



# A TTCN-3 based Benchmark for BPEL Engines

---

George Din, Klaus-Peter Eckert, Ina Schieferdecker

Fraunhofer FOKUS, Berlin, Germany



- Context
- Benchmarking Methodology
- TTCN-3 Benchmarking Framework
- A Benchmark Example
- Conclusions & Outlook



# Benchmark Understanding

- A benchmark evaluates the performance of a system by monitoring the system while it is being exposed to a particular workload

*A. J. Smith. Workloads (creation and use). Commun. ACM, 50:45–50, 2007.*

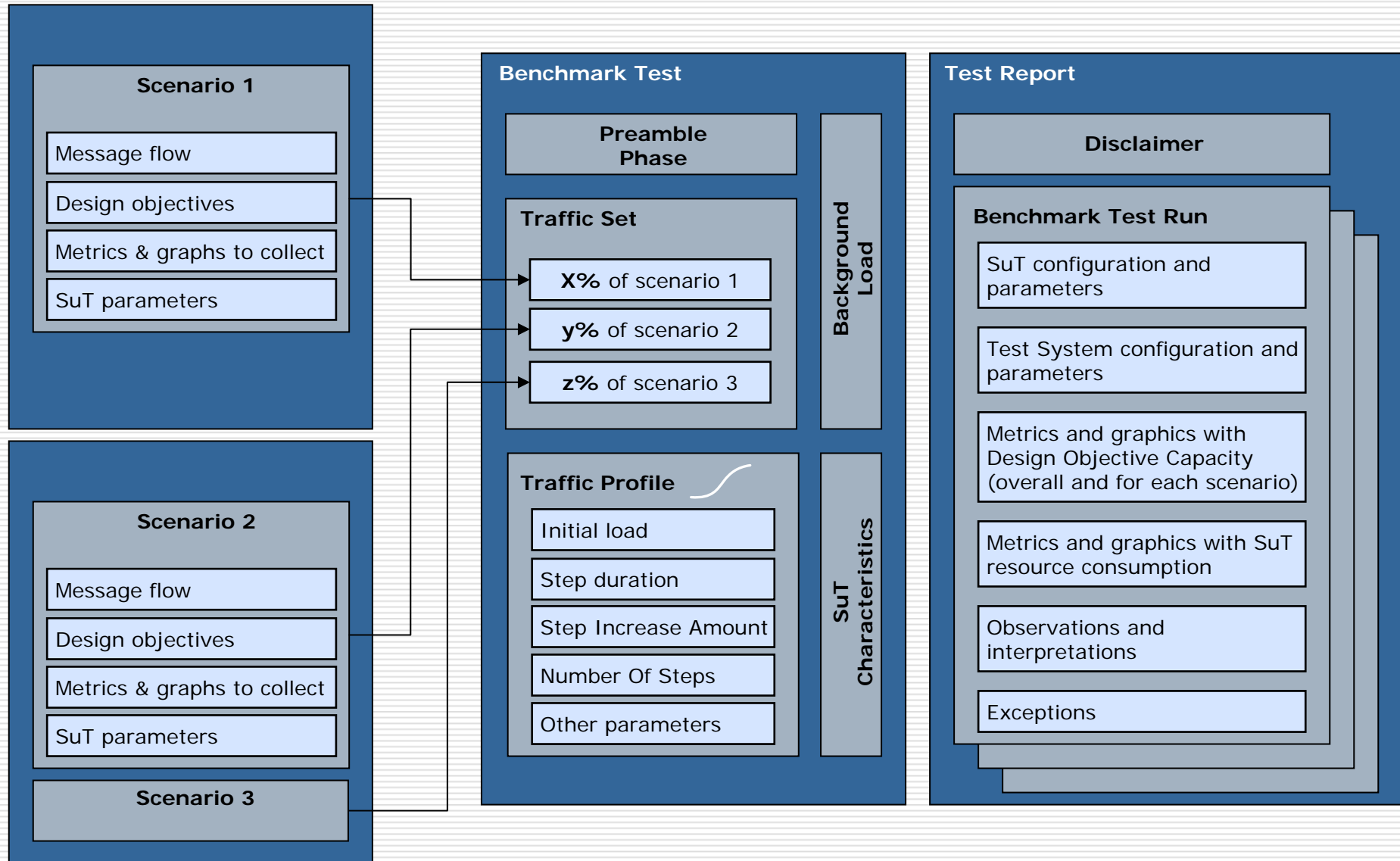
- The activity to select and define a workload is called workload characterization

*A. Avritzer, J. Kondek, D. Liu, and E. J. Weyuker. Software performance testing based on workload characterization. pp. 17–24, Rome, Italy, 2002. ACM. ISBN 1-58113-563-7.*

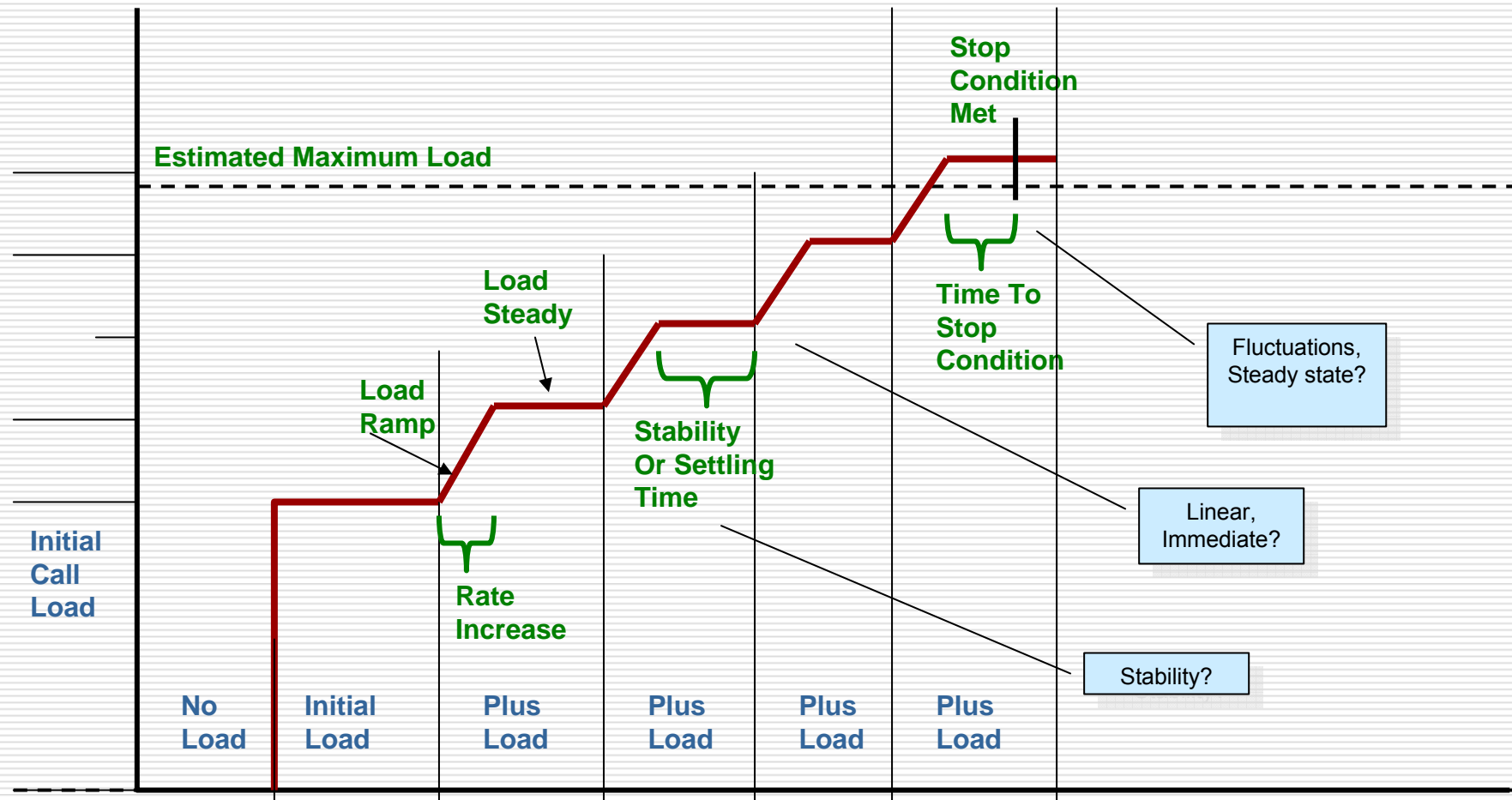


- **Goal - Performance Benchmarking for BPEL Engines**
  - Performance and scalability testing of BPEL Engines with simulated real-world traffic
  - Measurement and analysis of important QoS parameters
  
- **Why**
  - Creation of design objectives means to enable comparison of BPEL implementations of different providers by performance (and price)
  - Unify the test procedures, the test parameters and the Benchmark test report aiming at standardization
  
- **How**
  - Define standard scenarios and traffic models for the work load
  - Define the metrics to be measured
  - Use standardized and abstract testing language TTCN-3 on the test system side

# General Benchmark Model



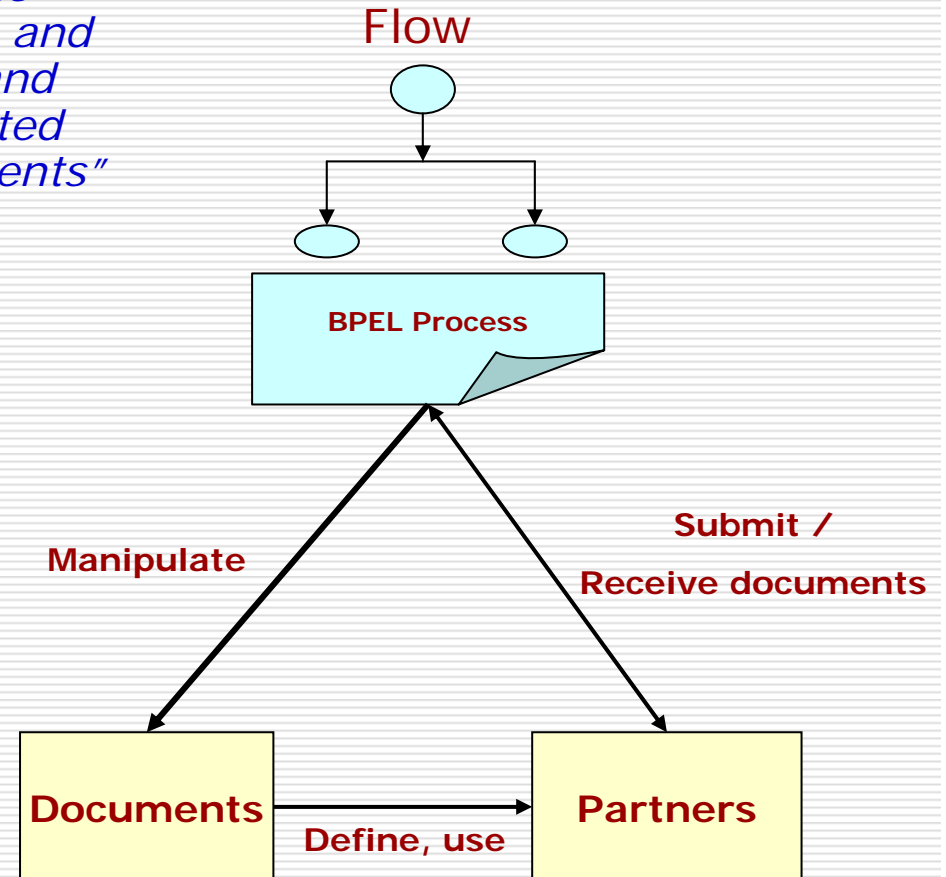
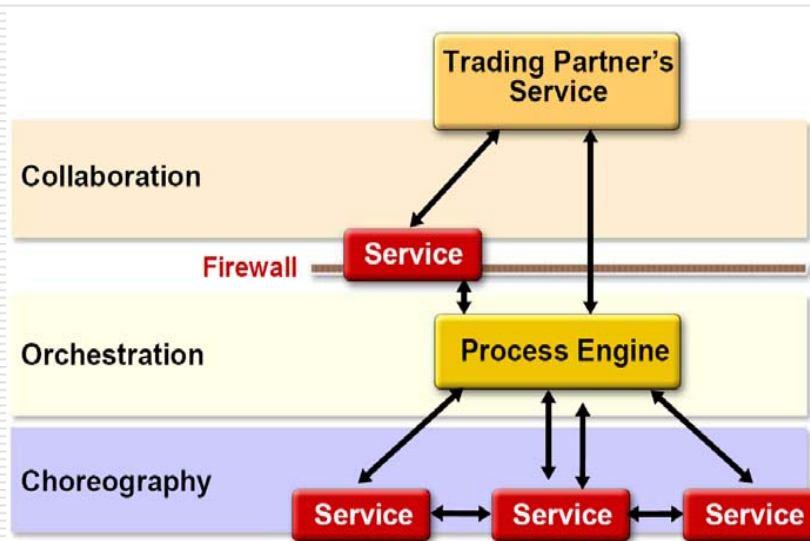
# Benchmark Procedure



- This benchmarking technology has been developed at ETSI/TIPSAN for IMS Testing – in this paper applied for BPEL engines

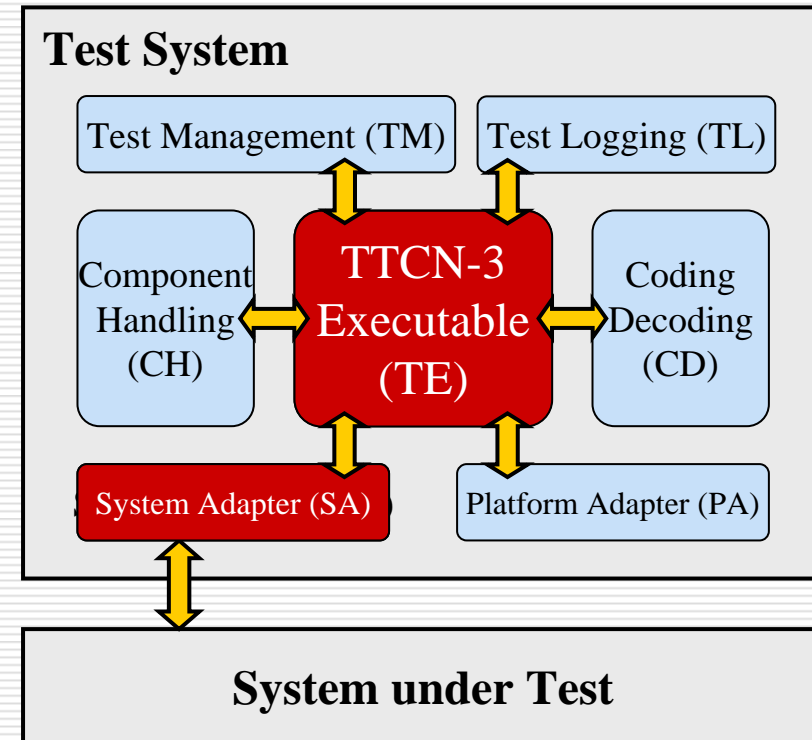
# BPEL Processes

- ❑ **Business Process Execution Language**
- ❑ *"XML-based standard that facilitates designing, defining, implementing, and deploying composite applications and services from a number of distributed and autonomous software components"*
- ❑ Interacts with Web Services
- ❑ WSDL, XML Schema, XPath, WS-Addressing
- ❑ Exposes itself as a web service



# TTCN-3 Test System Architecture

- **Testing and Test Control Notation**
- General architecture used for black-box distributed testing, standardized by ETSI
- TTCN-3 is used for many kinds of testing
  - for all types of reactive system tests over a variety of communication port
  - conformance, interoperability, robustness, regression, system and integration testing
  - its modularity allows the description of complex distributed test behavior
  - a common meta-model with multiple presentation formats (text, graphical, tabular)



# BPEL Benchmarking Scenario

## □ SUT

- ActiveBPEL running on Tomcat
- Synchronous interaction

## □ Benchmark Test

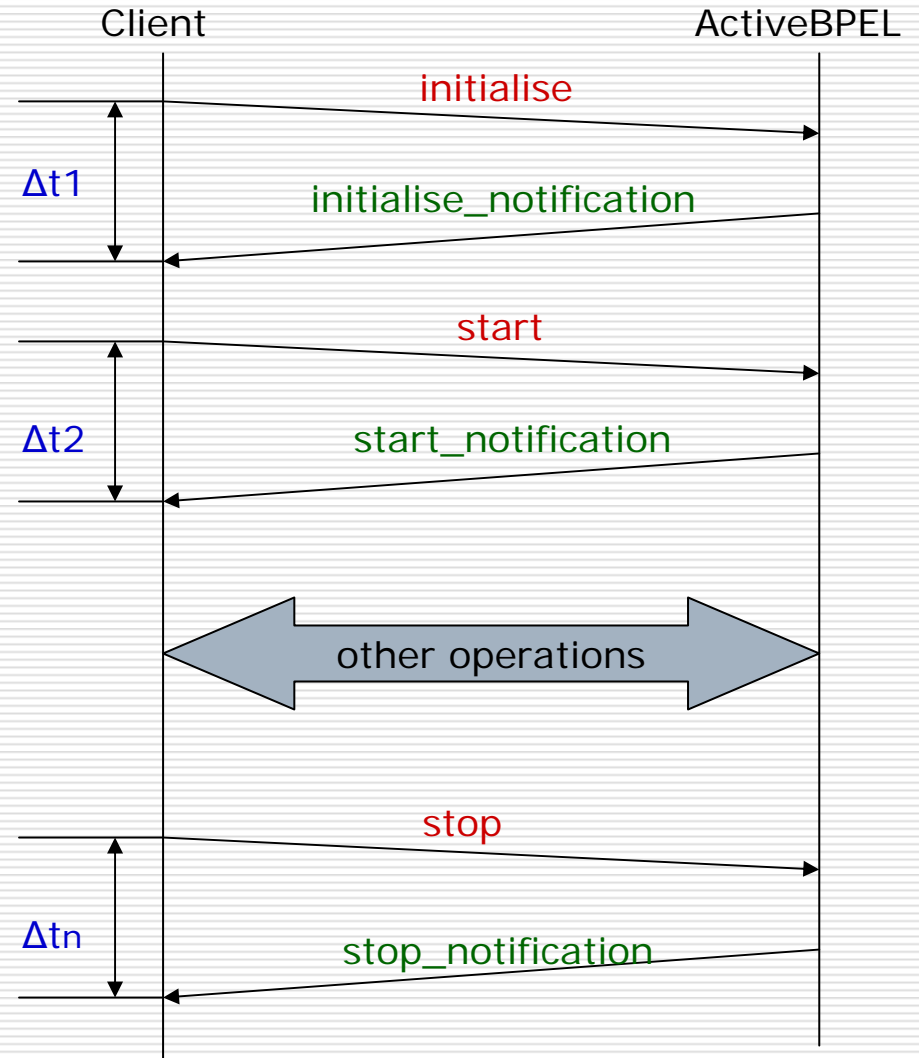
- Test the duration of the initialisation and start operations under different workloads

## □ Workload

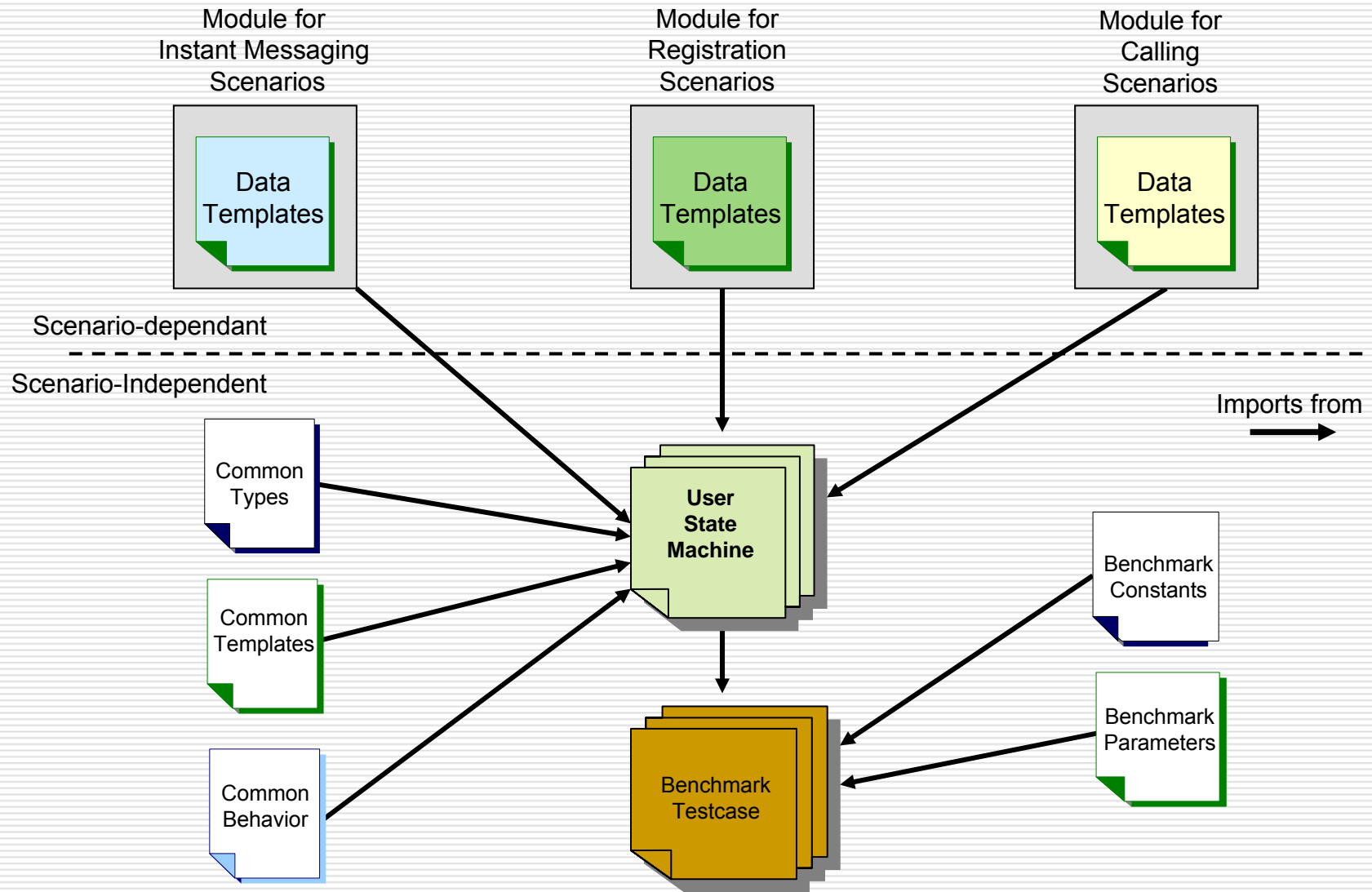
- 10..1000 Clients
- 10..50 new scenarios/second
- 10..30 minutes

## □ Measurements

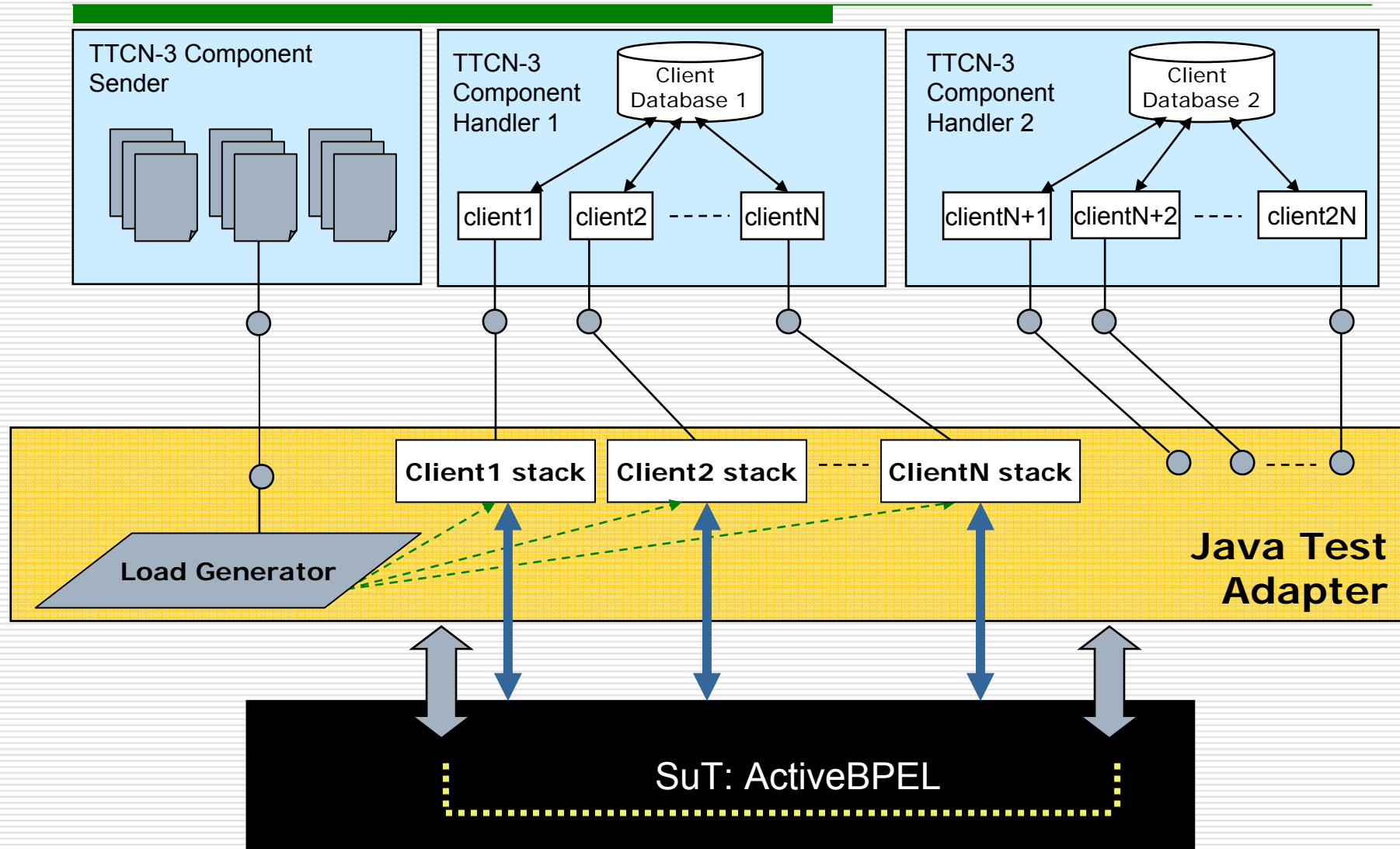
- Simultaneous users of a service
- Messages per second
- Latency (round-trip time)
- Throughput



# TTCN-3 ATS Architecture



# Test Configuration



# TTCN-3 Message Types

// Request/Response Types

```
type record EA_Initialisation_RequestType {
  String clientid,
  ProcessInitialisationRequestType ProcessInitialisationRequest
}
```

```
type record EA_Initialisation_NotificationType {
  String clientid,
  ProcessInitialisationResponseType ProcessInitialisationResponse
}
```

// Sub-Types

```
type record ProcessInitialisationRequestType {
  String customer_id,
  String authority_id,
  String process_type,
  String initialtime
}
```

```
type record ProcessInitialisationResponseType {
  String process_id,
  String state,
  String initialtime
}
```

# TTCN-3 State Handling

```

alt {
  [] EA_AdapterServicePort_PORT.receive(EA_Initialisation_Notification)
    -> value initialisationResponse {
      ... process message ...
      repeat;
    }
  [] EA_AdapterServicePort_PORT.receive(EA_Start_Notification)
    -> value startResponse {
      ... process message ...

      makeAvailable(startResponse.clientid); // release client
      repeat;
    }
  ... other states ...

  [] port2Adapter.receive(NO_USER_FOUND) {
    ... process error
    log("ERROR: not enough users or too many errors");
    stop;
  }
  [] port2Adapter.receive(BENCHMARK_TERMINATED) {
    log("test finished... wait for results");
    terminationTimer.start;
    repeat;
  }
  [] terminationTimer.timeout {
    stop;
  }
}

```

# TTCN-3 Message Handling

```
// Client <----- REQUEST ----- BPEL Engine
[] EA_AdapterServicePort_PORT.receive(EA_Initialisation_Notification) ->
    value initialisationNotification {

    // initialize new message with a template
    startRequest := EA_Start_Request;

    startRequest.process_id := initialisationNotification.
        ProcessInitialisationResponse.process_id;
    startRequest.clientid := initialisationNotification.clientid;

    // ... other fields

    clientid := str2int(startRequest.clientid);

    // send new request
    EA_AdapterServicePort_PORT.send(startRequest) to clientid;

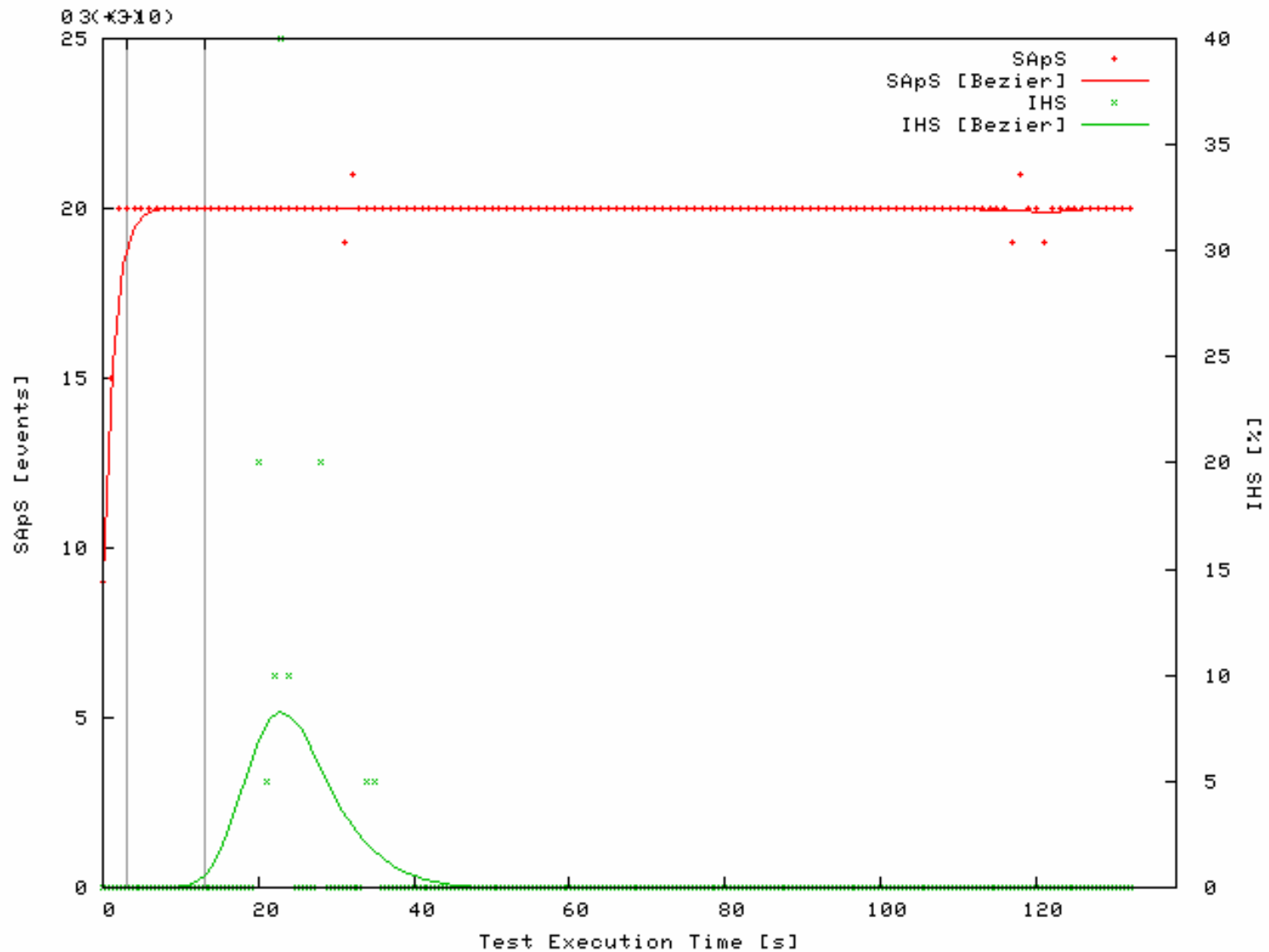
    // return to process a new SUT message
    repeat;
}
}
```

# Experimental Observations

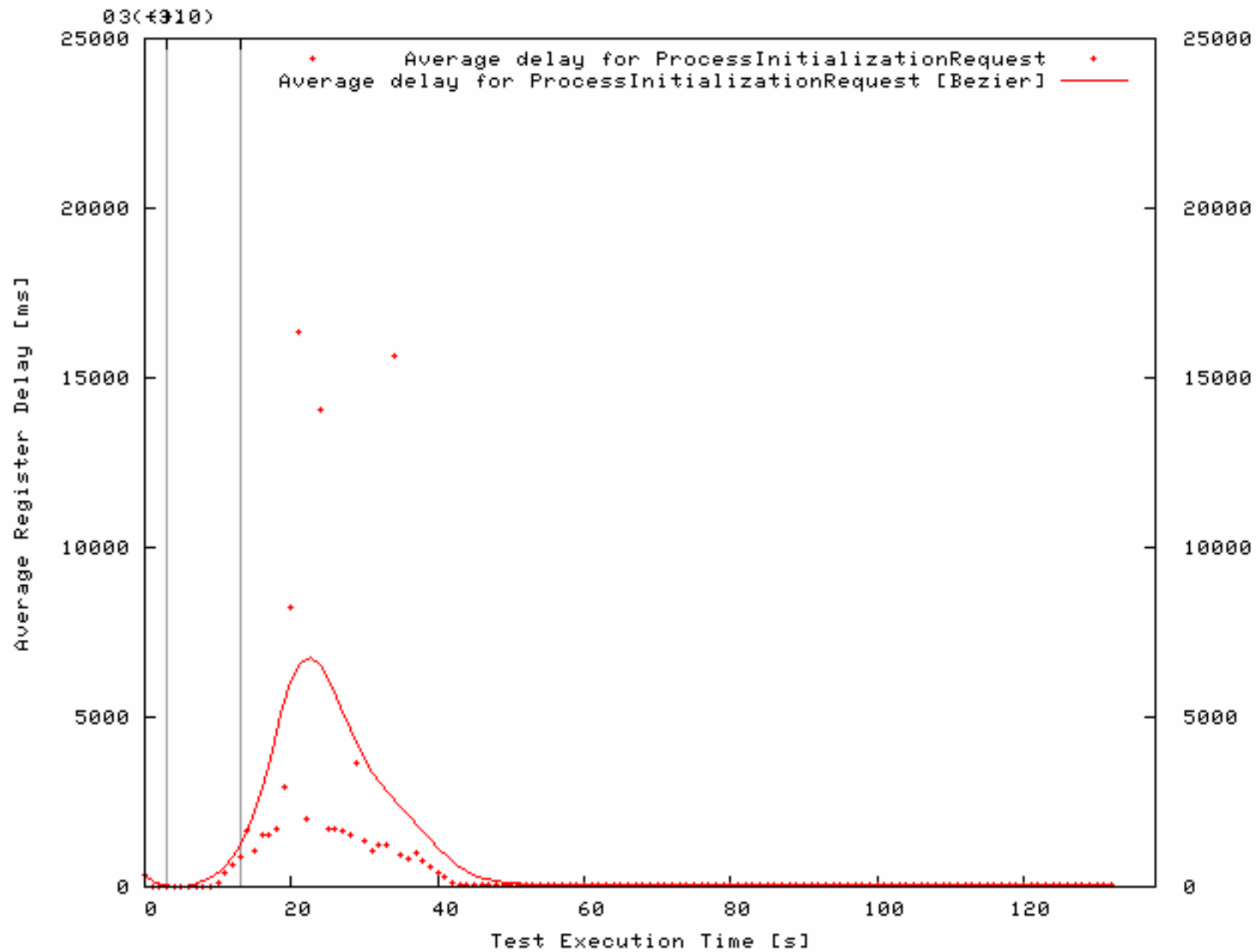
- The results depend on:
  - Number of parallel invocations
  - Concurrent processes
  
- Performance test interruption reasons
  - Tomcat thread limitations (default=150)
  - ActiveBPEL max thread pool (default = 1000)
  - Out of memory (for large number of threads, long durations, etc)



# Example Load Intensity = 20 SAPS



# Average Latency of Process Initialization



- A general benchmarking framework based on
  - test scenarios, test procedures, metrics and design objectives, and ...
  - instantiated for the standardized test language TTCN-3
- Although based on simple scenarios ...
  - the benchmark allows comparison between different BPEL engines + hardware configurations
  - easy to extend with more states and interactions
- Next steps
  - More complex scenarios involving web service orchestration
  - Parallel scenarios