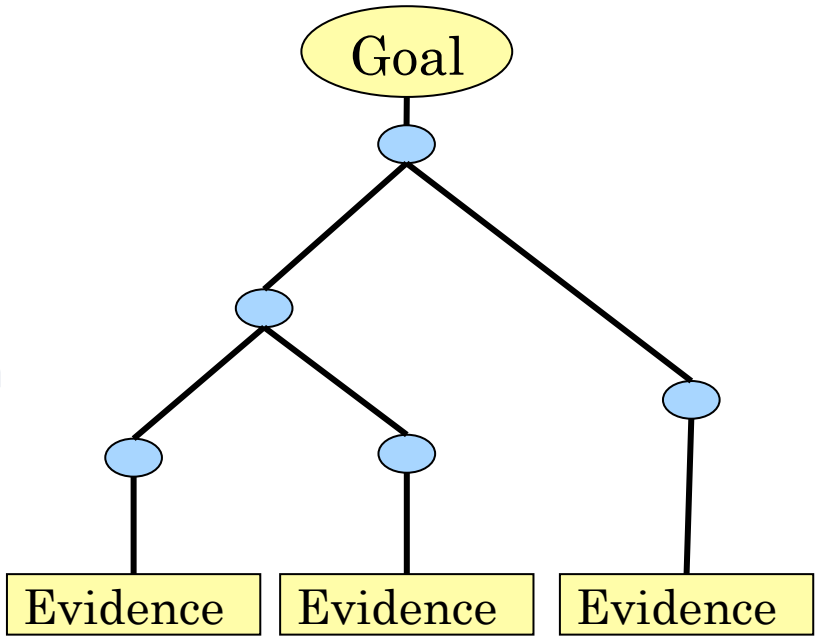


D-Case Editor



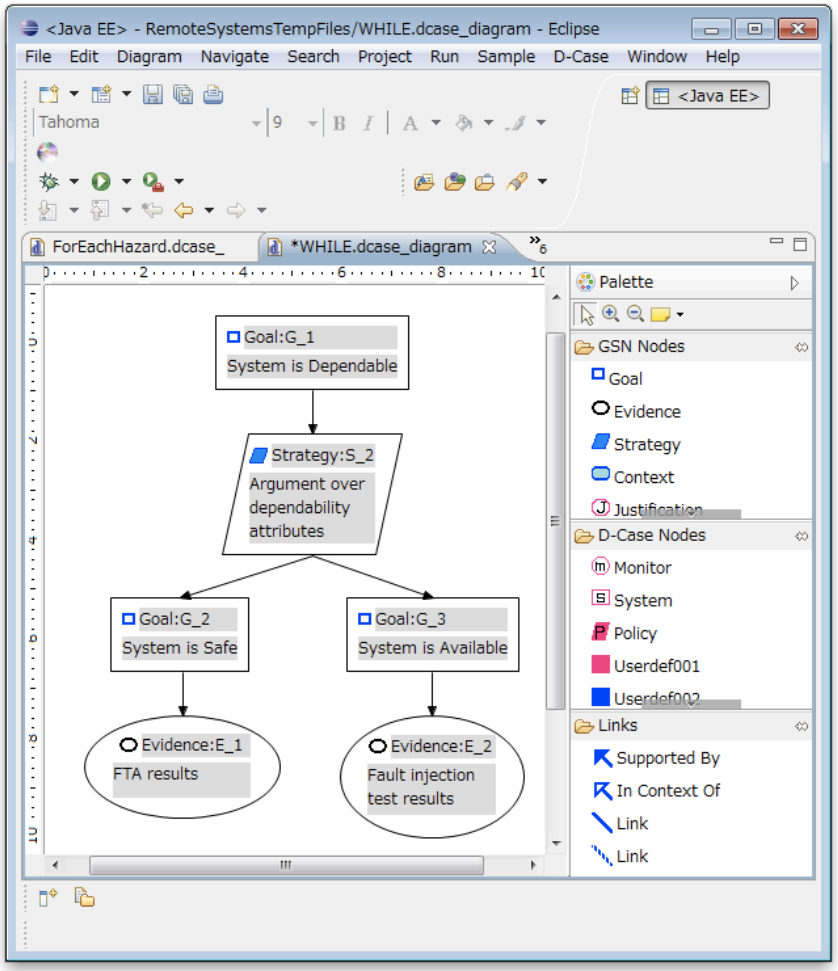
D-Case [Matsuno PRDC2010]

- A kind of Assurance Cases with development tools and runtime monitoring systems
- Assurance Cases
  - "A documented body of evidence that provides a convincing and valid argument that a system is adequately dependable for a given application in a given environment" [Adelard]
  - Becoming a standard for safety-critical systems
  - A Graphical notations GSN (Goal Structuring Notation)

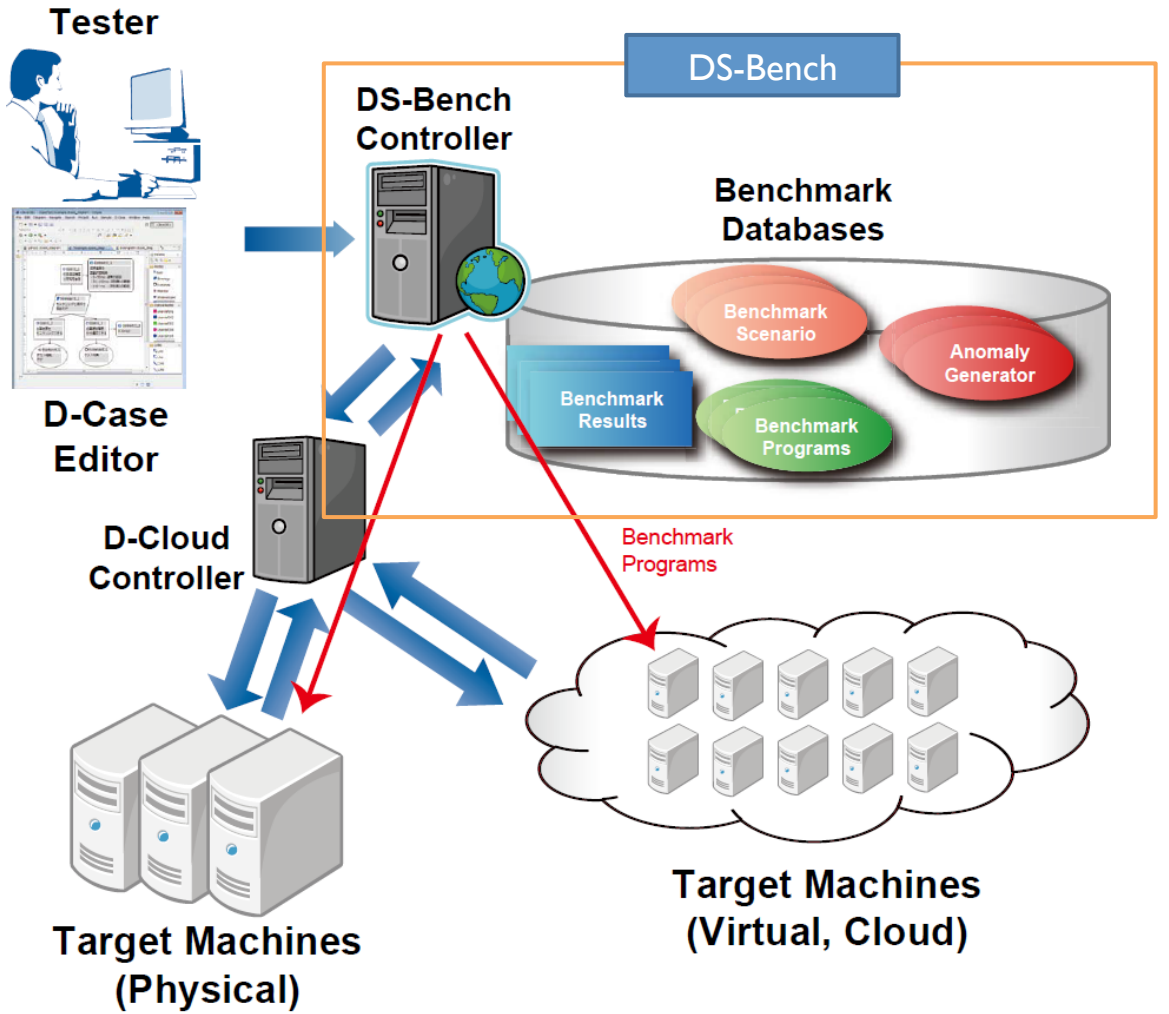


D-Case Editor

- A free assurance case editor
  - An Eclipse plugin using Eclipse GMF
  - Supports GSN
- Key Features
  - Variable type checking and pattern library [Matsuno QSIC2011]
  - Conducting benchmark tests using DS-Bench

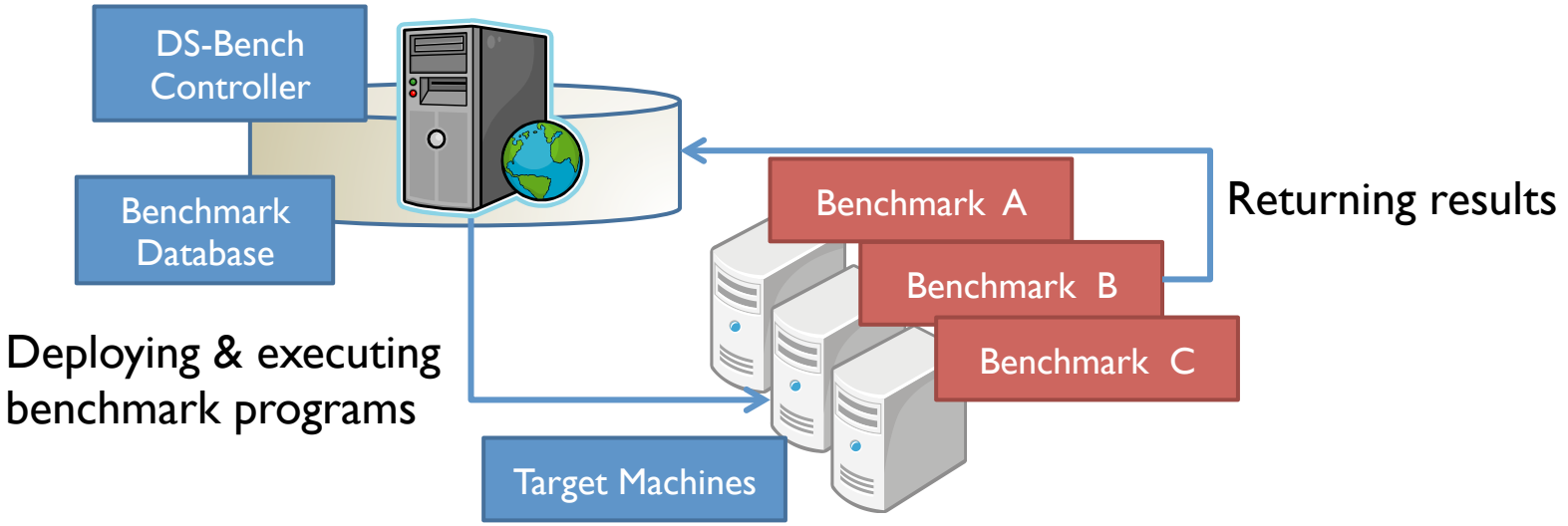


DS-Bench


Tester
DS-Bench
DS-Bench Controller
Benchmark Databases
Benchmark Scenario
Anomaly Generator
Benchmark Results
Benchmark Programs
D-Case Editor
D-Cloud Controller
Benchmark Programs
Target Machines (Physical)
Target Machines (Virtual, Cloud)

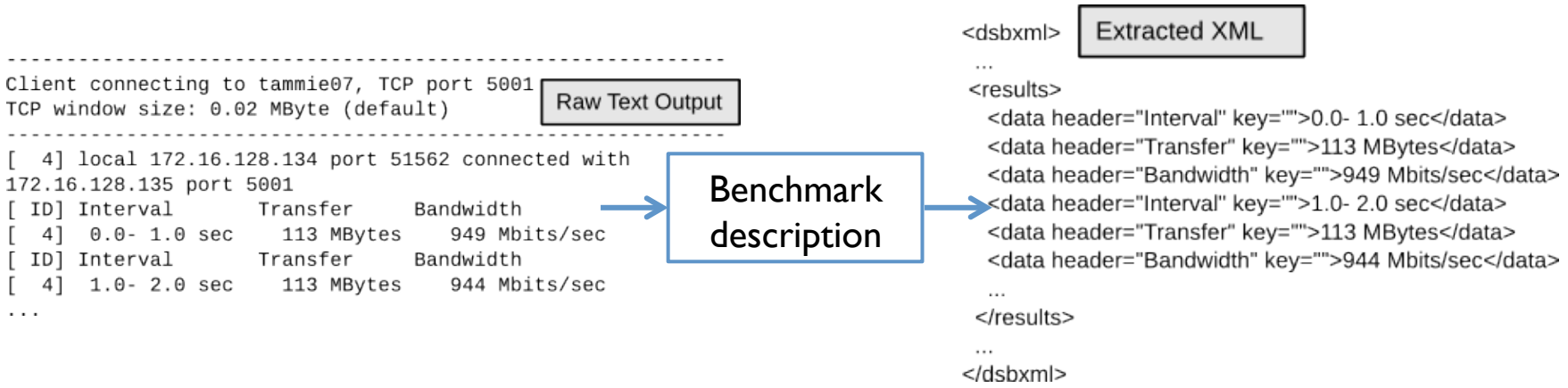
DS-Bench: Overview

- A framework for automatic benchmark tests
  - Dependability metrics are defined and measured by each program. DS-Bench itself does not define them.
- Supports multiple benchmark programs
  - Existing benchmark programs, as well as user-developed ones, can be executed on DS-Bench



DS-Bench: Benchmark Results

- Handling outputs from various benchmark programs
  - Output style of benchmark programs may vary
  - Usually the output is pre-formatted for human readability
- Benchmark description is prepared for each benchmark program
  - Describes cutting rules for interpreting the raw result so that a text table can be converted to a list of machine readable values
- Converted results are stored in an XML database



DS-Bench: Anomaly Loads

- Anomaly loads simulate several irregular situations that may occur to the system
  - E.g. Whole machine failure, device failure, performance degradation, …
- Essential for dependability benchmark testing
  - We want to know if the system is still dependable under such conditions

DS-Bench: Pre-installed Programs

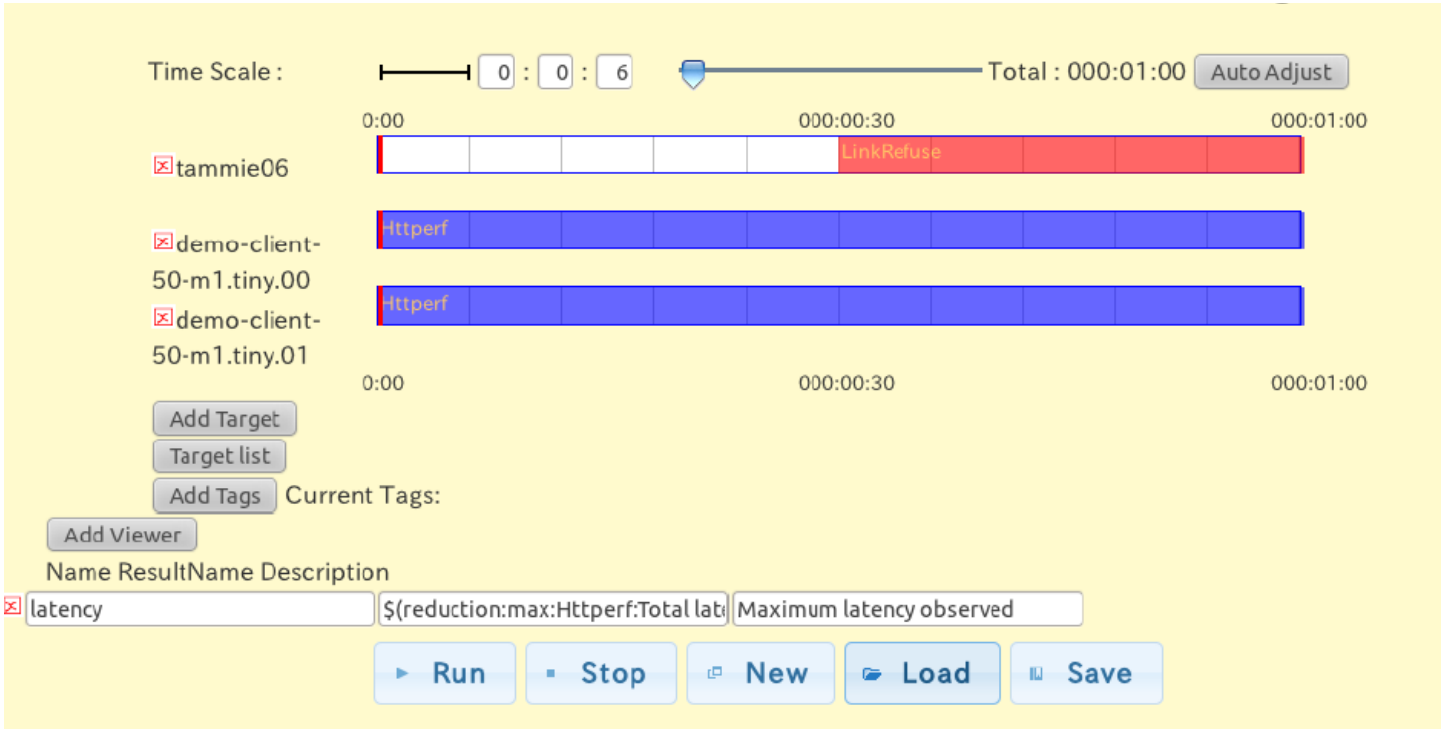
Name	Description
bonnie++	I/O benchmark
lmbench	Generic performance benchmark
hackbench	System benchmark; creates a lot of processes
httperf	Measures performance of HTTP servers
iperf	Measures network bandwidth
cpustress	Stresses CPU; just consumes CPU time
memstress	Stresses memory; just consumes memory
netcmd	Injecting network anomalies; delay, packet drop, reordering
terminator	Kills a process

Each program may be used as a benchmark program or anomaly generator. Some of them may be used as both.

e.g. bonnie++ can be used as a benchmark program to measure I/O bandwidth, as well as an anomaly generator that consumes I/O bandwidth.

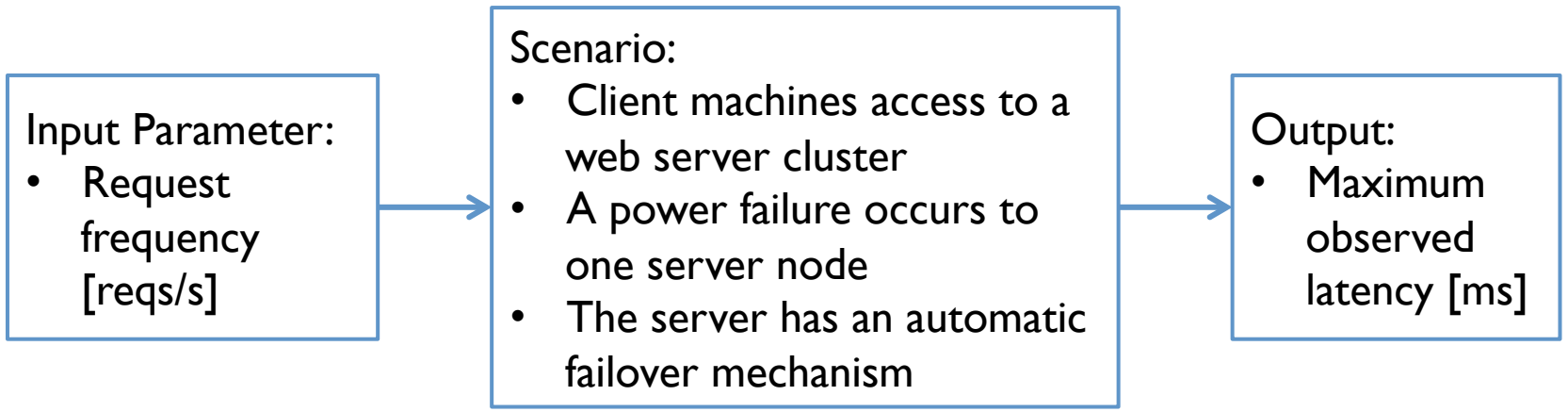
DS-Bench: Benchmark Scenario (1)

- Benchmark programs and anomaly loads are executed concurrently in a specific timing

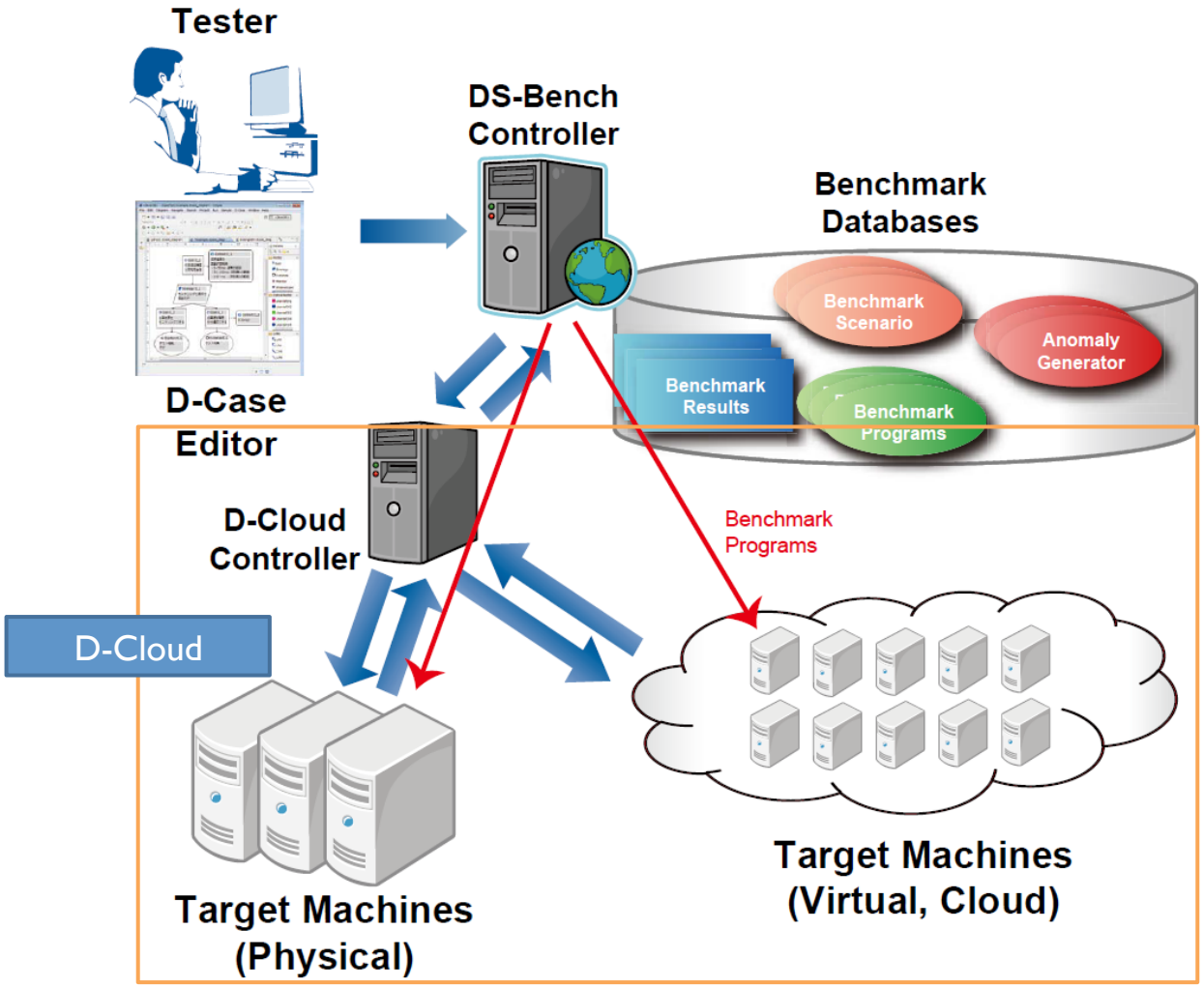


DS-Bench: Benchmark Scenario (2)

- Each scenario may define input parameters and an output result
  - Input parameters are adjustable from D-Case Editor
  - An output result can be obtained from D-Case Editor
- Example:



D-Cloud


Tester
DS-Bench Controller
Benchmark Databases
Benchmark Scenario
Anomaly Generator
Benchmark Results
Benchmark Programs
D-Case Editor
D-Cloud Controller
Benchmark Programs
D-Cloud
Target Machines (Physical)
Target Machines (Virtual, Cloud)

D-Cloud: Overview

- D-Cloud manages hardware resources needed for conducting benchmark tests
- Two types of computing resources are provided
  - Physical machines
    - For performance-sensitive tests
  - Virtual machines
    - Managed by OpenStack, a management software for private clouds
    - An arbitrary number of virtual machines can be created simultaneously
    - For functional, performance-insensitive tests

D-Cloud: Fault Injection

- D-Cloud also performs fault injection
  - Adding anomalies from the outside of target machines
  - Mostly requires special equipment (e.g. intelligent PDUs, IPMI, SNMP-enabled switch), or special software (e.g. VMM)
- FaultVM [Banzai CLOUD2010][Hanawa PRDC2010]
  - A virtual machine monitor that comes with a hardware failure simulation feature
  - Based on QEMU, an open source full system emulator
- Fault injection functions are exported to DS-Bench and can be used as anomaly generators
  - i.e. These fault injectors can be put in a benchmark scenario

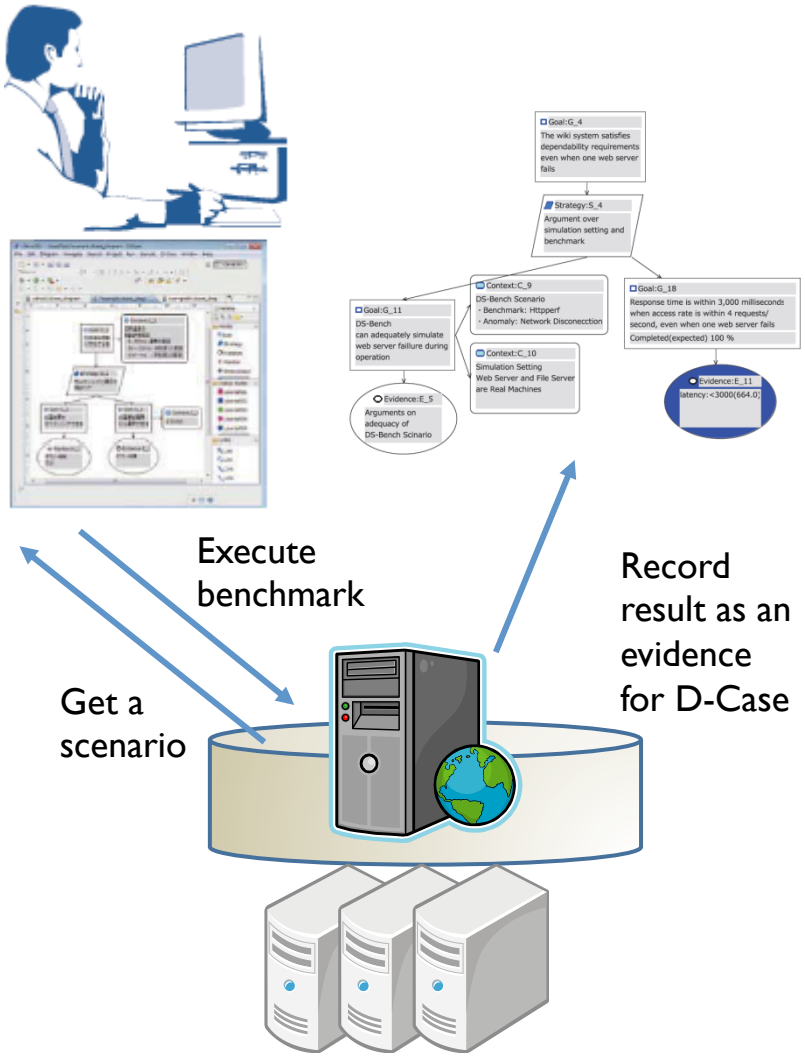
D-Cloud: List of Fault Injection Methods

Name	Target	Description
LinkRefuse	Physical Machine	Shuts down a network switch port
SupplyRefuse	Physical Machine	Shuts down a power supply to a target
IPMICtrl	Physical Machine	Controls a power status of a target via IPMI
VMMemFlip	Virtual Machine	Injects memory flip into VM
VMStop	Virtual Machine	Pauses a VM
VMNicFault	Virtual Machine	Injects a various fault into VM's virtual NIC device

Fault injection methods for physical machines are implemented as a program, and can be added by users just as benchmark programs

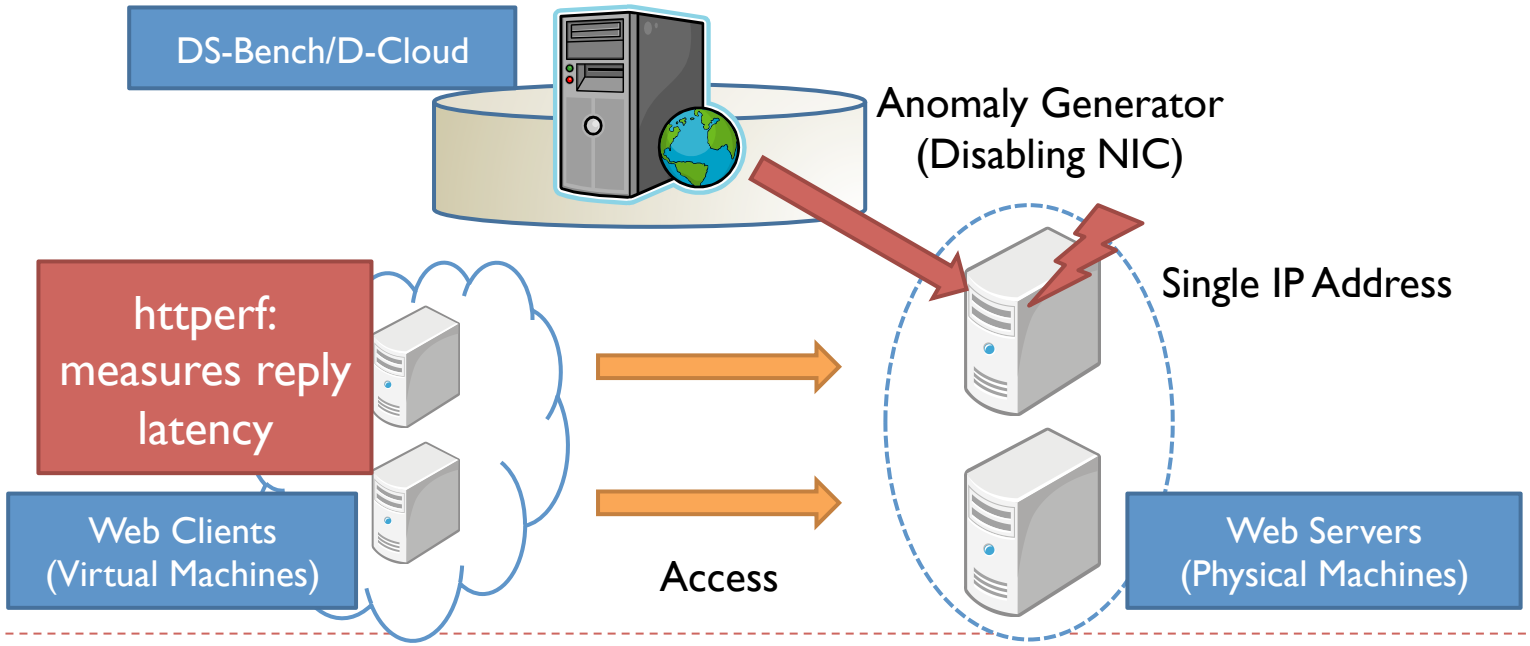
Typical Workflow with the Toolset

1. Determine the required dependability metrics and describe it in D-Case
2. Find a suitable benchmark scenario in DS-Bench database
   - If not exist, create new one
3. Adjust parameters and run the benchmark test from D-Case Editor
4. Result is returned to D-Case Editor and recorded as an evidence for D-Case

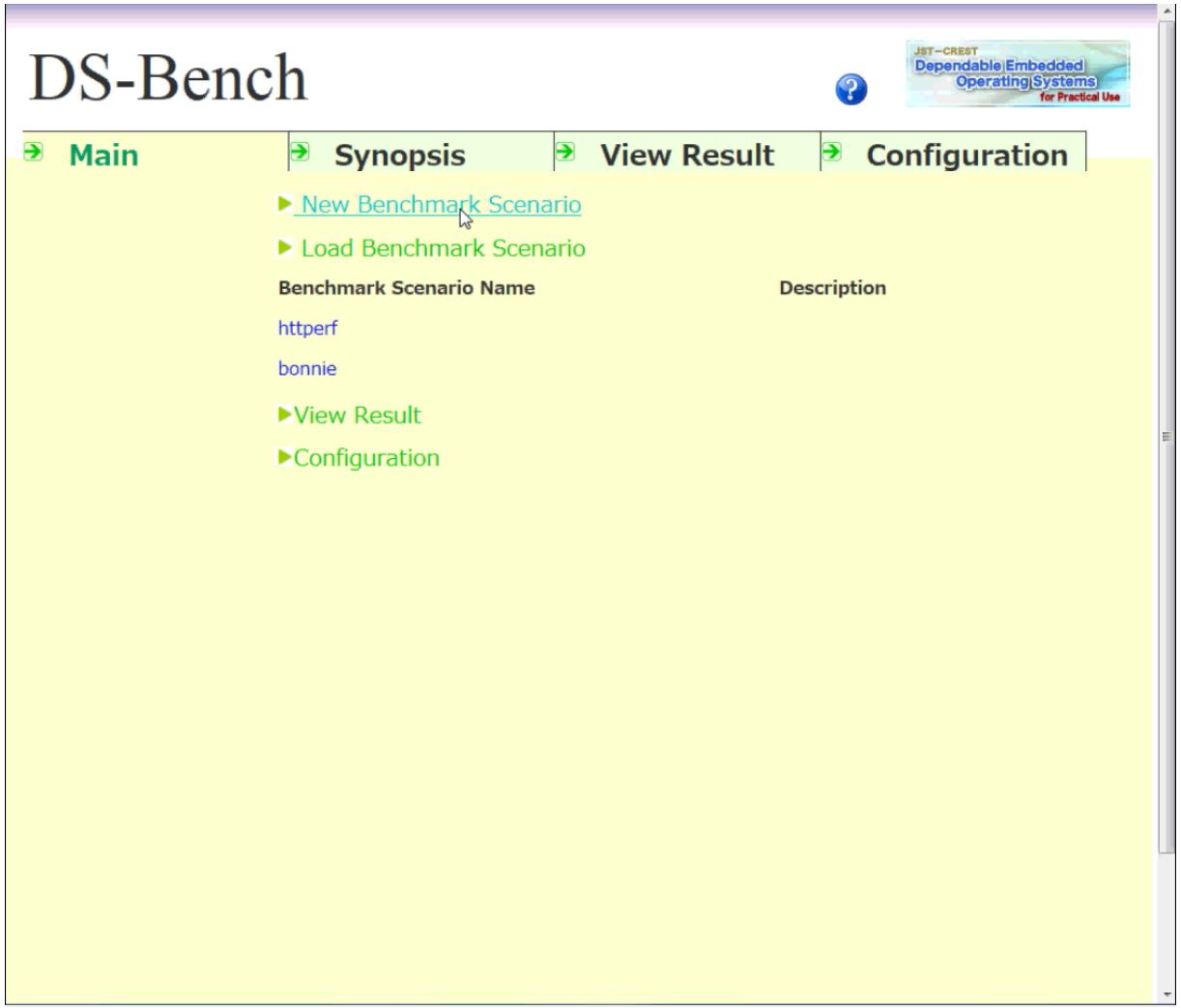


Demonstration

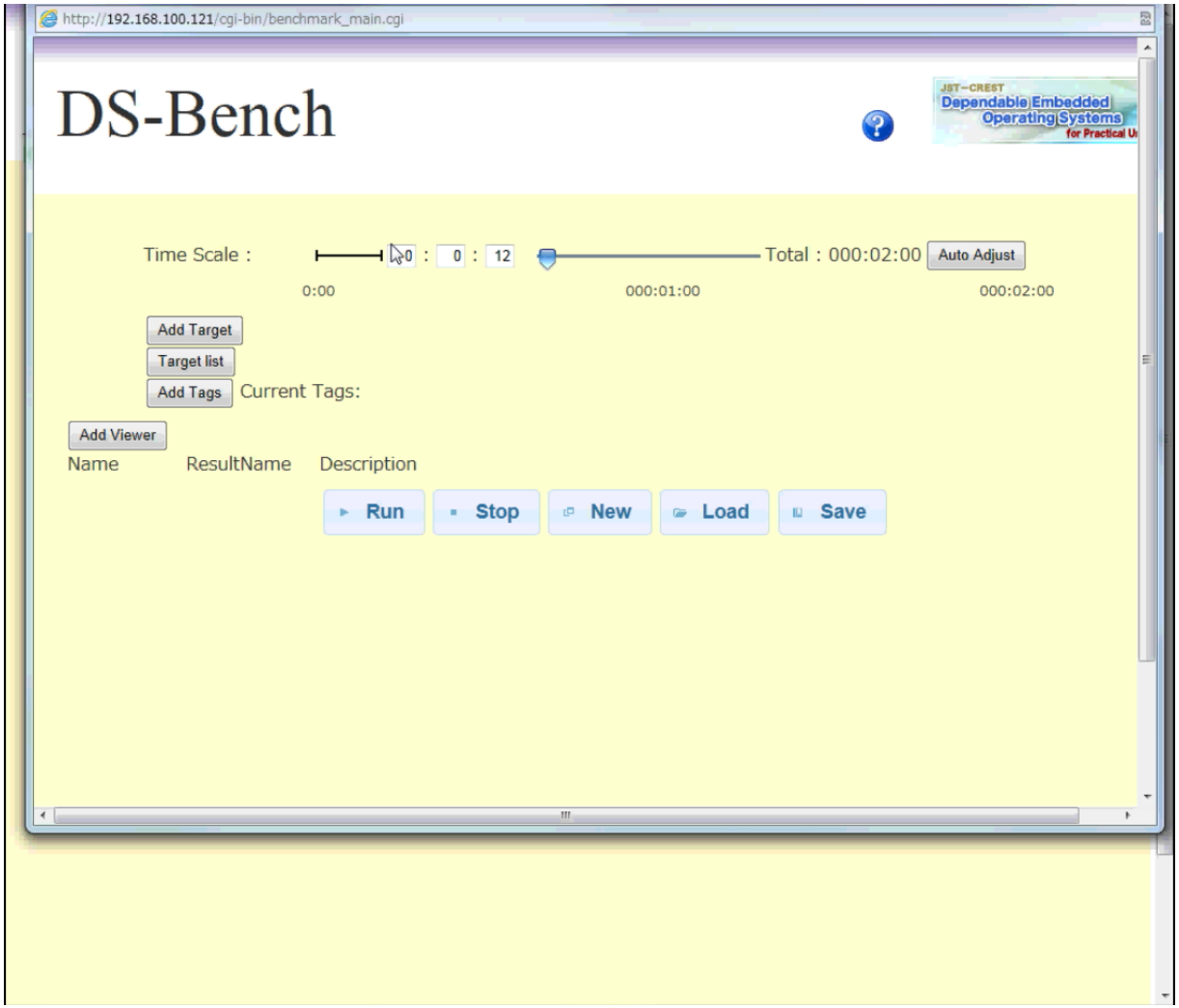
- Evaluating a web server system
- Dependability requirement
  - Keep the access latency lower than 3s even one server node fails
- Environment
  - The server provides a single system view using SSPA, a load balancing/high availability mechanism [Fujita HASE2011]



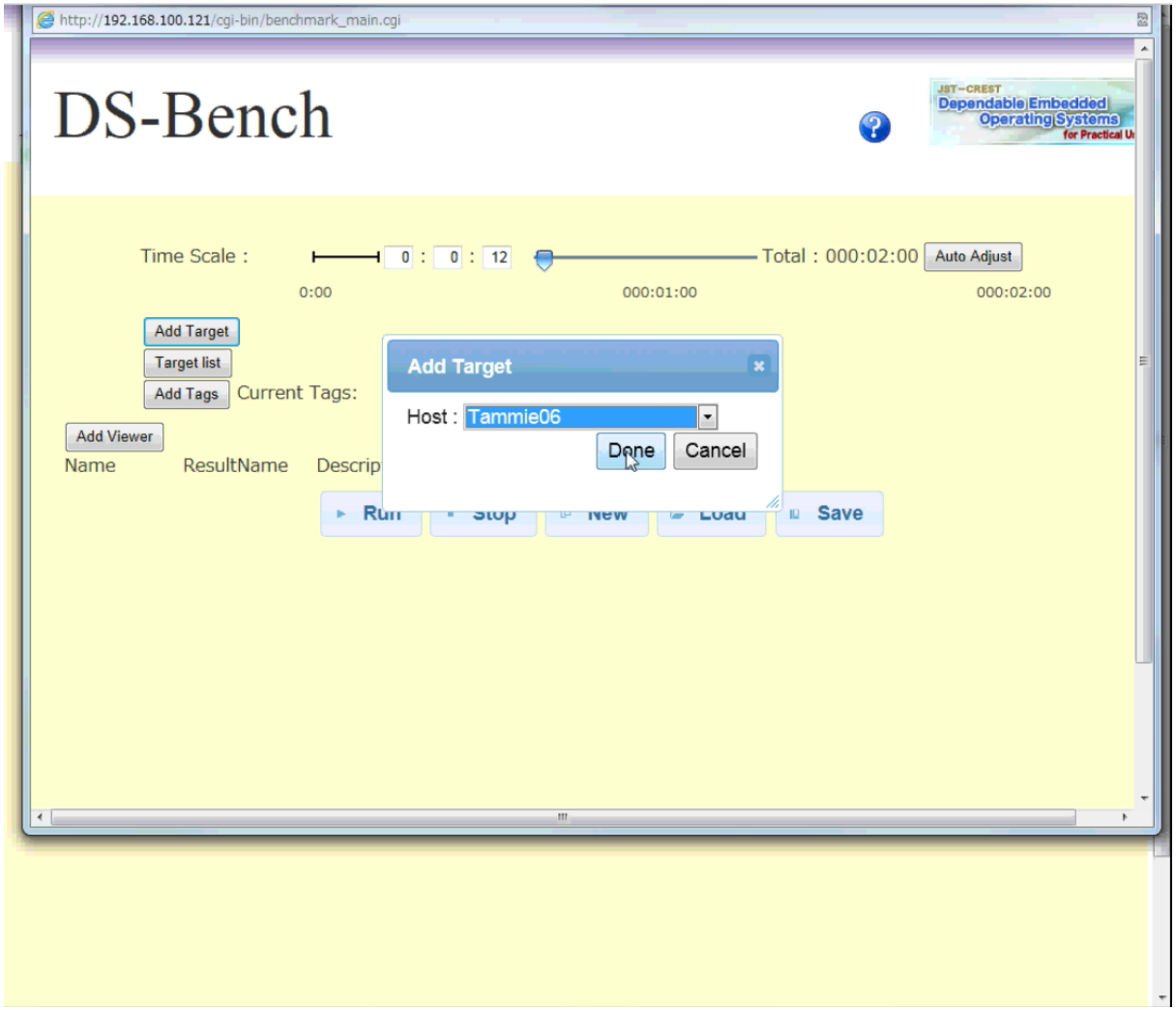
Demo: Creating a New Scenario



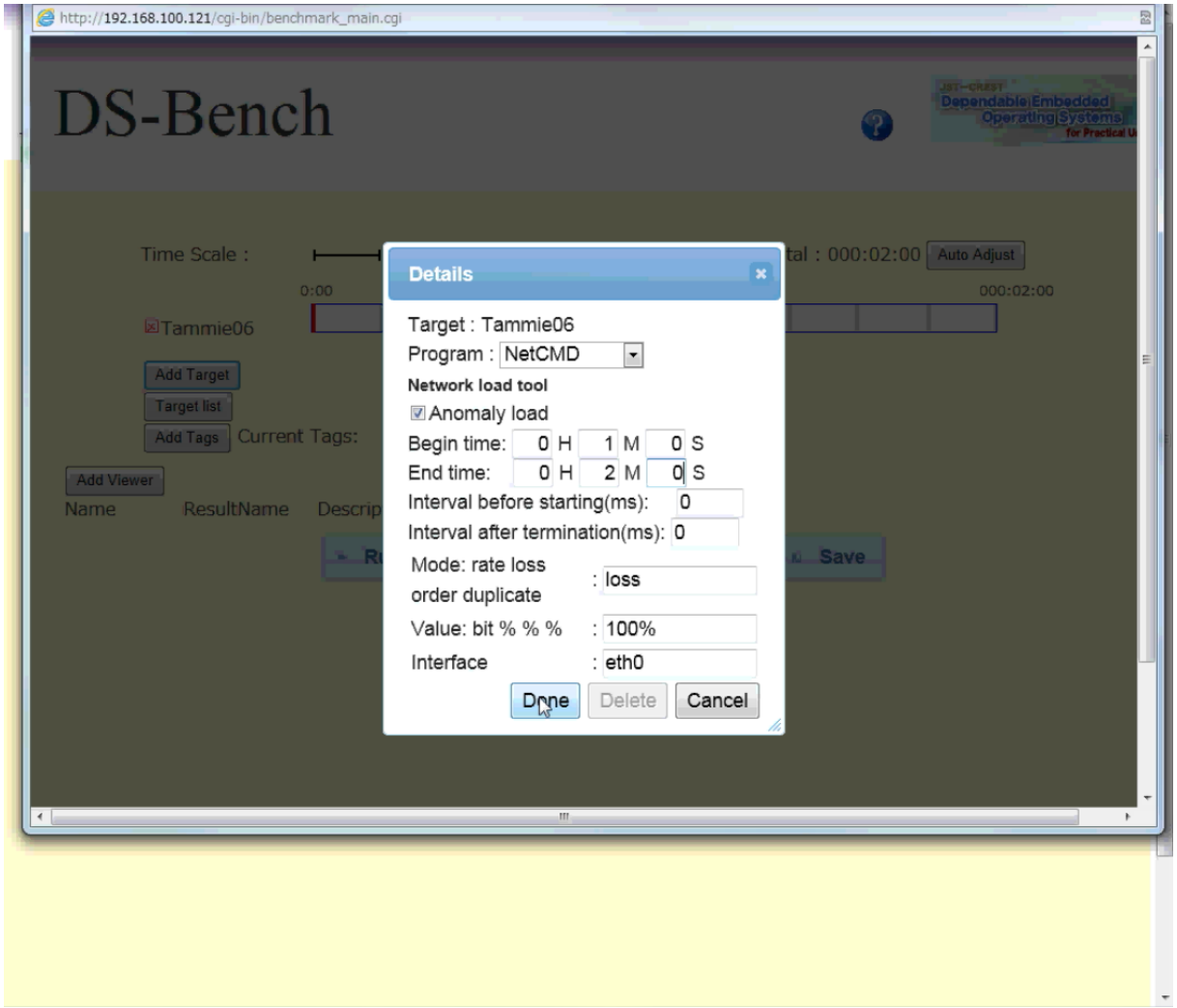
Demo: Creating a New Scenario



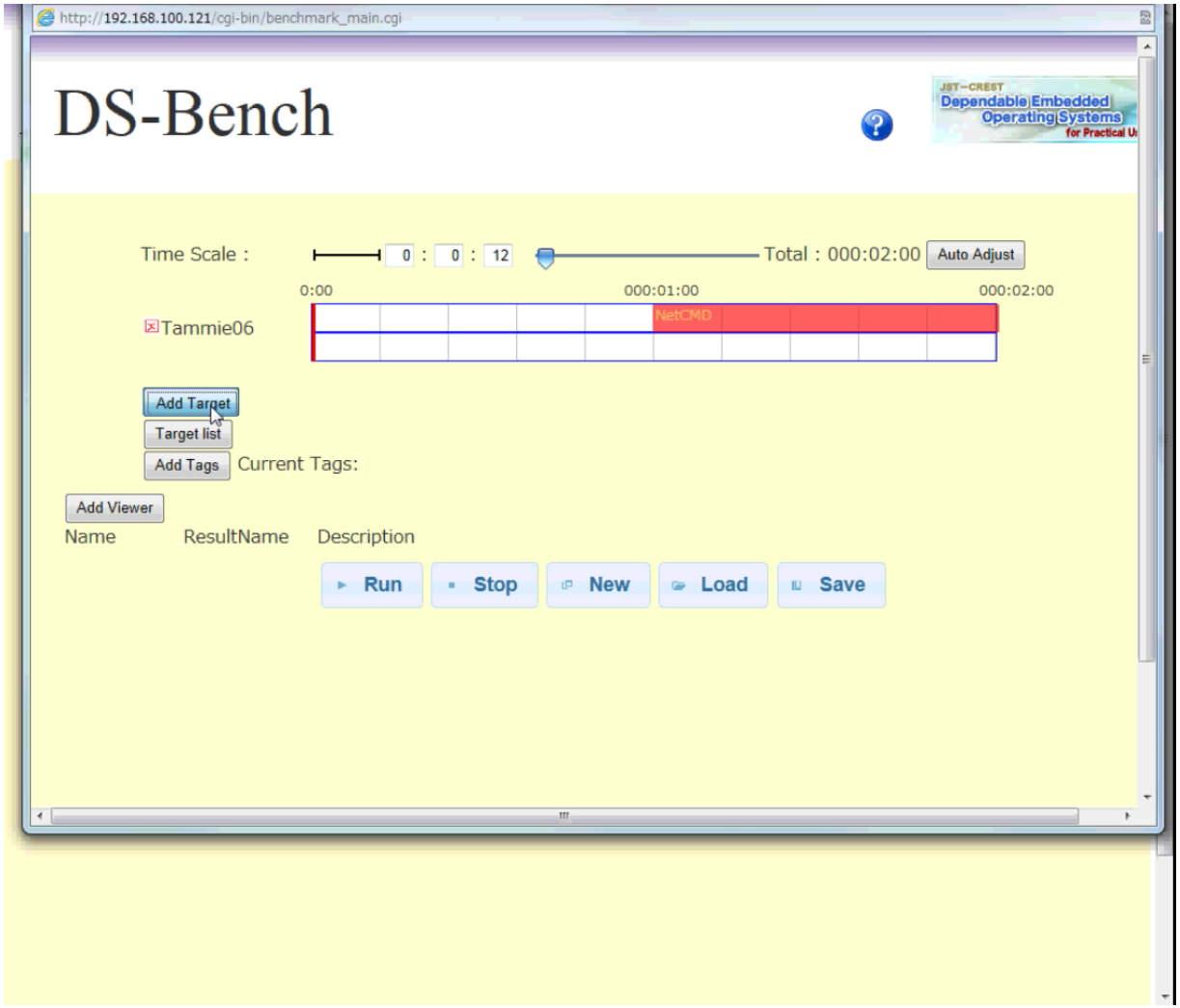
Demo: Creating a New Scenario



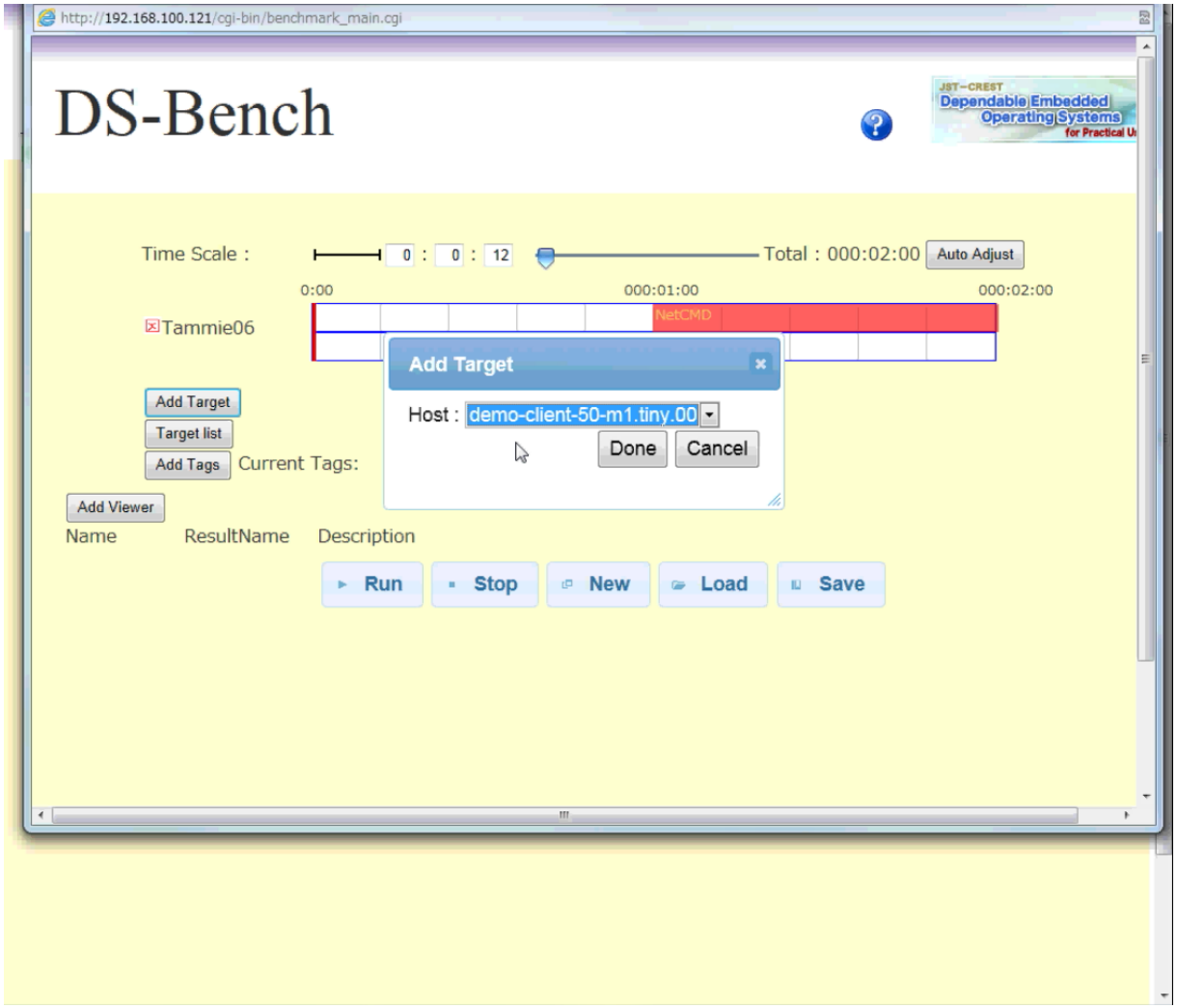
Demo: Creating a New Scenario



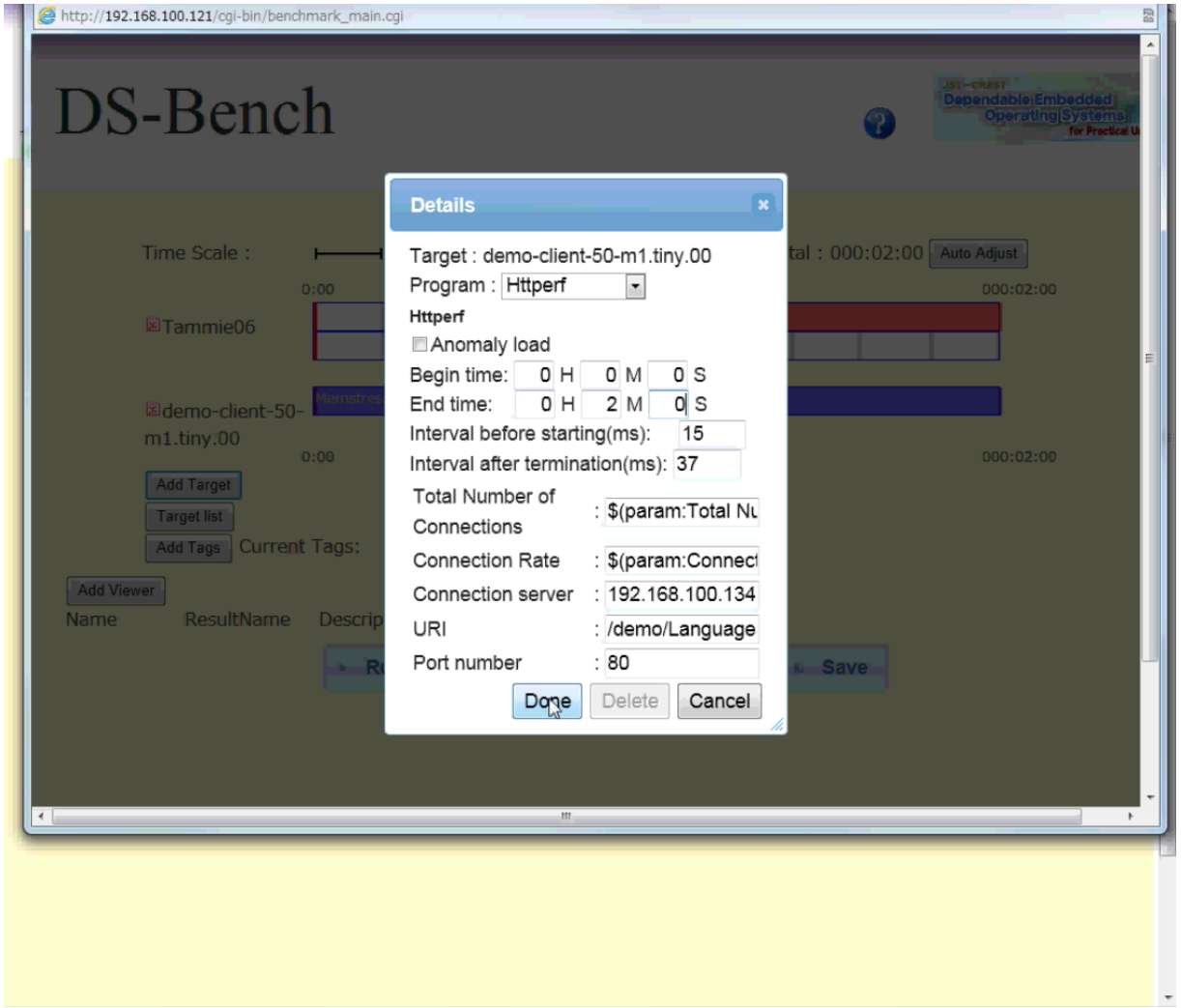
Demo: Creating a New Scenario



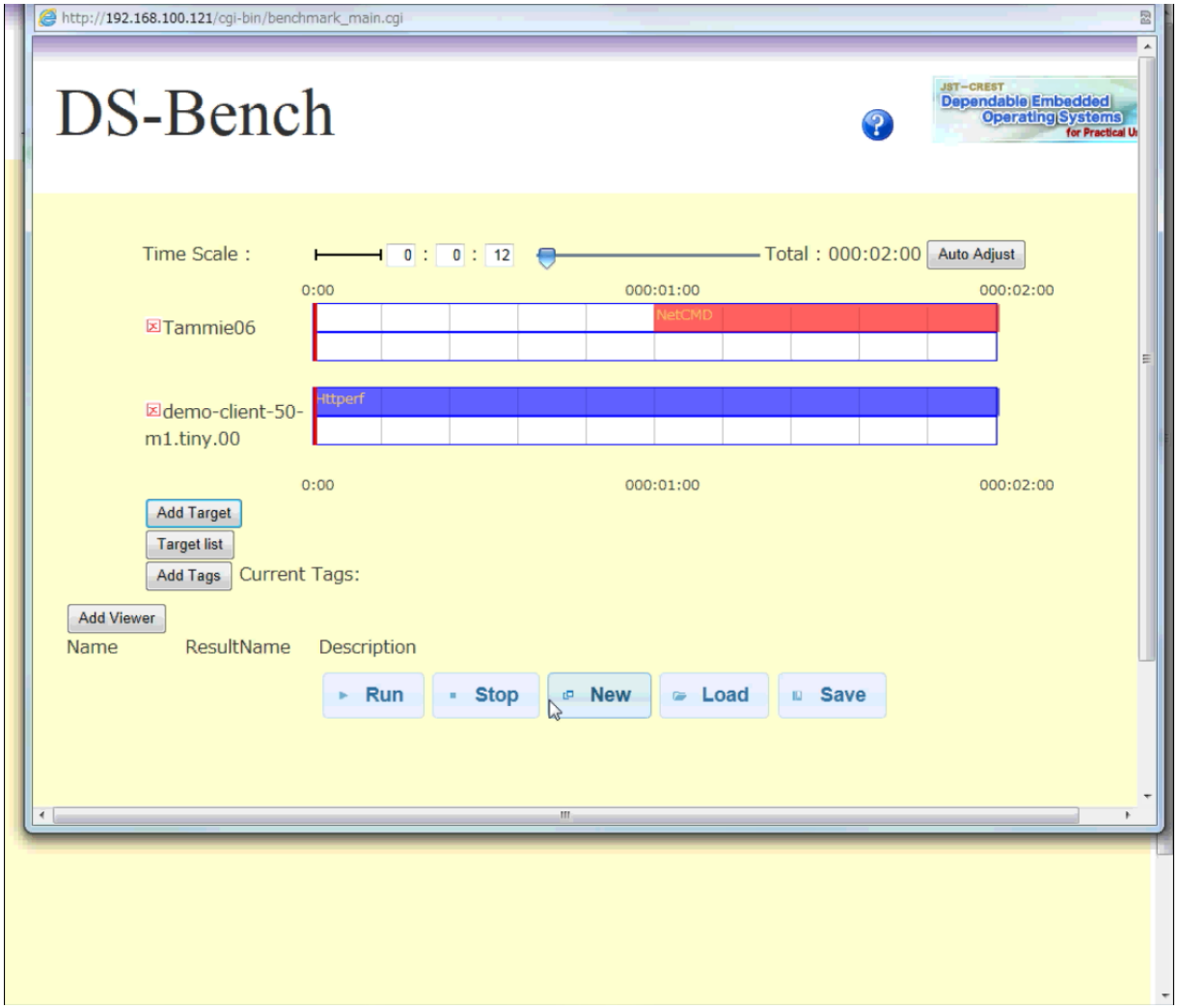
Demo: Creating a New Scenario



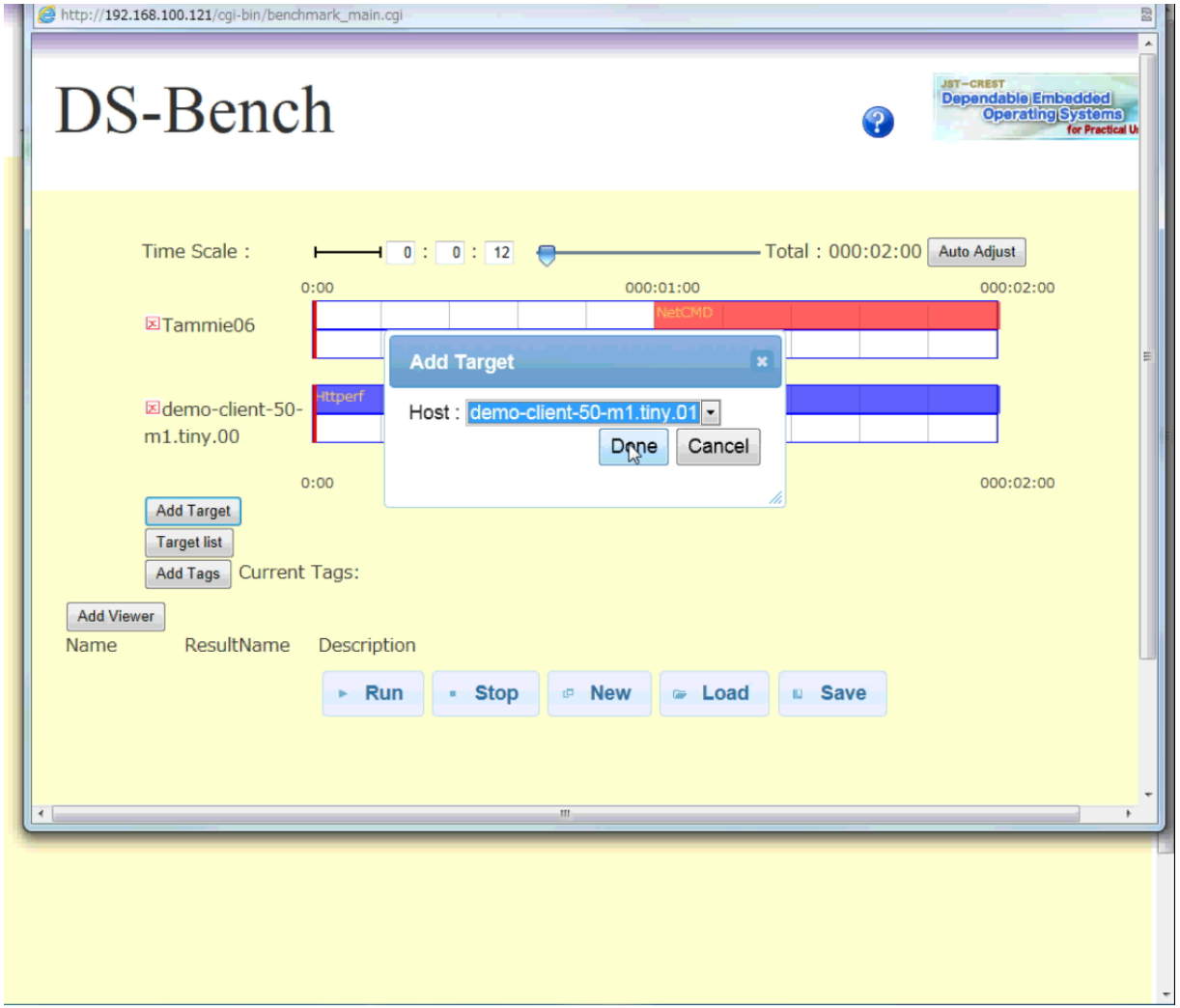
Demo: Creating a New Scenario



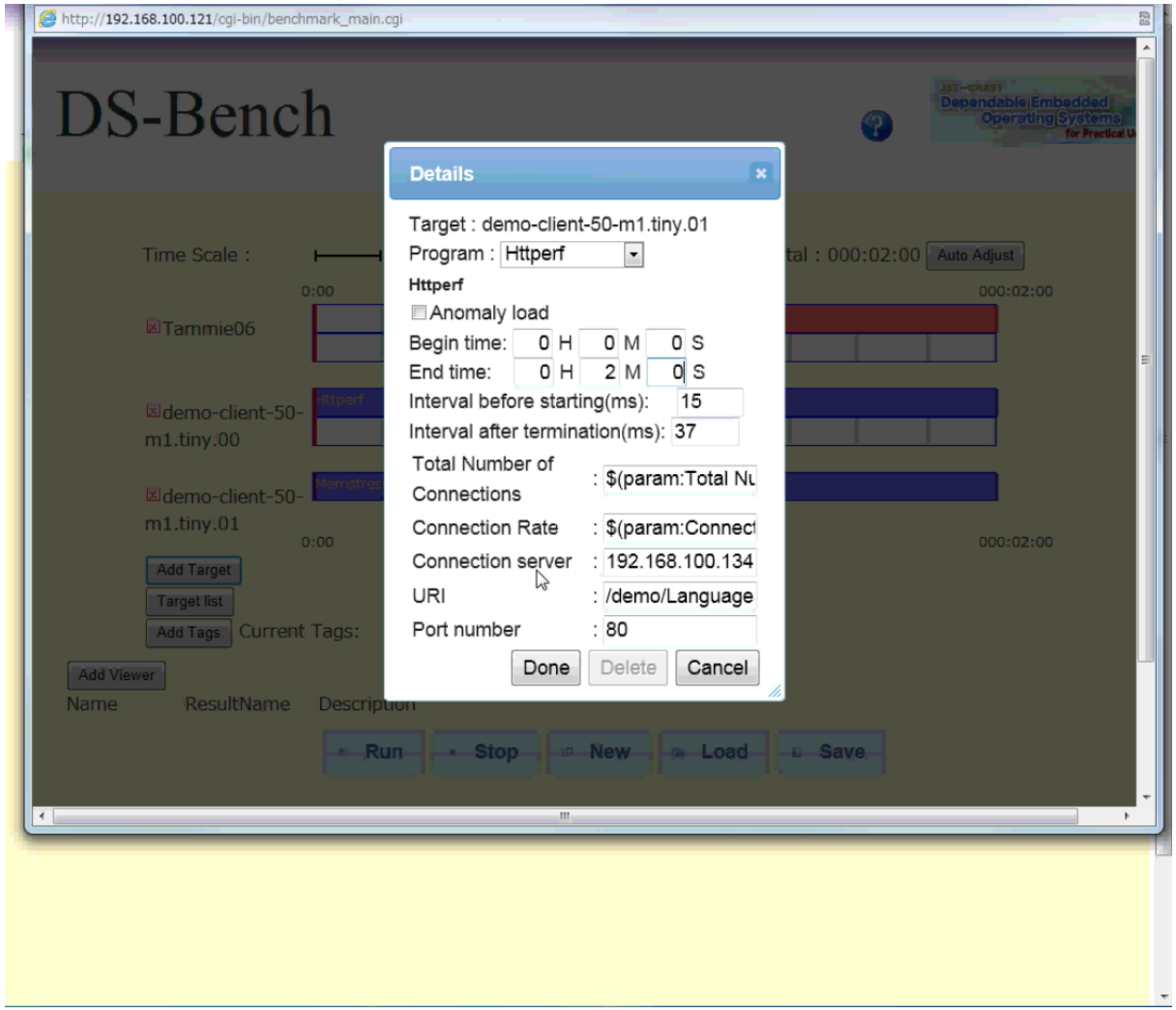
Demo: Creating a New Scenario



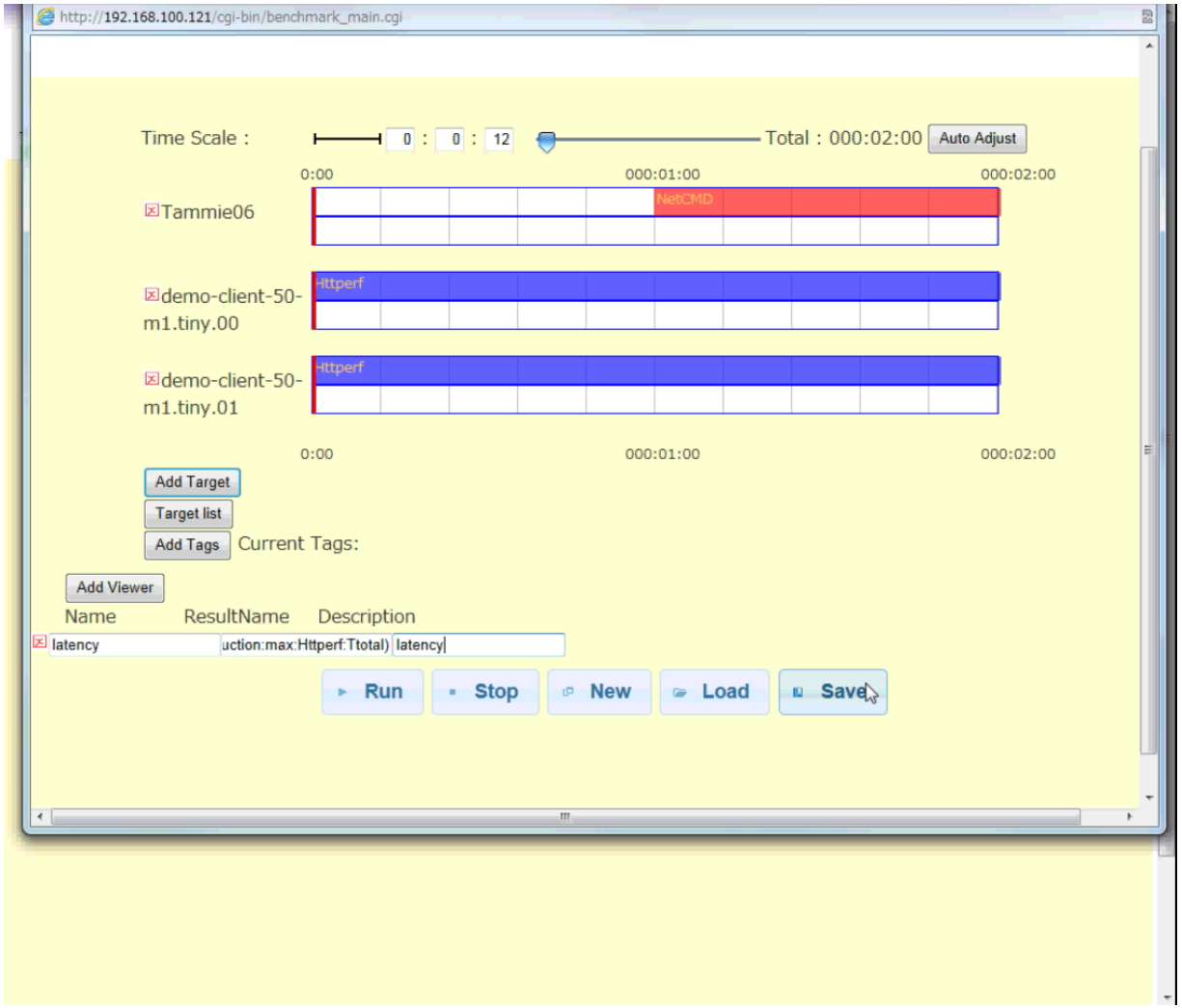
Demo: Creating a New Scenario



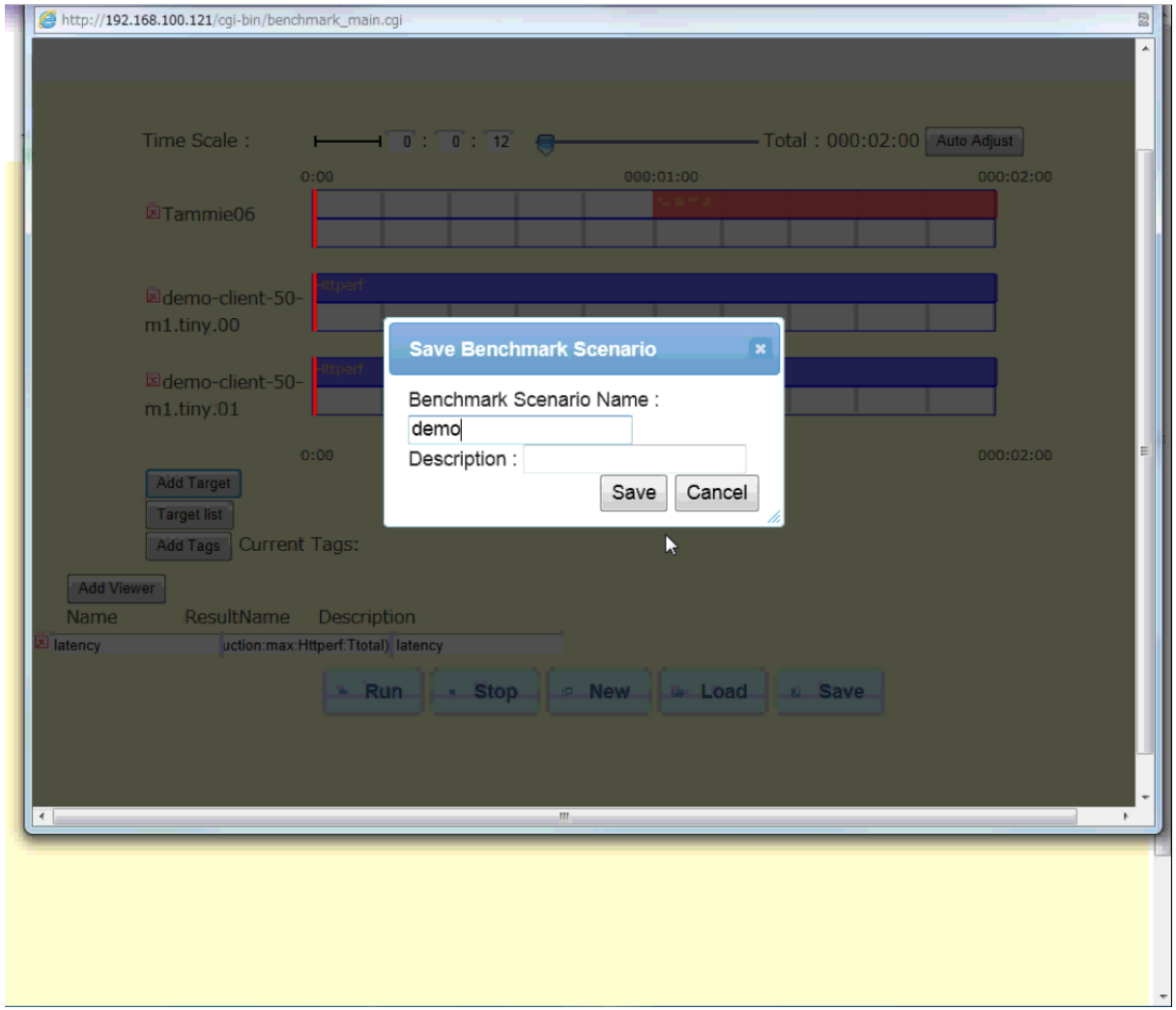
Demo: Creating a New Scenario



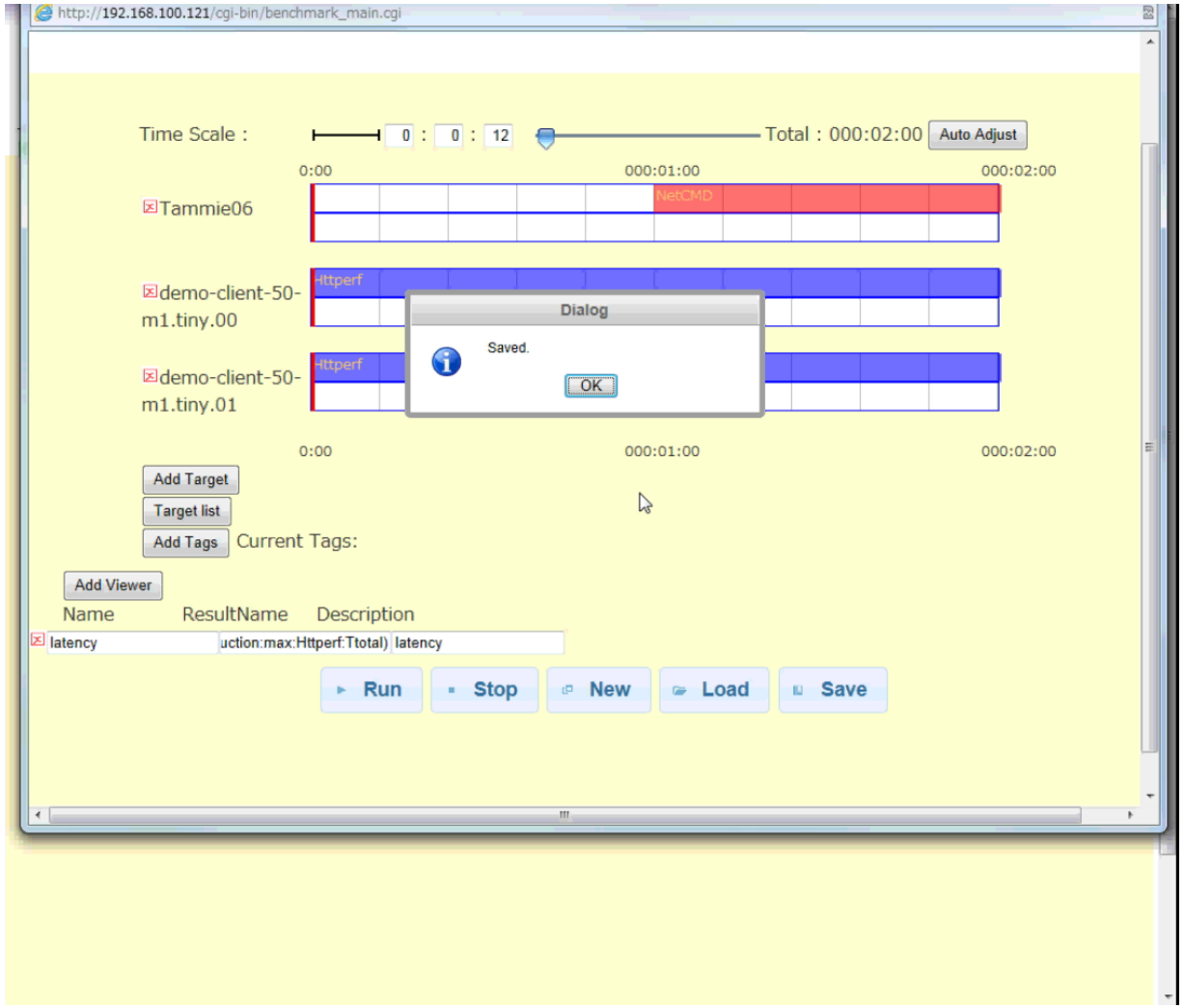
Demo: Creating a New Scenario



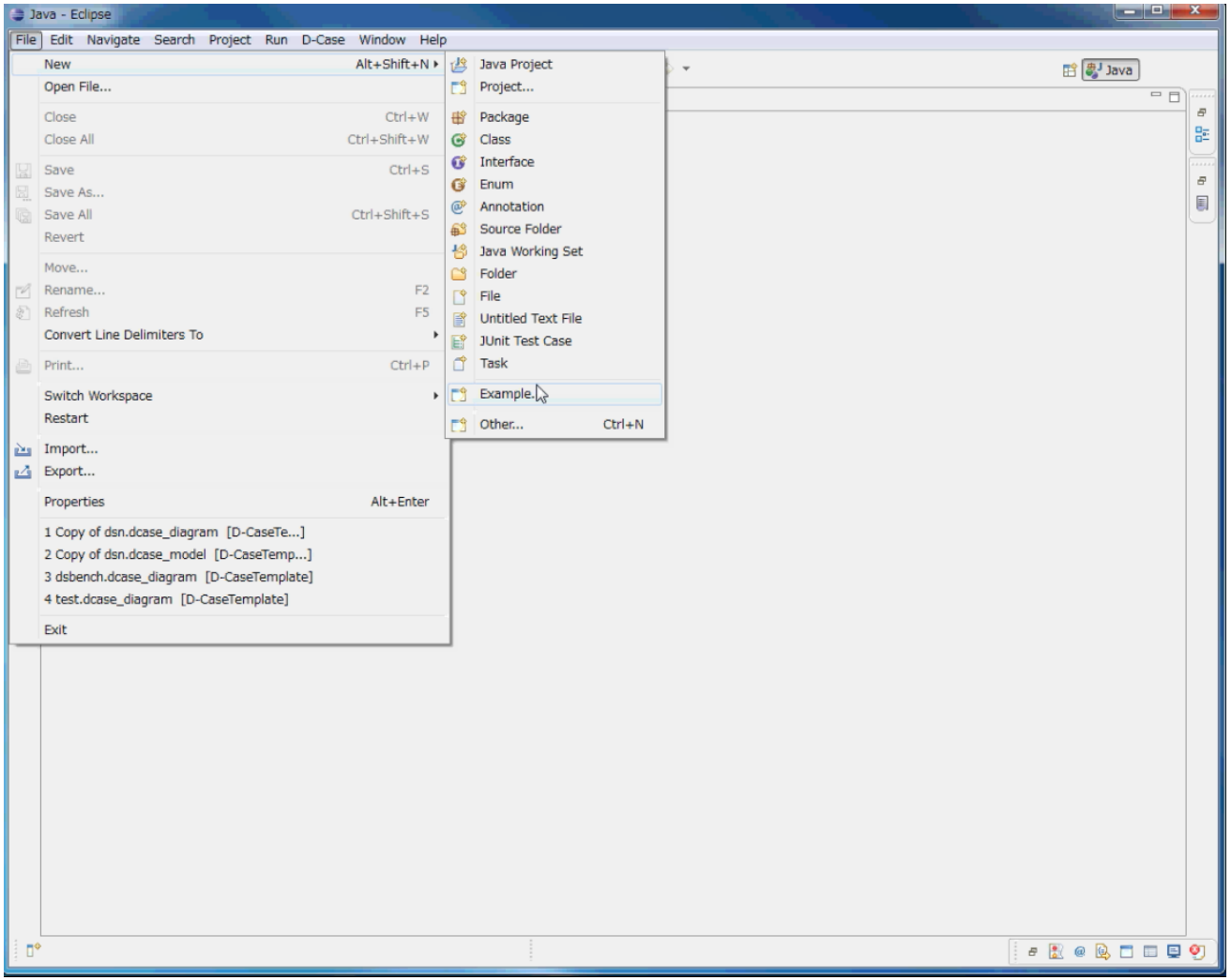
Demo: Creating a New Scenario



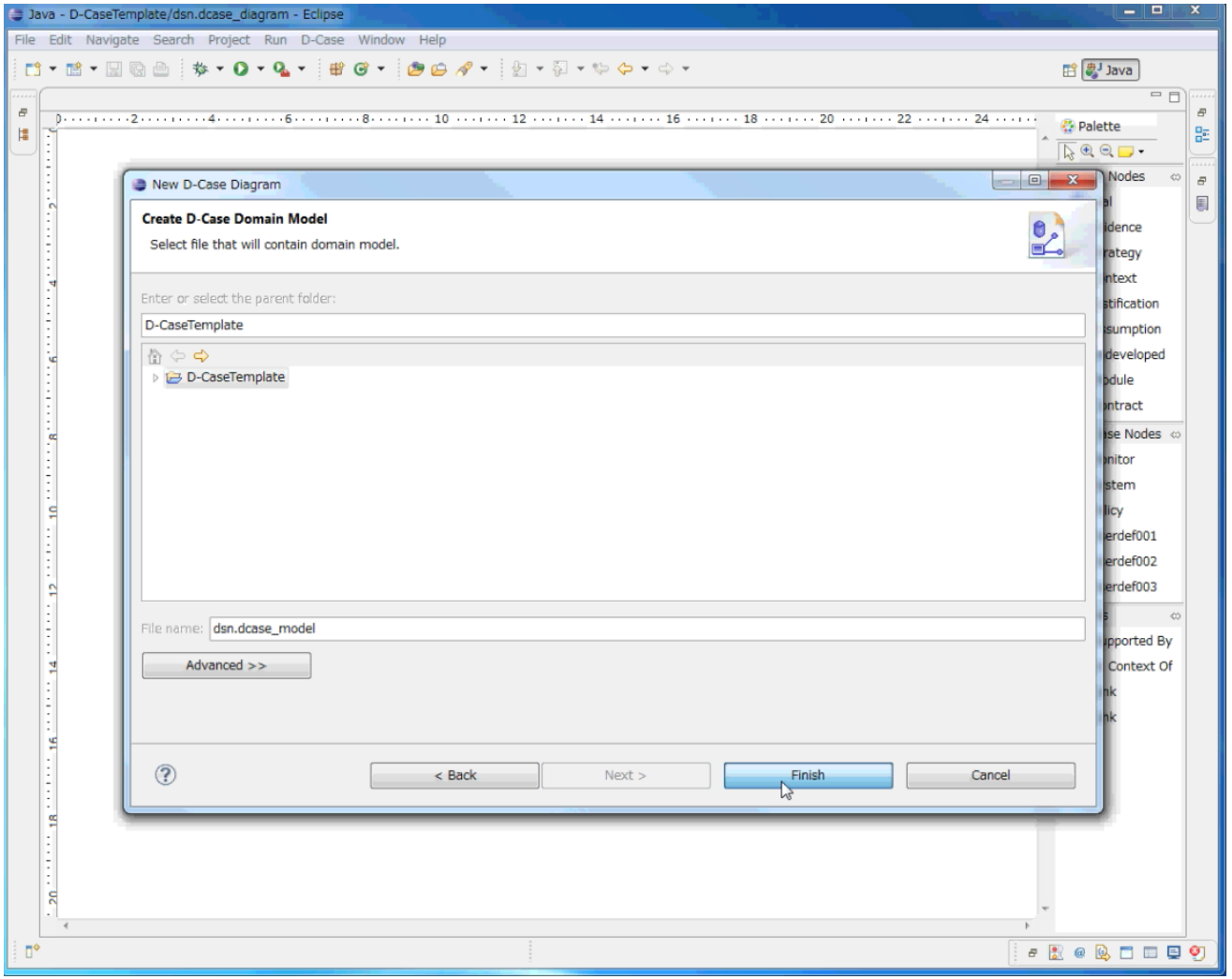
Demo: Creating a New Scenario



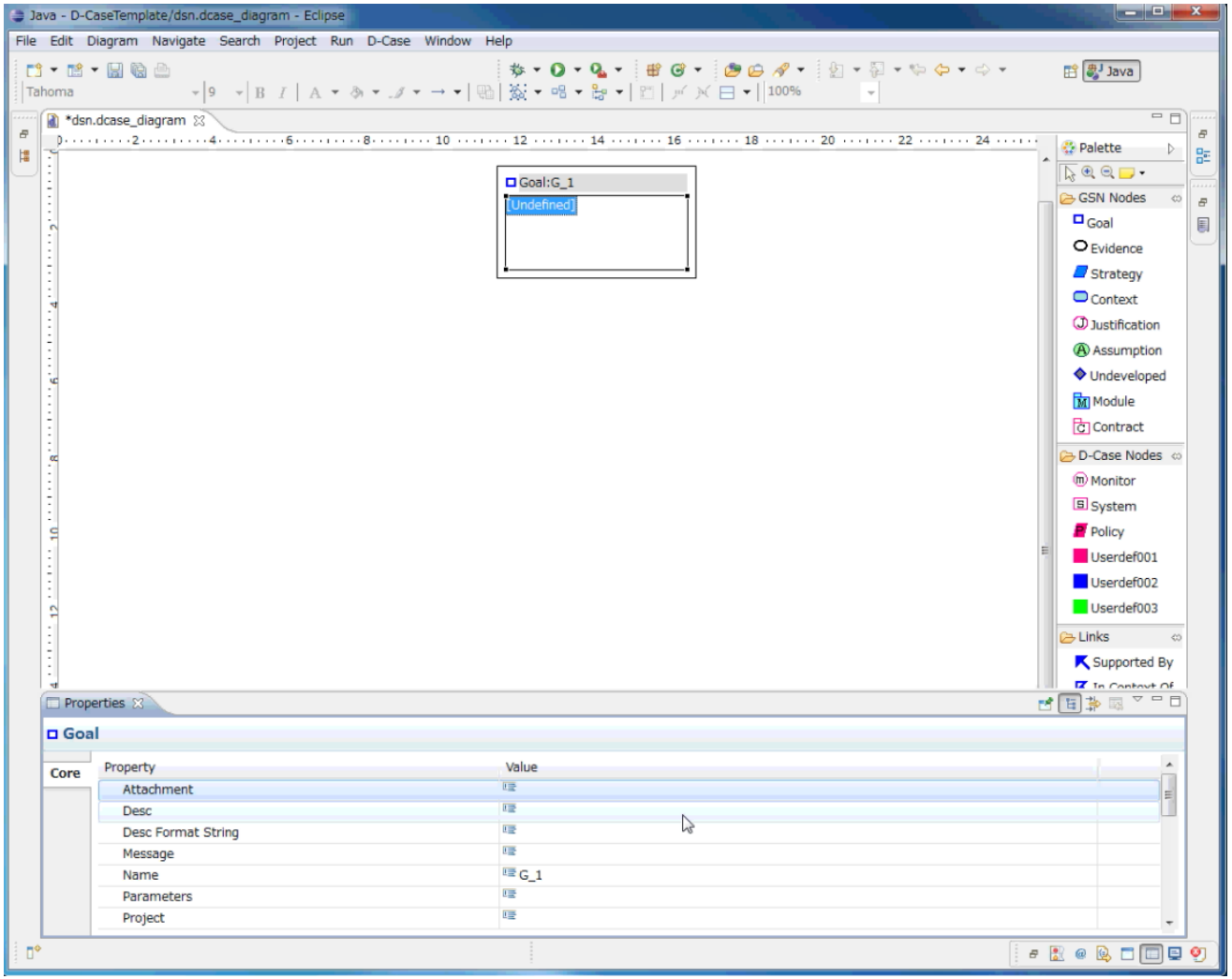
Demo: Creating a D-Case Diagram



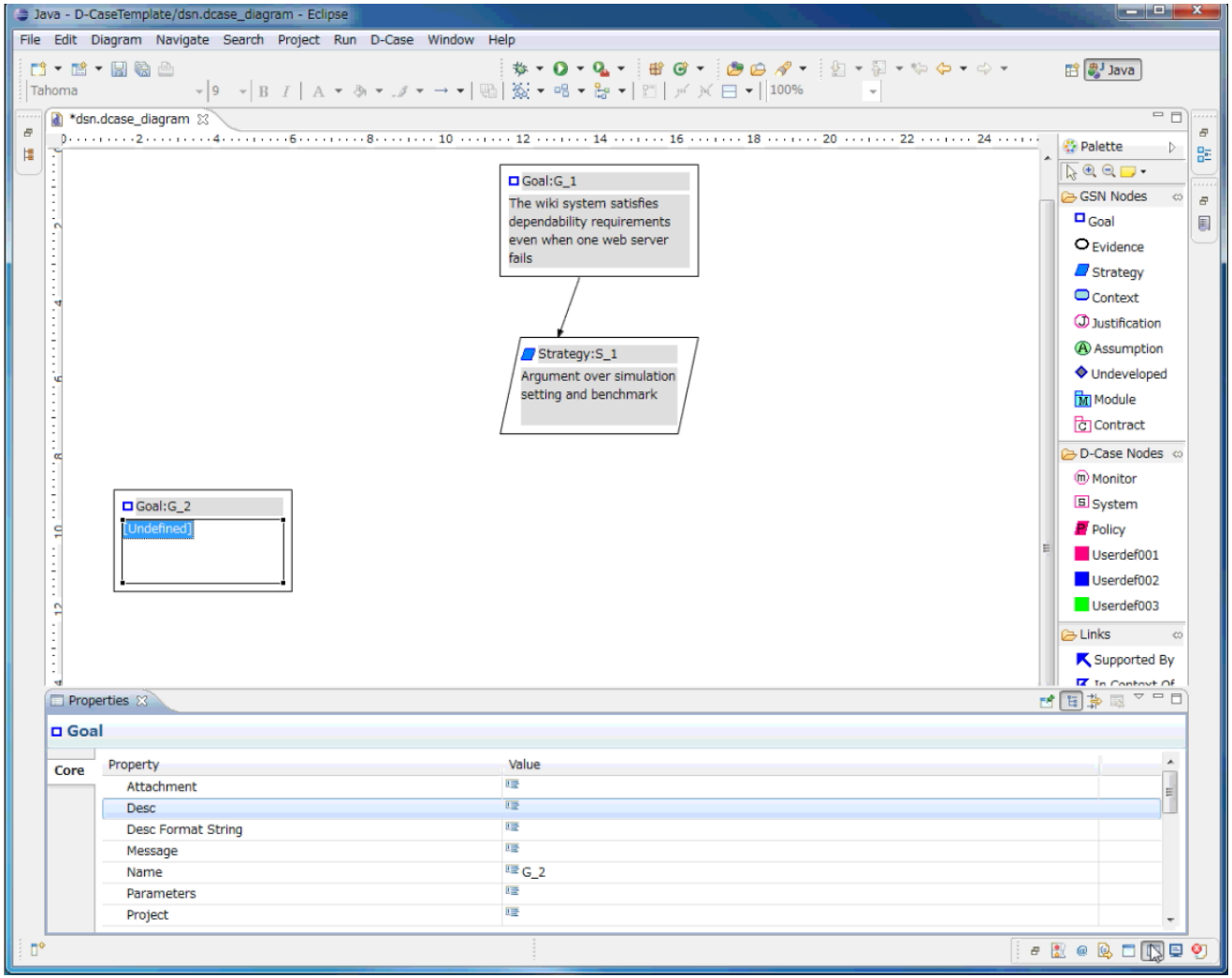
Demo: Creating a D-Case Diagram



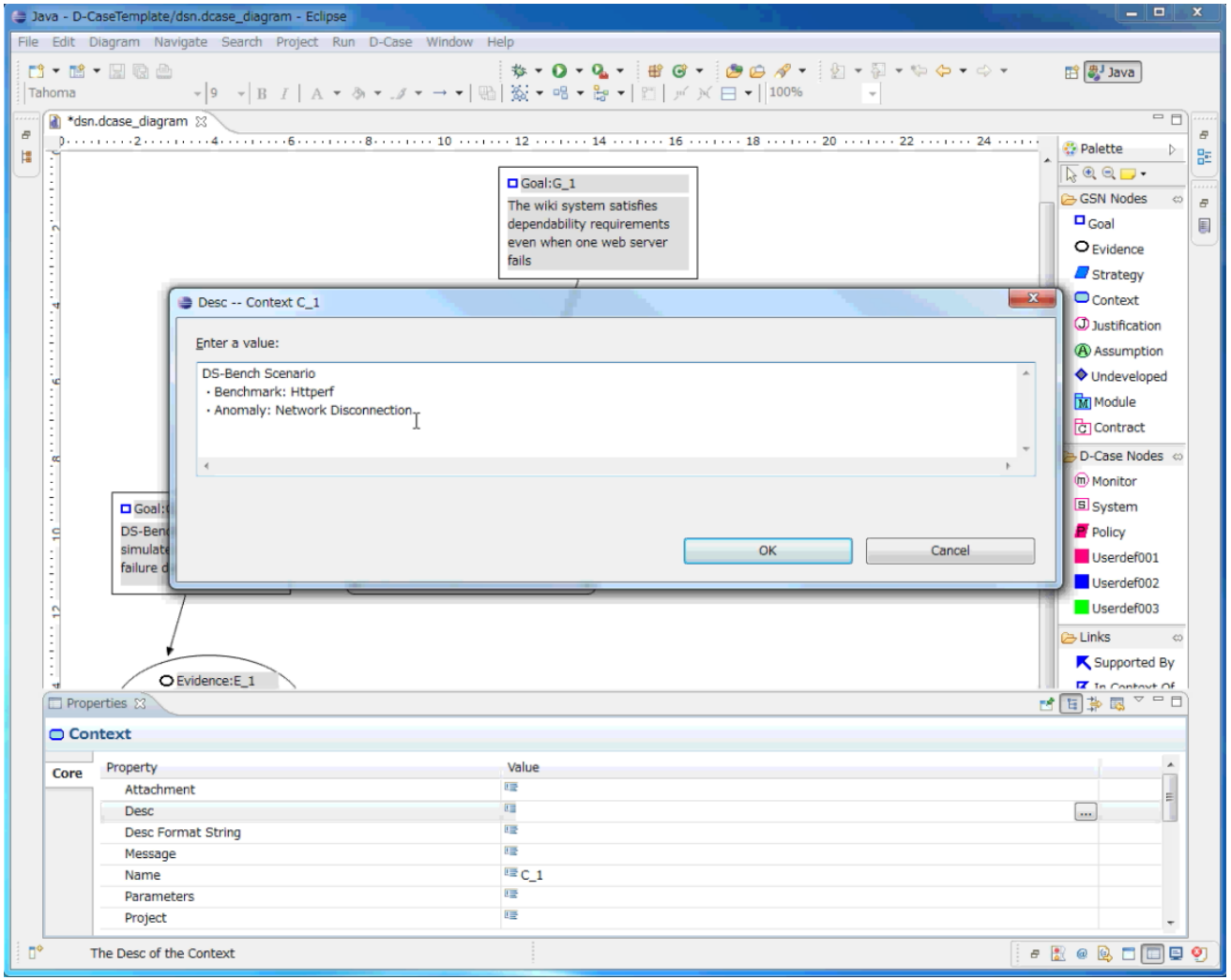
Demo: Creating a D-Case Diagram



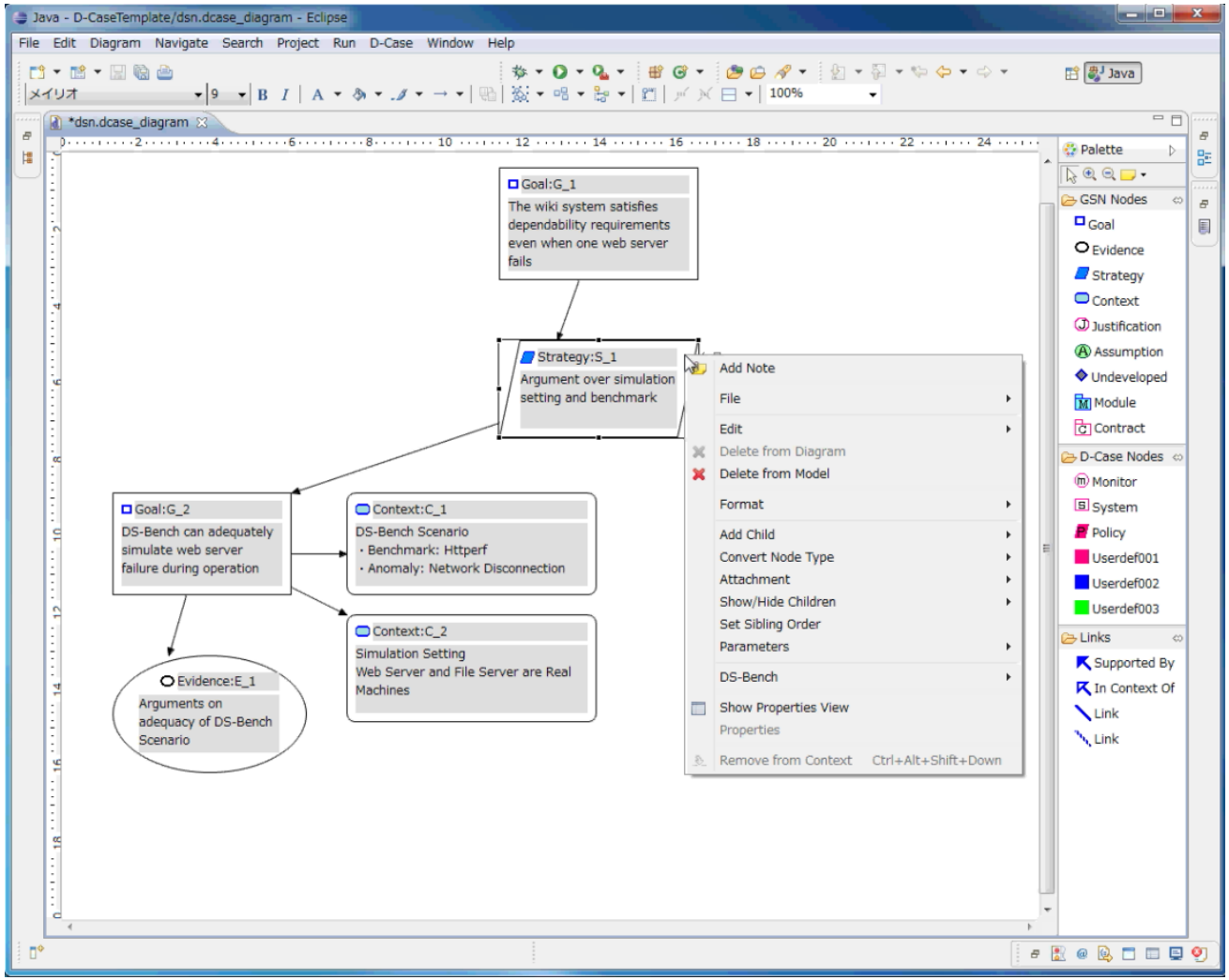
Demo: Creating a D-Case Diagram



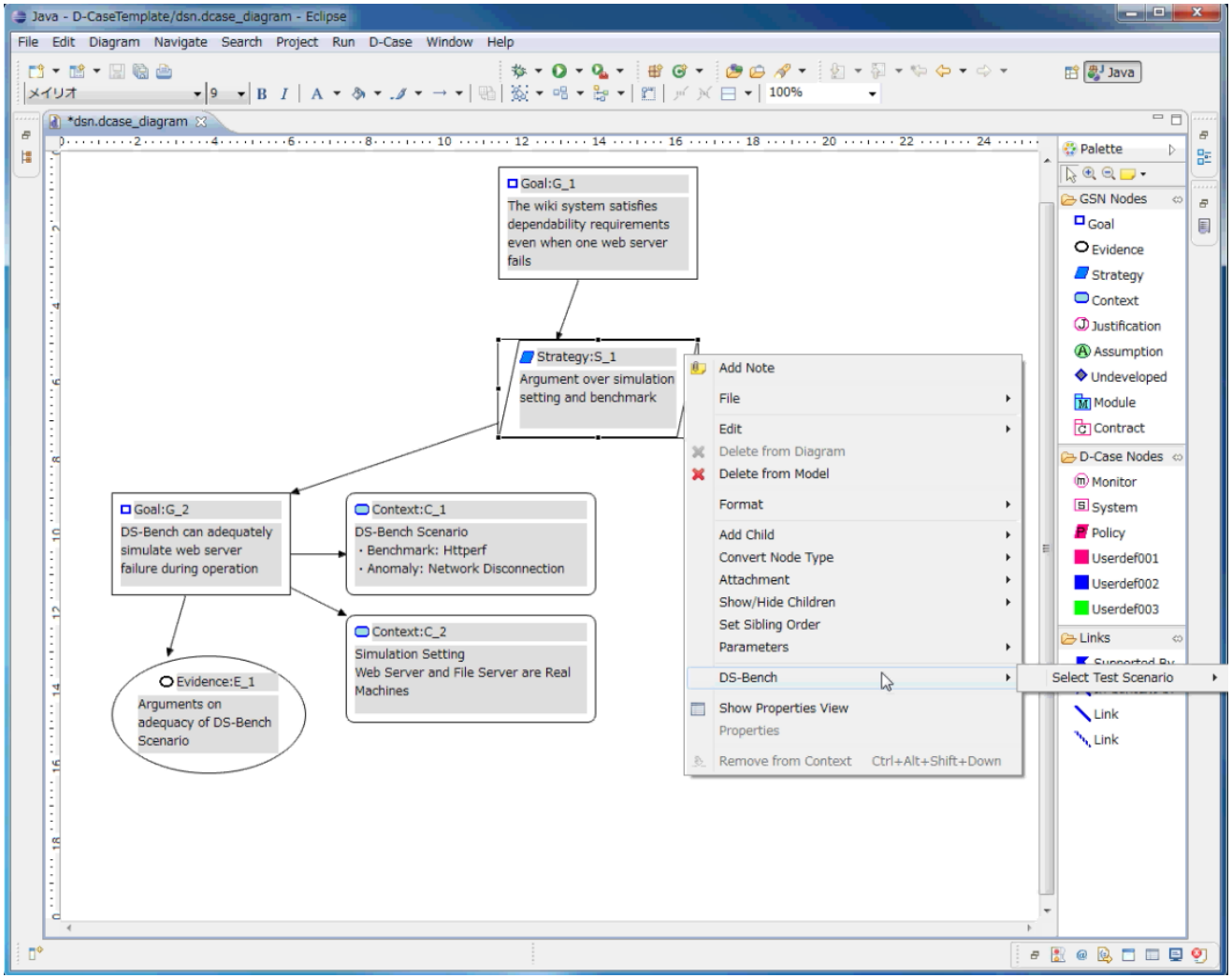
Demo: Creating a D-Case Diagram



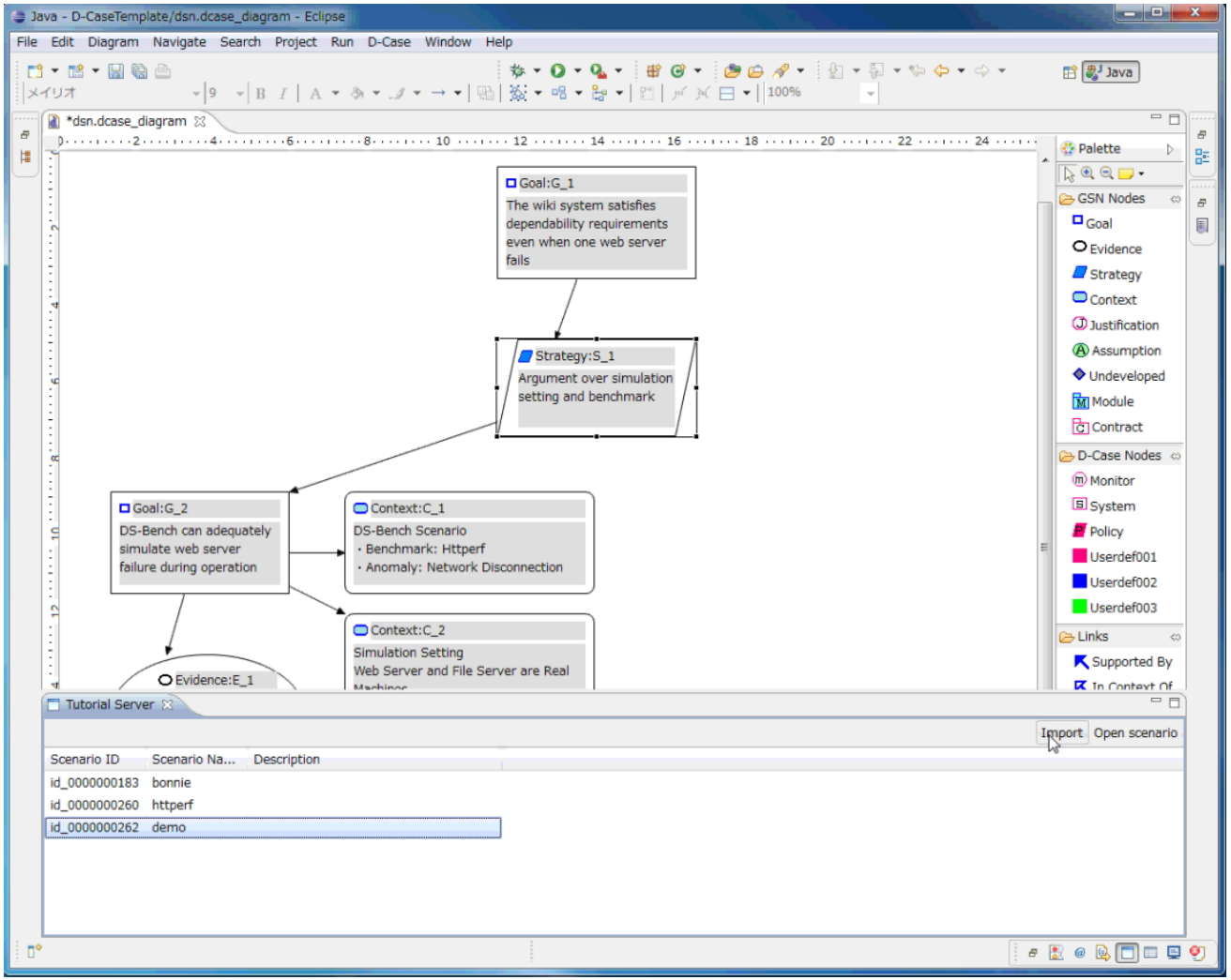
Demo: Selecting a Benchmark Scenario



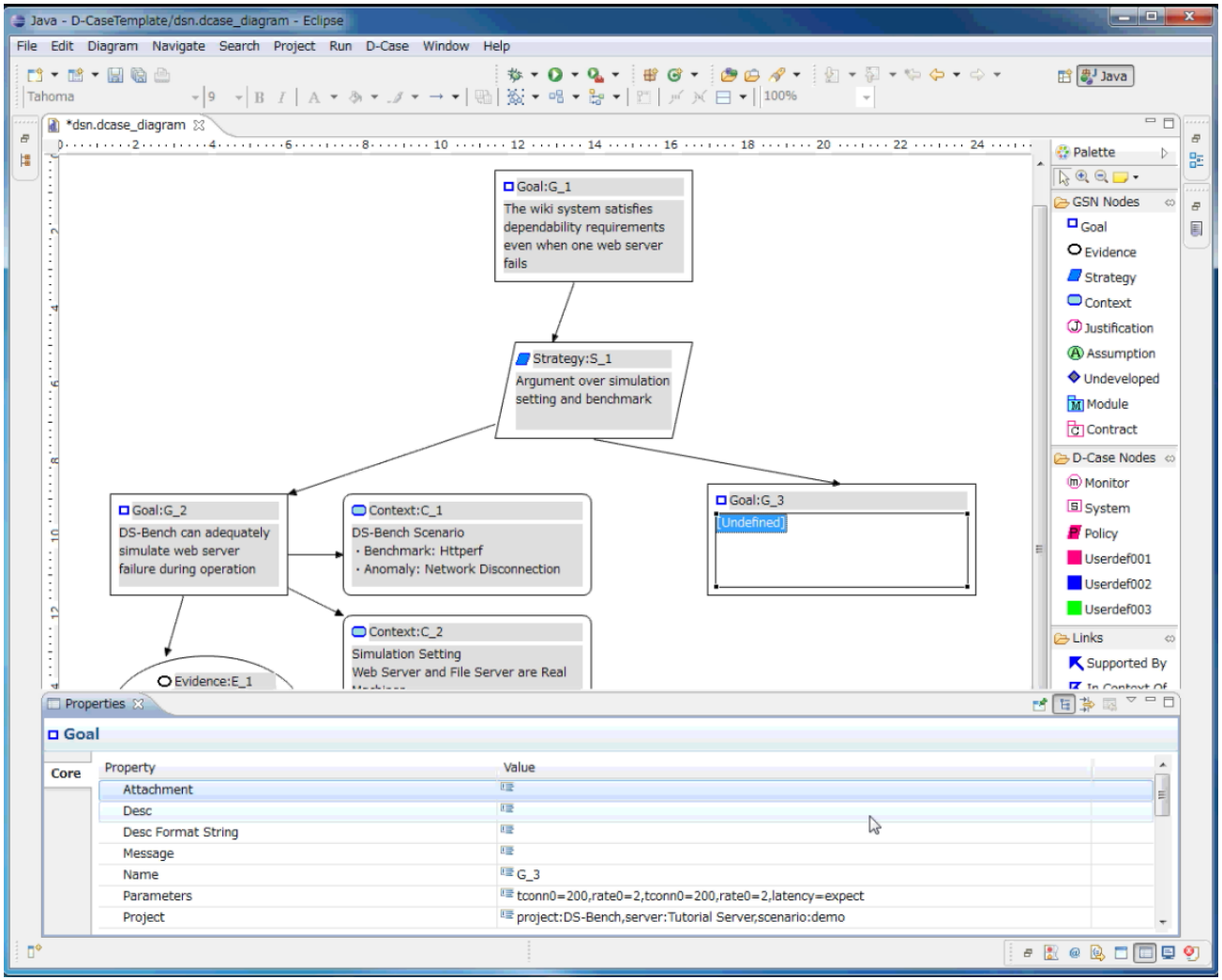
Demo: Selecting a Benchmark Scenario



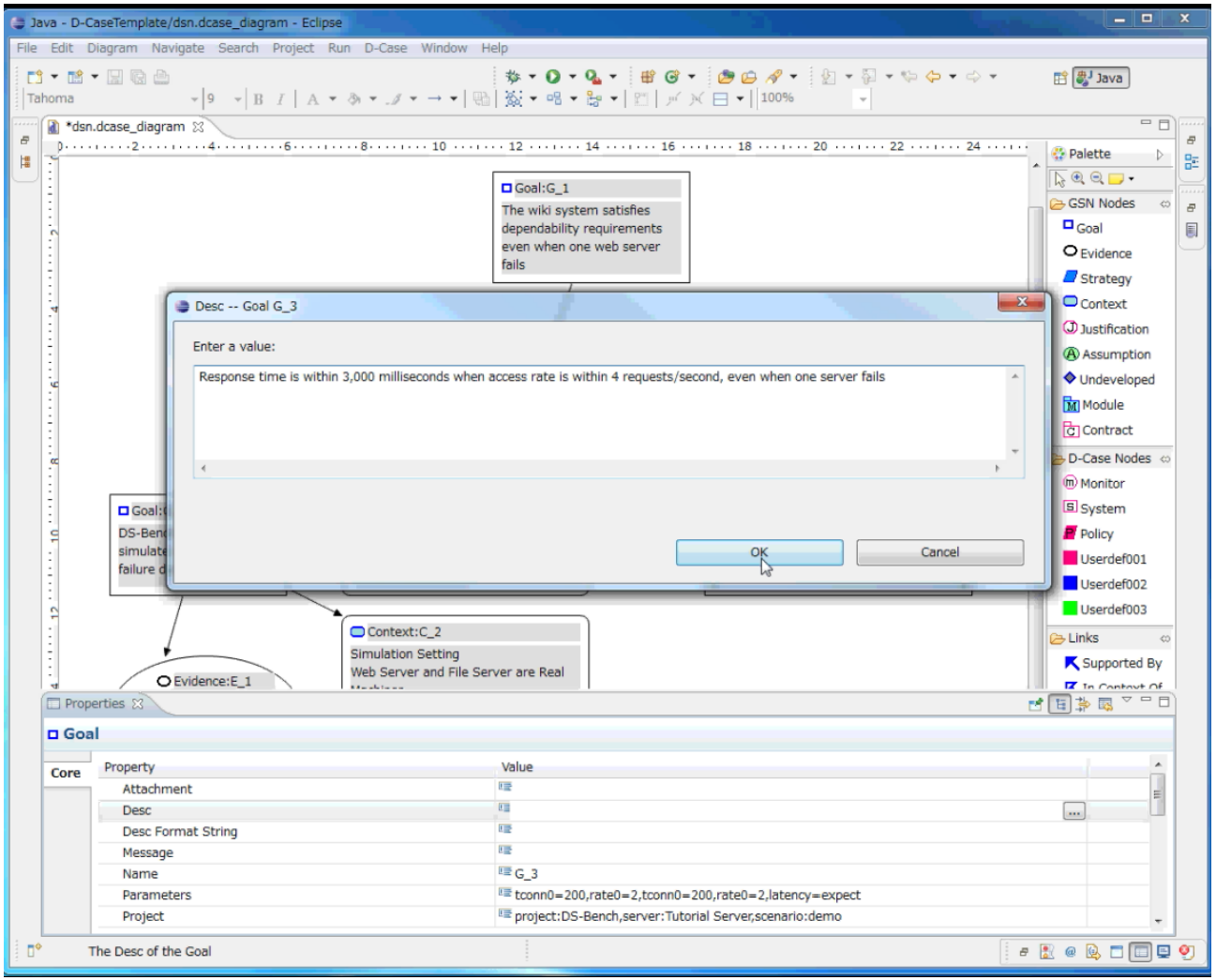
Demo: Selecting a Benchmark Scenario



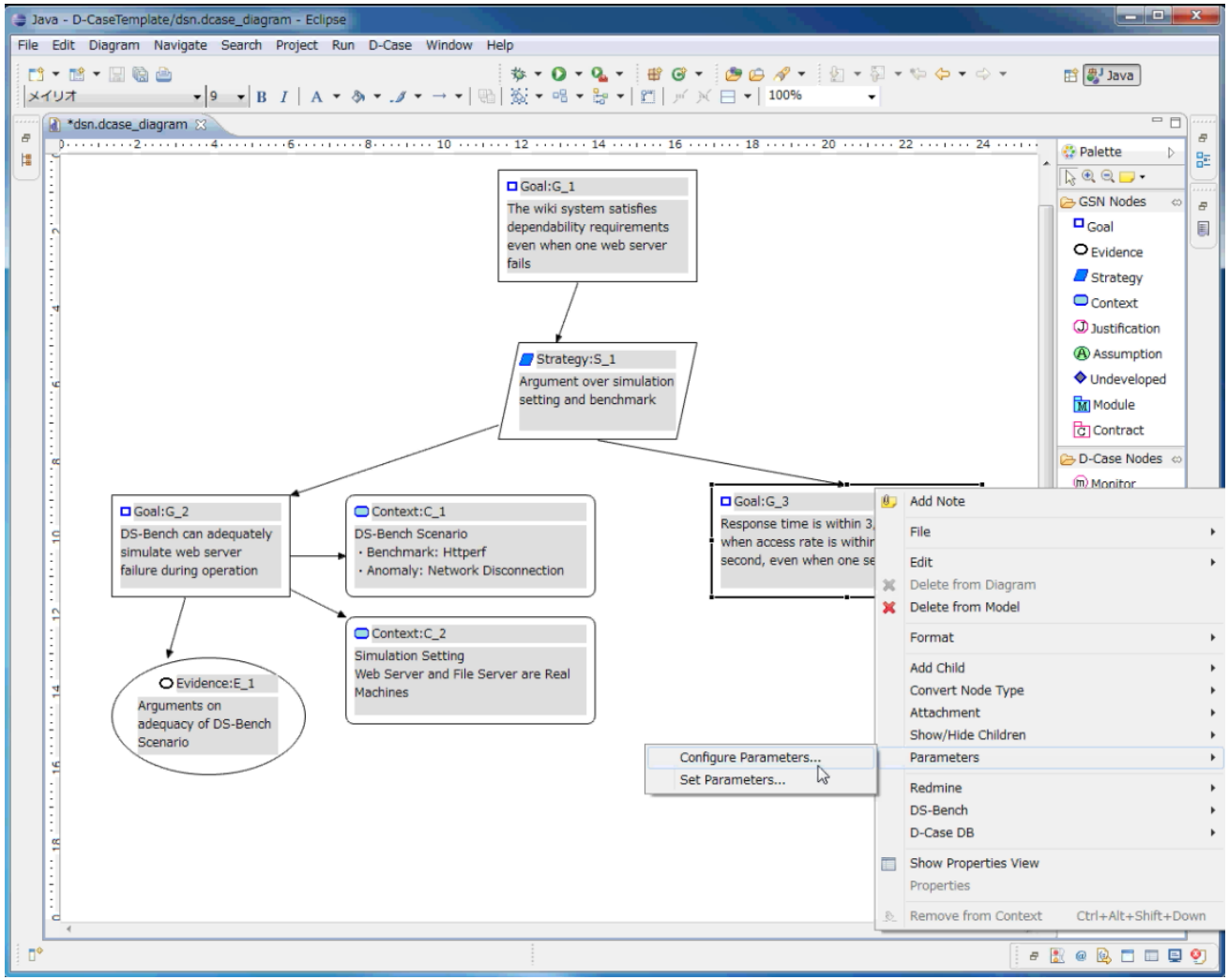
Demo: Selecting a Benchmark Scenario



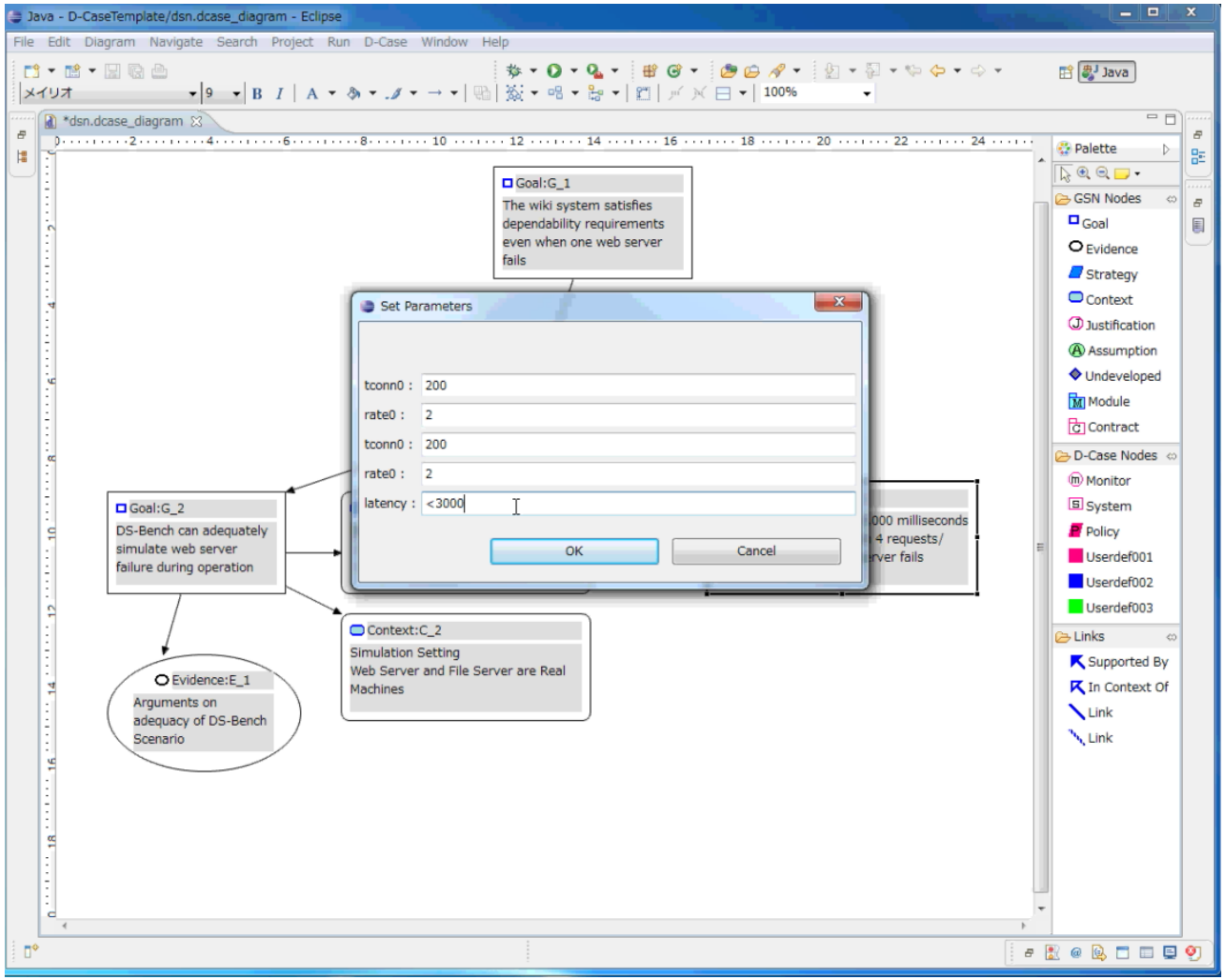
Demo: Selecting a Benchmark Scenario



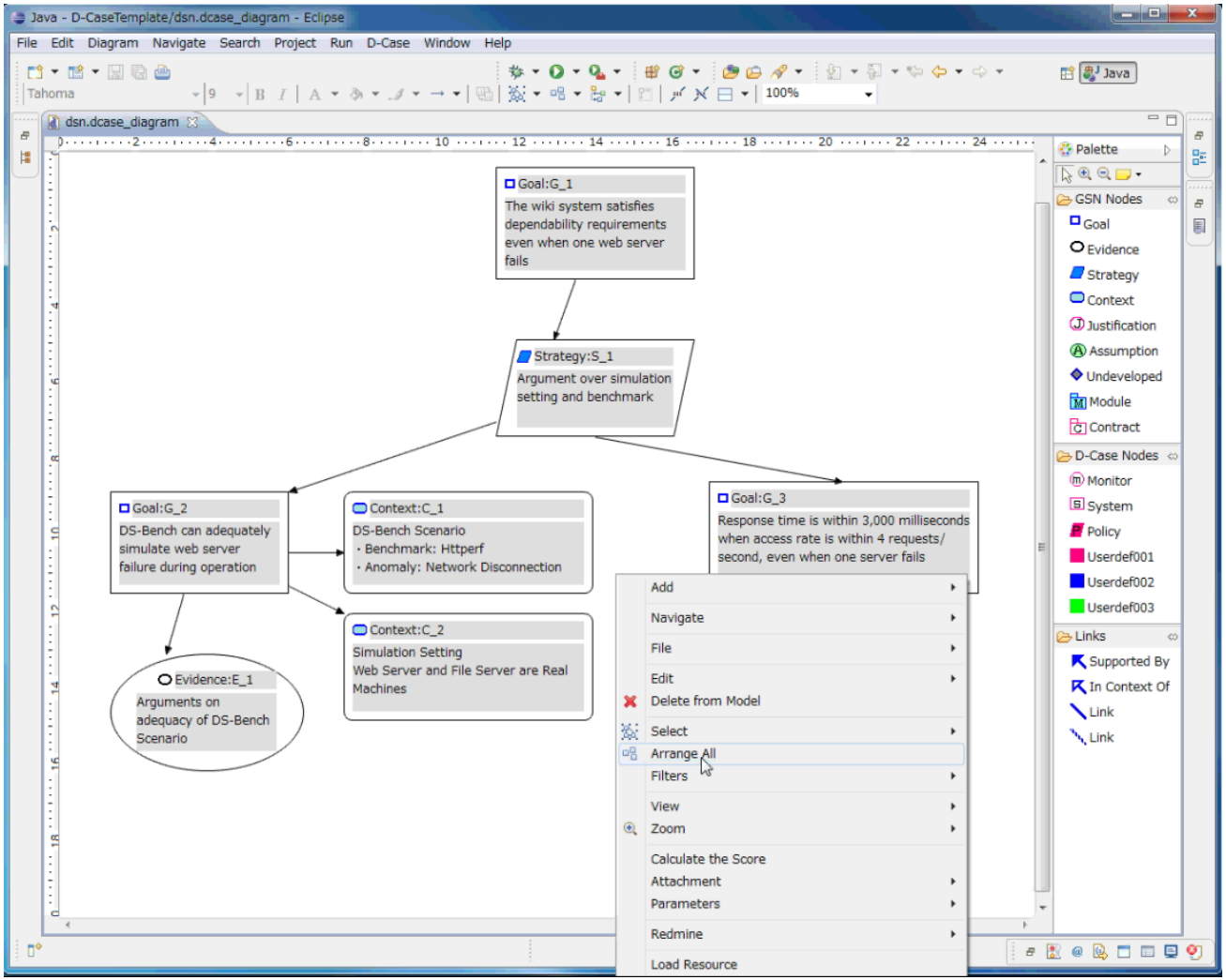
Demo: Setting Benchmark Parameters



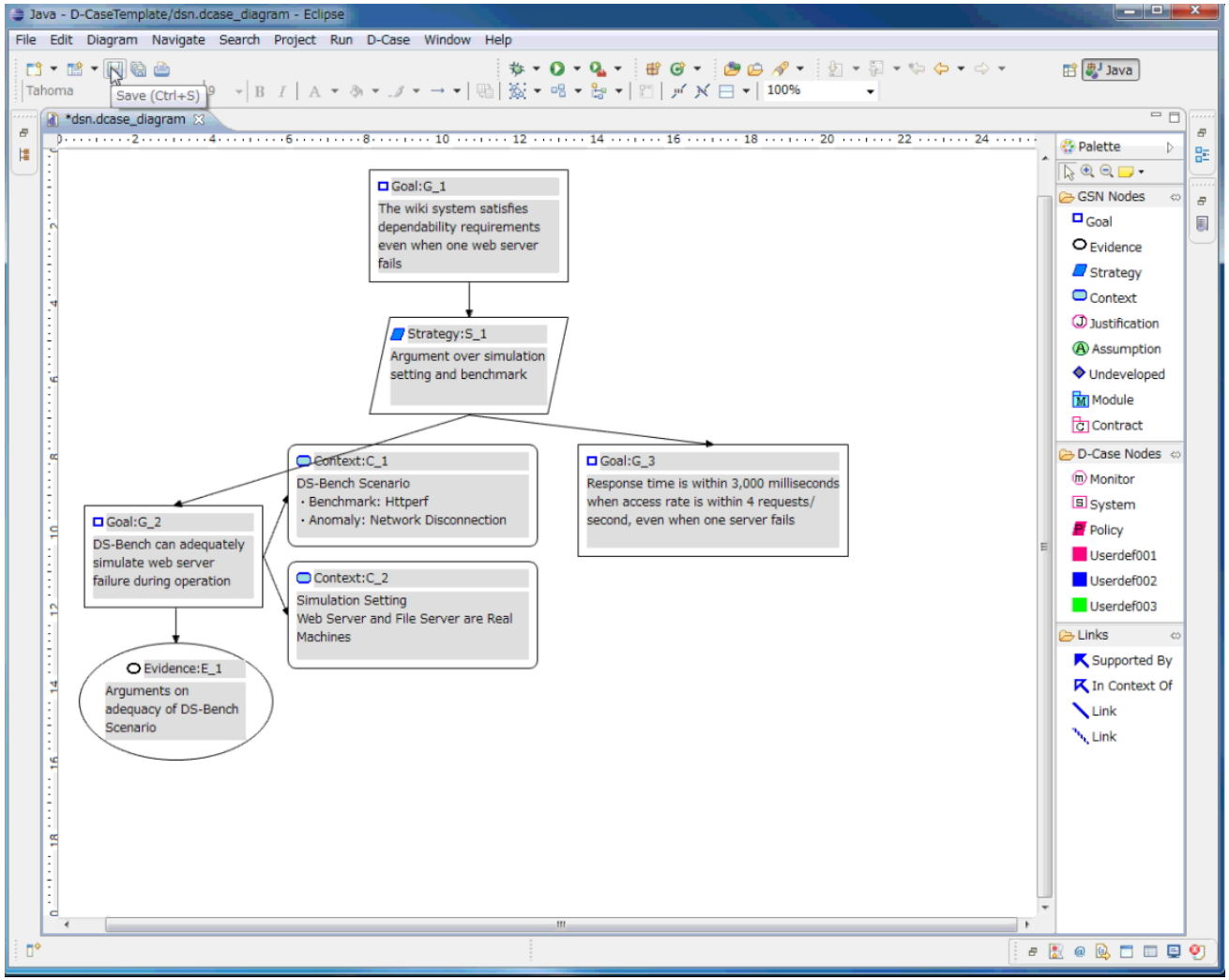
Demo: Setting Benchmark Parameters



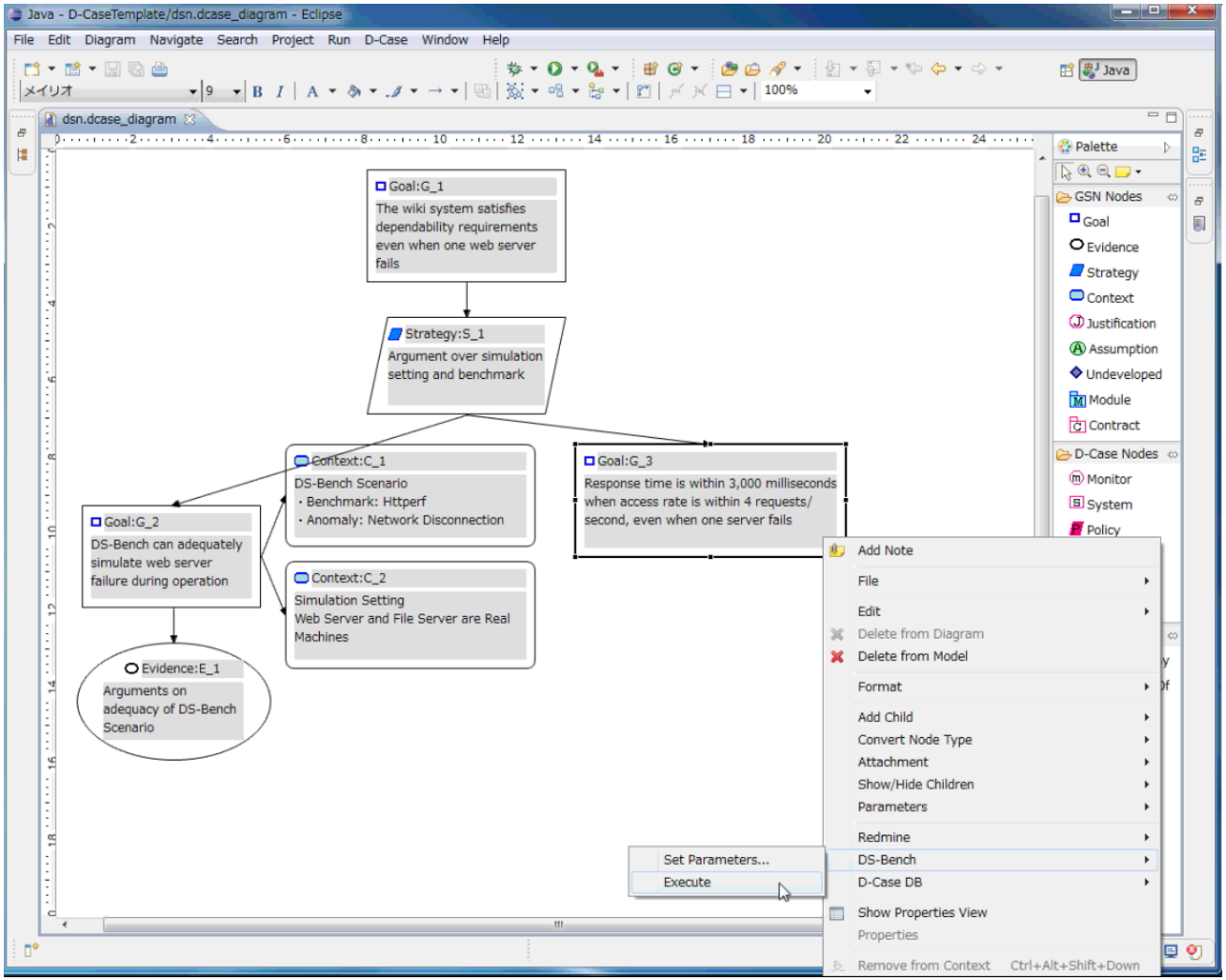
Demo: Setting Benchmark Parameters



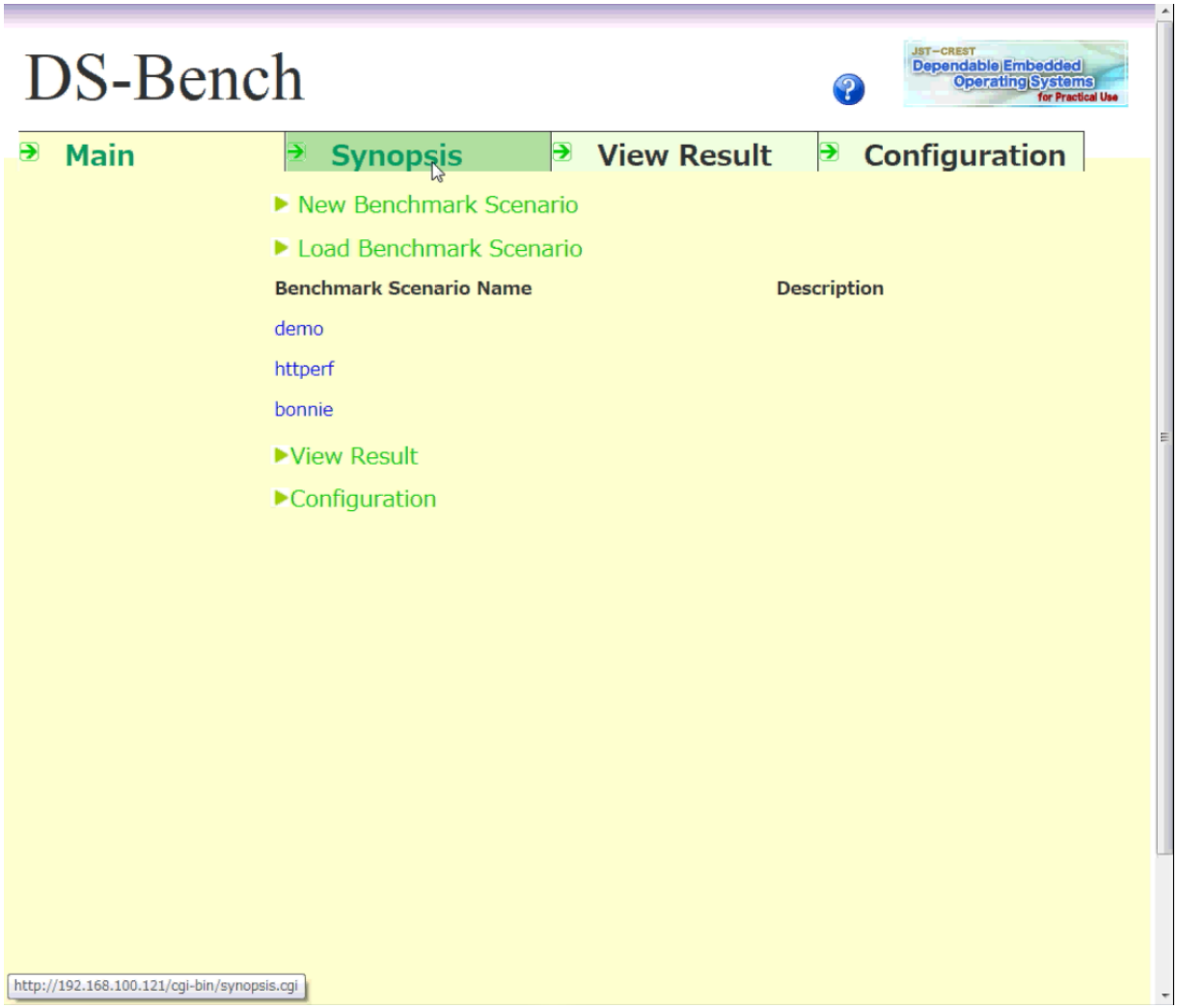
Demo: Benchmark Execution



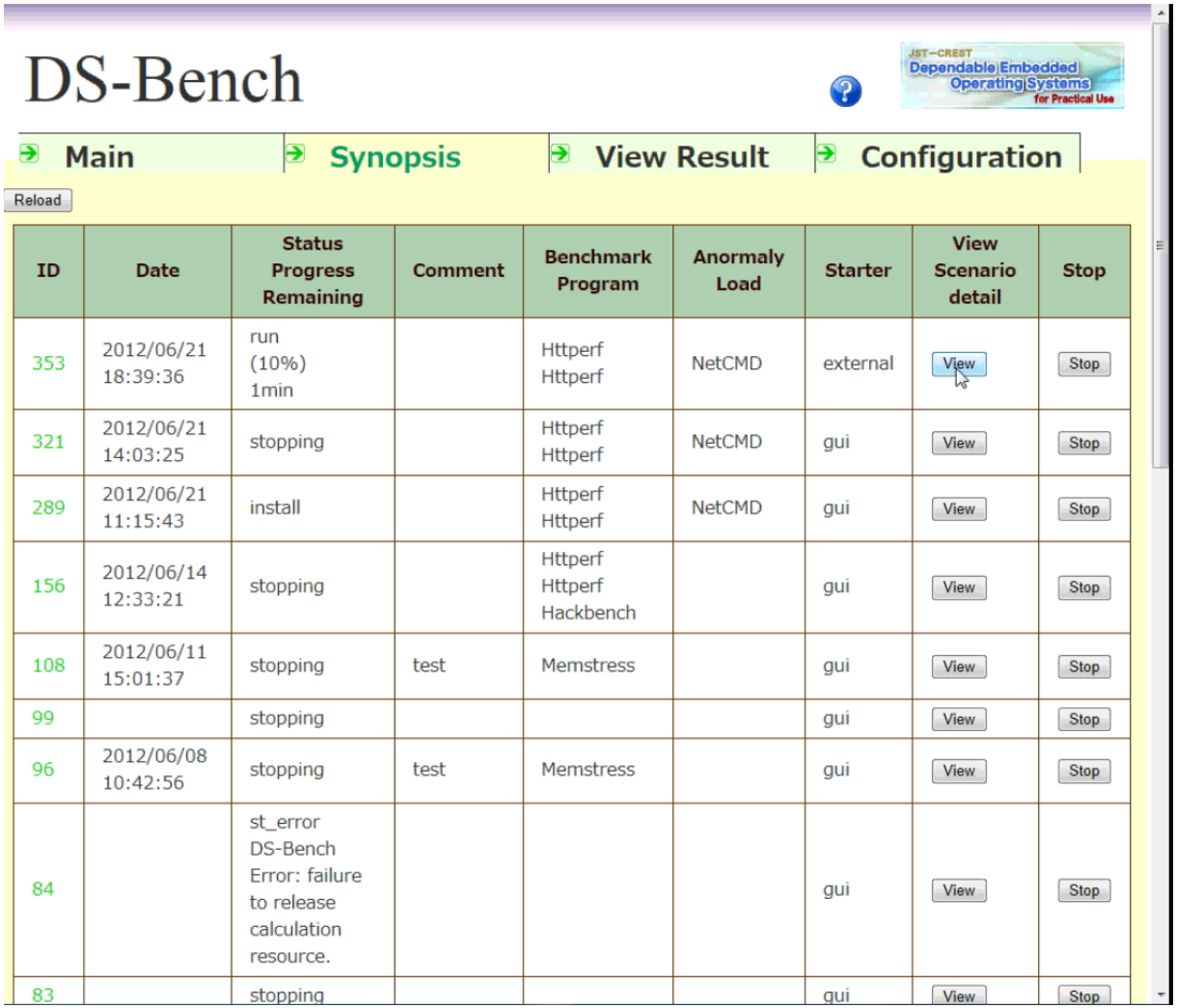
Demo: Benchmark Execution



Demo: Benchmark Execution

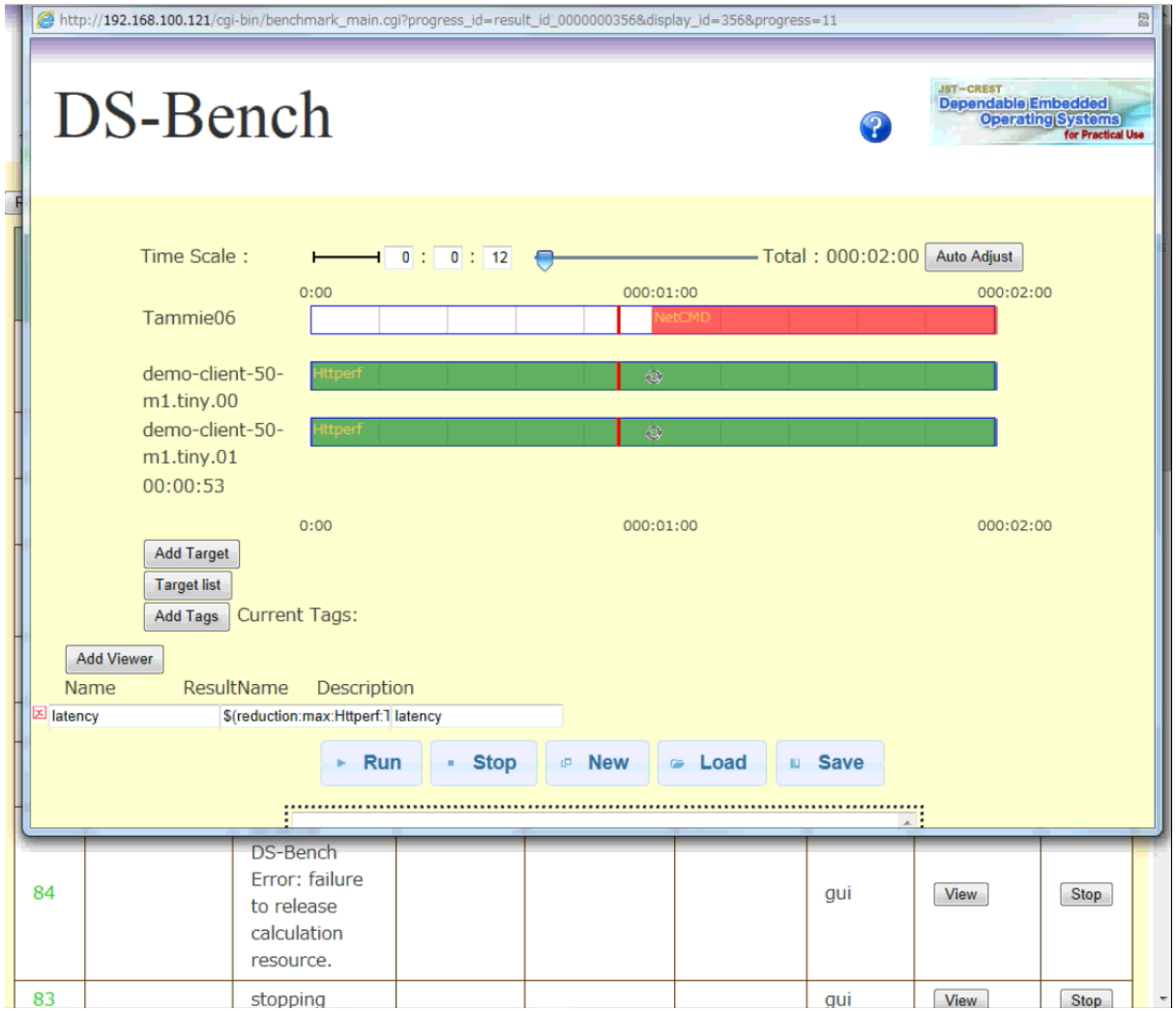


Demo: Benchmark Execution

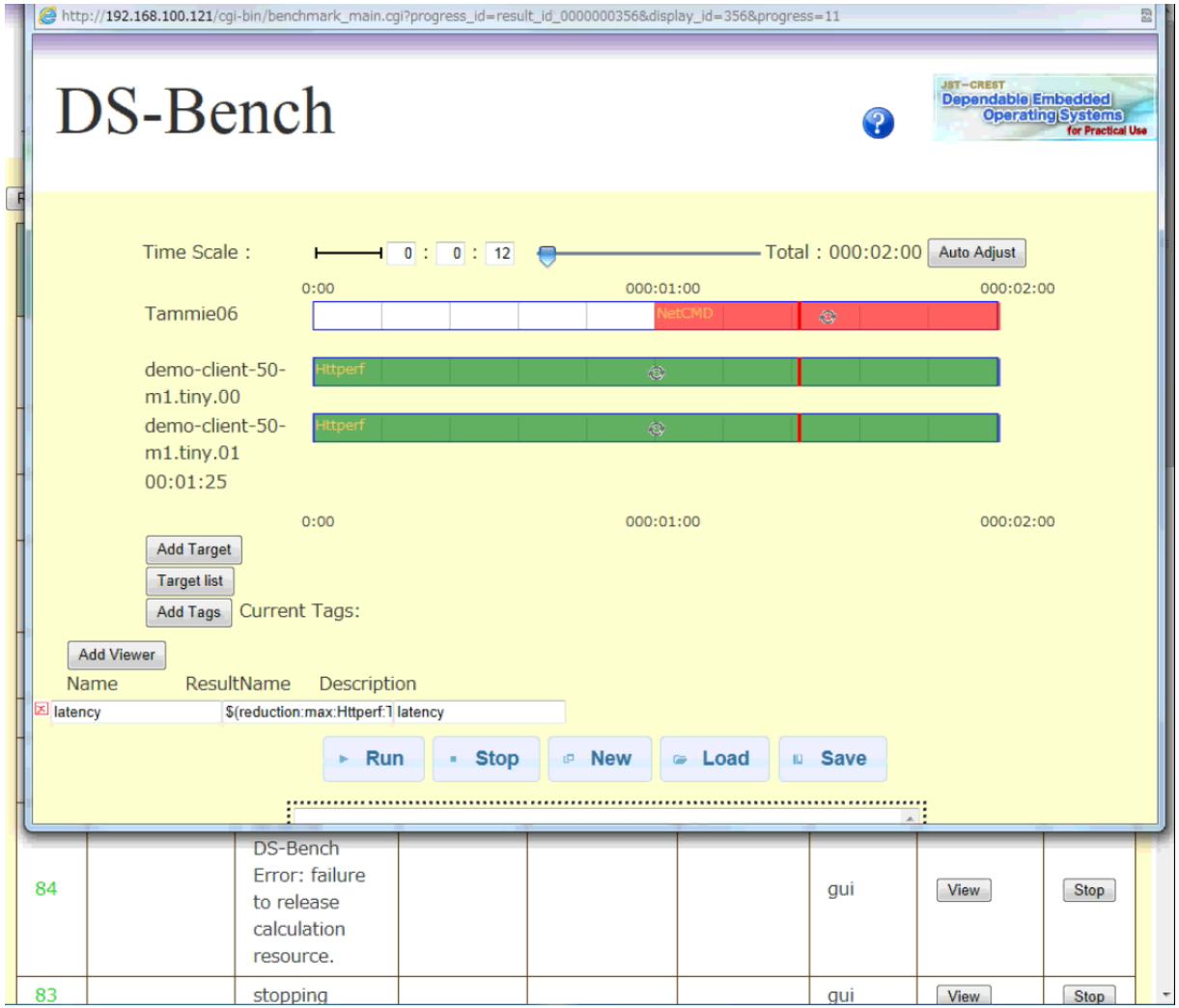


ID	Date	Status Progress Remaining	Comment	Benchmark Program	Anormaly Load	Starter	View Scenario detail	Stop
353	2012/06/21 18:39:36	run (10%) 1min		Httperf Httperf	NetCMD	external	View	Stop
321	2012/06/21 14:03:25	stopping		Httperf Httperf	NetCMD	gui	View	Stop
289	2012/06/21 11:15:43	install		Httperf Httperf	NetCMD	gui	View	Stop
156	2012/06/14 12:33:21	stopping		Httperf Httperf Hackbench		gui	View	Stop
108	2012/06/11 15:01:37	stopping	test	Memstress		gui	View	Stop
99		stopping				gui	View	Stop
96	2012/06/08 10:42:56	stopping	test	Memstress		gui	View	Stop
84		st_error DS-Bench Error: failure to release calculation resource.				gui	View	Stop
83		stopping				gui	View	Stop

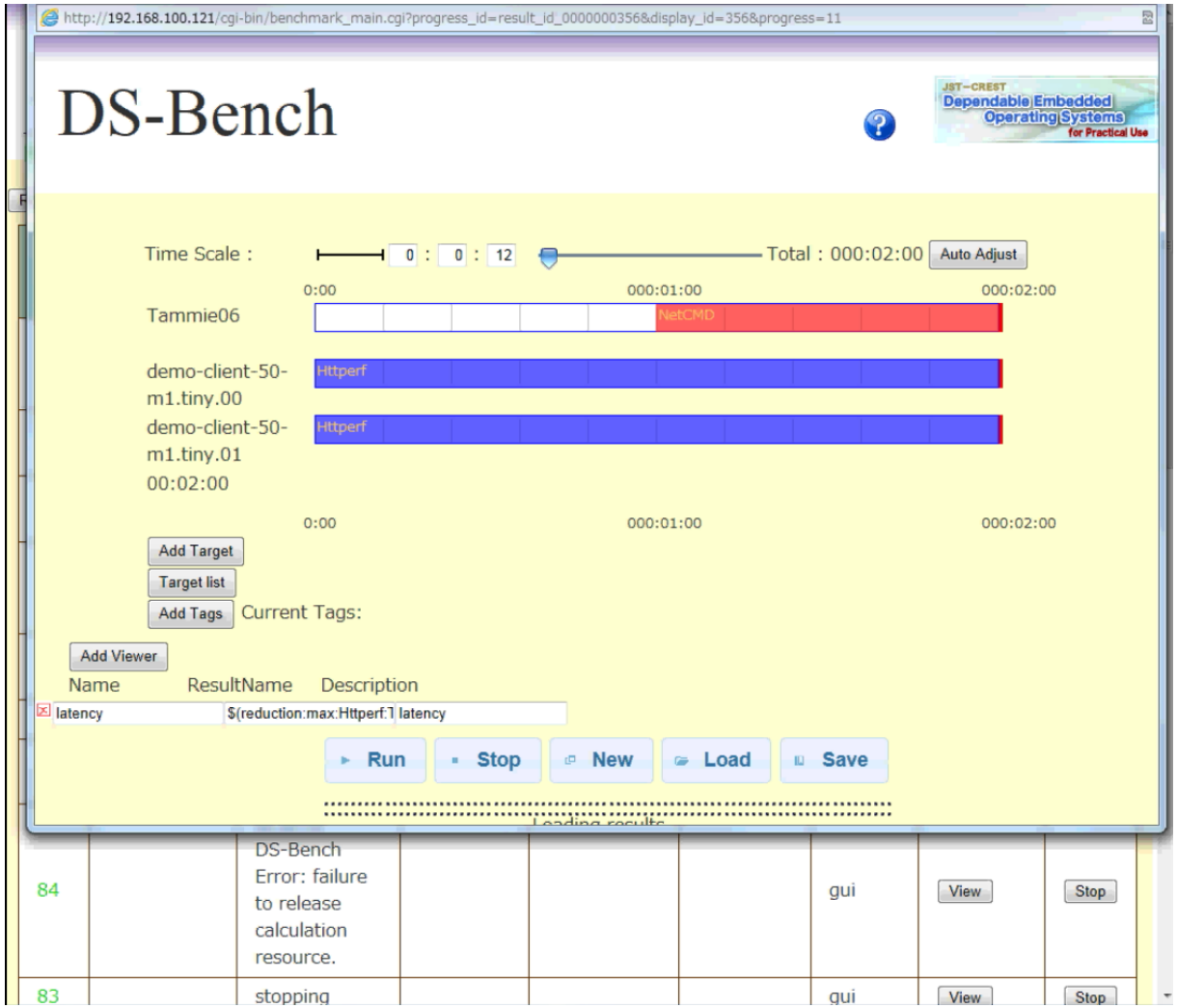
Demo: Benchmark Execution



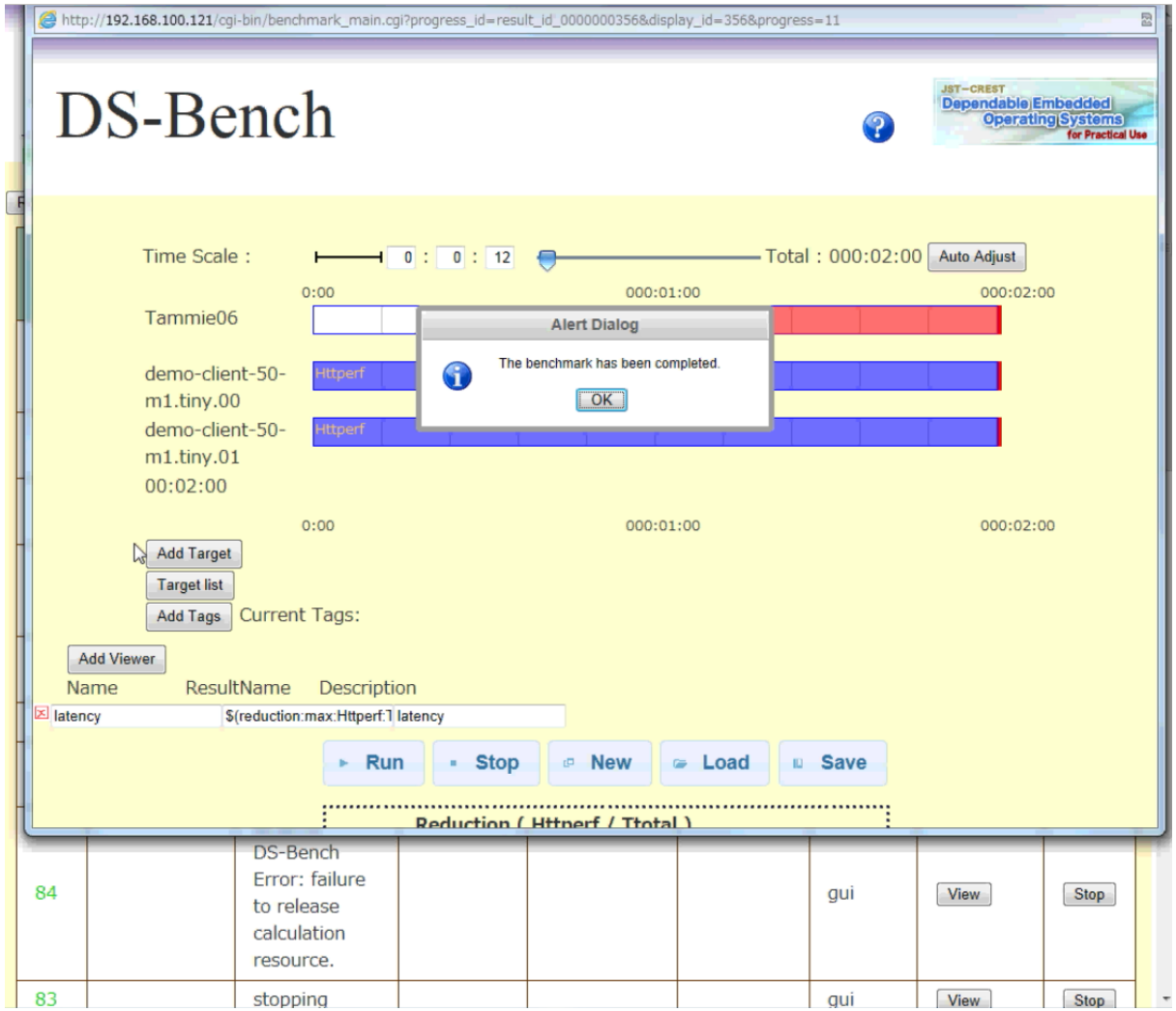
Demo: Benchmark Execution



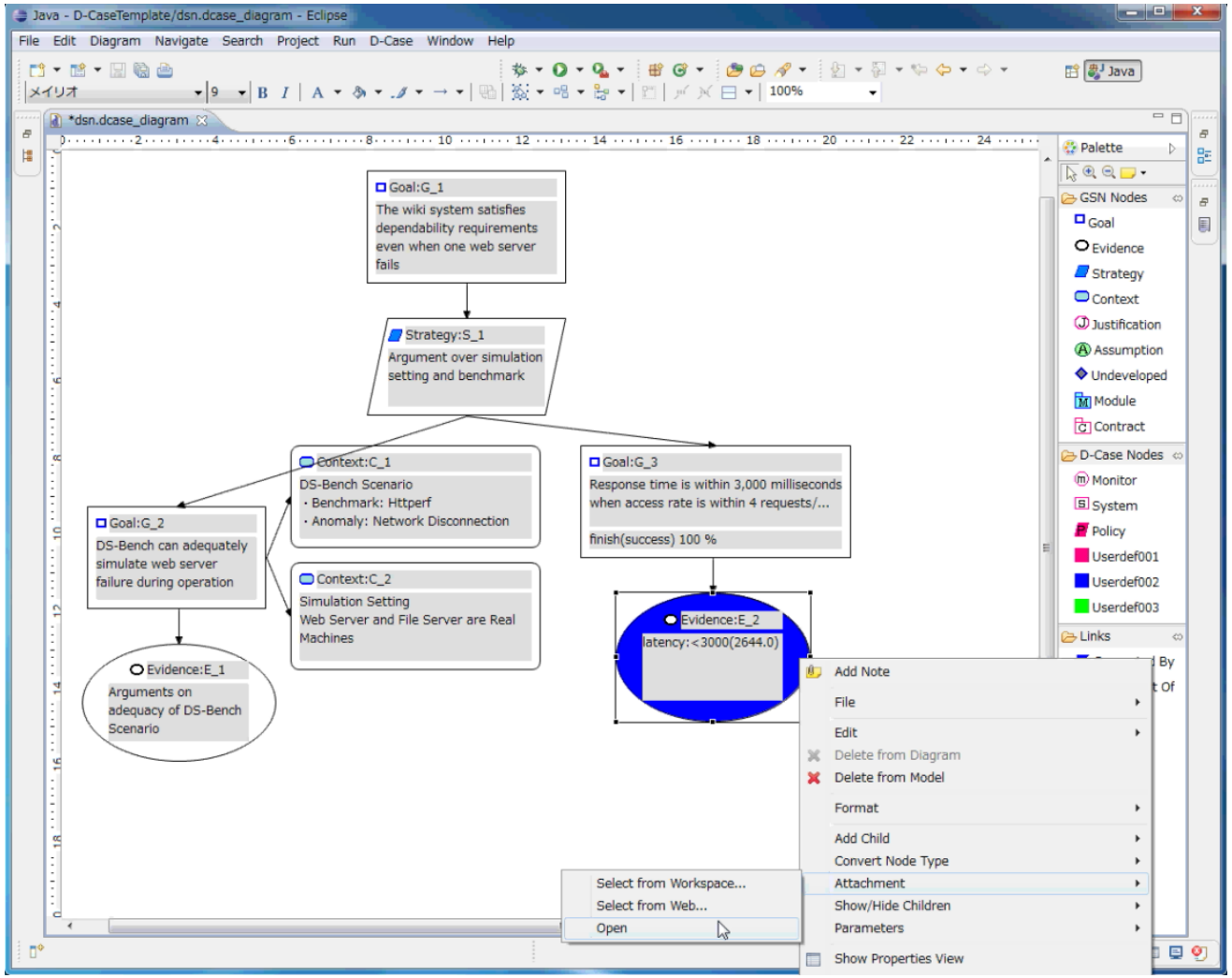
Demo: Benchmark Execution



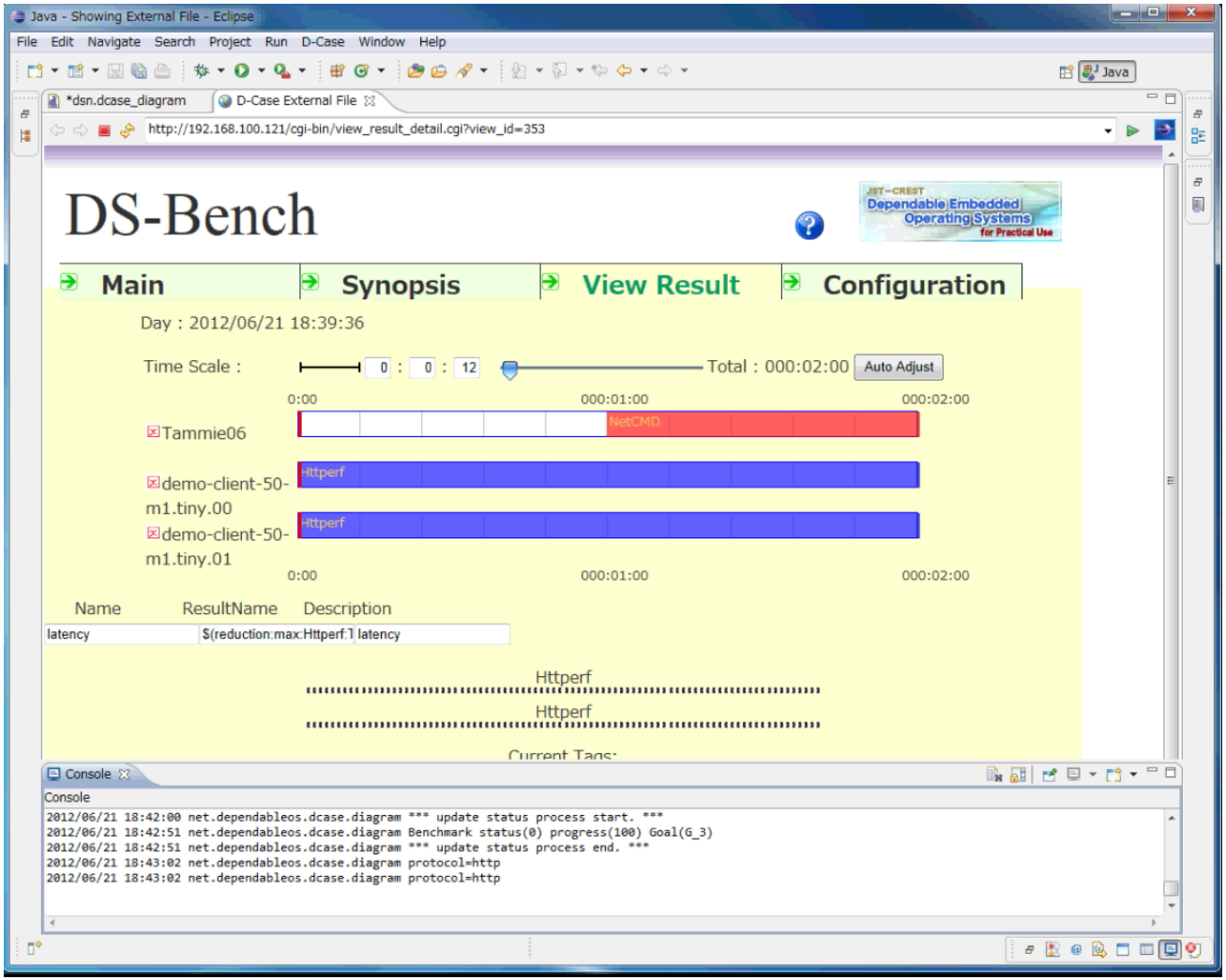
Demo: Benchmark Execution



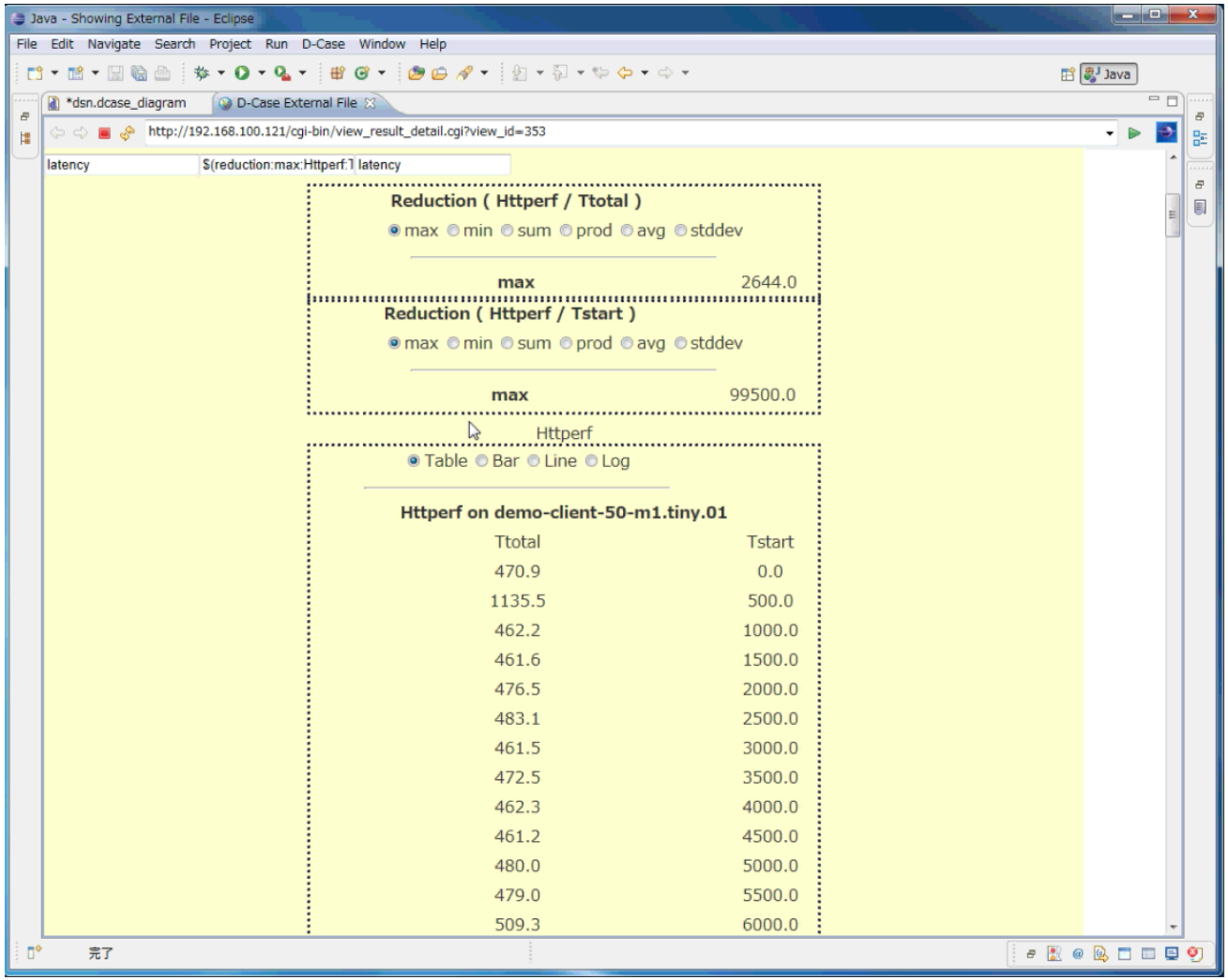
Demo: Benchmark Result



Demo: Benchmark Result



Demo: Benchmark Result



Related Work

- ASCE Tool [Adelard]
  - An well-used assurance case editor. Mainly for generating certification documents
- D-Bench [Kanoun 2004, Durães 2004]
  - A project aimed at establishing dependability benchmarking methods for several domains
  - Proposed dependability metrics and measurement tools for several target domains, however did not provide a total software framework to integrate multiple benchmarks
- Fault injection tools
  - As far as we know, no effort have been made to combine assurance cases and benchmark results dynamically

Summary

- An automated tool for dependability benchmarking, with dependability assurance, is needed
- DS-Bench Toolset
  - D-Case Editor, an assurance case editor
  - DS-Bench, a framework for benchmark test
  - D-Cloud, a system for managing hardware resources for benchmark test
- Tight collaboration between D-Case Editor and DS-Bench
  - Dependability requirements are described in D-Case
  - DS-Bench conducts benchmark tests to obtain a quantitative evidence for D-Case

Questions?

- D-Case Editor is available from
  - http://www.dependable-os.net/tech/D-CaseEditor/
- DS-Bench/D-Cloud will be available from
  - http://www.dependable-os.net/tech/DSBenchDCloud/

- We thank Hajime Ueno for designing D-Case Editor with us. We also thank Shingo Takeda and Hideaki Koizumi for helping us with preparing the demonstration.