BENCHFLOW

A FRAMEWORK FOR BENCHMARKING BPMN 2.0 WORKFLOW MANAGEMENT SYSTEMS

Vincenzo Ferme, Ana Ivanchikj, Cesare Pautasso
Faculty of Informatics
University of Lugano (USI)
Switzerland
BPMN 2.0: A Widely Adopted Standard

![Graph showing the number of BPMN 2.0 WfMSs from 2009 to 2015](https://en.wikipedia.org/wiki/List_of_BPMN_2.0_engines)

**Year of the First Version Supporting BPMN 2.0**
- 2009: BETA BPMN 2.0
- 2010: BPMN 2.0
- 2011: BPMN 2.0
- 2012: BPMN 2.0
- 2013: BPMN 2.0
- 2014: ISO/IEC 19510 BPMN 2.0.2
- 2015: ISO/IEC 19510 BPMN 2.0.2

**Number of BPMN 2.0 WfMSs**
- 2009: 0
- 2010: 5
- 2011: 10
- 2012: 15
- 2013: 20
- 2014: 25

**Grand Total**: 21
Workflow Management System’s Main Components

WES

Application Server

Job Executor

Core Engine
Workflow Management System’s Main Components

WES

Application Server

Process Navigator

Job Executor

Core Engine

WfMS Components » WfMSs Diversification » BenchFlow » Requirements » BenchFlow Framework » Experiments » Future Work » …
Workflow Management System’s Main Components

WES

- Task Dispatcher
- Process Navigator
- Job Executor
- Core Engine

Users

Application

Application Server

WFMS Components » WFMSs Diversification » BenchFlow » Requirements » BenchFlow Framework » Experiments » Future Work » …
Workflow Management System’s Main Components

WES

- Task Dispatcher
- Process Navigator
- Job Executor
- Core Engine
- Service Invoker
- Event Handler

Application Server

Users

Web Service

Application

Workflow Management System’s Main Components

WfMS Components » WfMSs Diversification » BenchFlow » Requirements » BenchFlow Framework » Experiments » Future Work » …
Workflow Management System’s Main Components

WES

- Task Dispatcher
- Process Navigator
- Job Executor
- Core Engine
- Service Invoker
- Event Handler
- Transaction Manager
- Persistent Manager

Application Server

Application

Users

Web Service

Instance Database

DBMS

Context » WiMS Components » WiMSs Diversification » BenchFlow » Requirements » BenchFlow Framework » Experiments » Future Work » ...
Workflow Management System's Diversification

Supported Languages
- BPMN, BPEL, Petri-Nets, YAML

System's Architecture
- Distributed workflow support
- Migrating workflow objects support
- Transactional workflow support

Functionalities
- Dynamic workflow changes
- Integration capabilities

Deployment Infrastructure
- Standalone
- Cluster Deployment
- Cloud Deployment
- Mobile Deployment
Workflow Management System’s Diversification

**Supported Languages**
- BPMN, BPEL, Petri-Nets, YAML

**System’s Architecture**
- Distributed workflow support
- Migrating workflow objects support
- Transactional workflow support

**Functionalities**
- Dynamic workflow changes
- Integration capabilities

**Deployment Infrastructure**
- Standalone
- Cluster Deployment
- Cloud Deployment
- Mobile Deployment
Workflow Management System’s Diversification

Supported Languages
- BPMN, BPEL, Petri-Nets, YAML

Functionalities
- Dynamic workflow changes
- Integration capabilities

System’s Architecture
- Distributed workflow support
- Migrating workflow objects support
- Transactional workflow support

Deployment Infrastructure
- Standalone
- Cluster Deployment
- Cloud Deployment
- Mobile Deployment
Workflow Management System’s Diversification

Supported Languages
- BPMN, BPEL, Petri-Nets, YAML

Functionalities
- Dynamic workflow changes
- Integration capabilities

System’s Architecture
- Distributed workflow support
- Migrating workflow objects support
- Transactional workflow support

Deployment Infrastructure
- Standalone
- Cluster Deployment
- Cloud Deployment
- Mobile Deployment
The BenchFlow Project

“Design the first benchmark to assess and compare the performance of WfMSs that are compliant with Business Process Model and Notation 2.0 standard.”
BenchFlow Framework: Requirements & Functionalities

**System Under Test (SUT)**

- Automate the SUT deployment
- Simplify the SUT’s deployment configuration
- Adapt to different API provided by different WfMSs
- Deal with the asynchronous execution of business processes
BenchFlow Framework: Requirements & Functionalities

Performance Benchmark

• Simulate all the entities interacting with the WfMS

• Accomodate and automate different kinds of performance test:
  • Ensure reliable execution
  • Ensure repeatability
  • Automate the performance data collection and analyses

Similar Tools:
SOABench, SOArMetrics, Betsy, LoadUI + SoapUI
BenchFlow Framework: Requirements & Functionalities

Performance Benchmark

- Simulate all the entities interacting with the WfMS
- Accomodate and automate different kinds of performance test:
  - Ensure reliable execution
  - Ensure repeatability
- Automate the performance data collection and analyses

Similar Tools:

SOABench, SOArMetrics, Betsy, LoadUI + SoapUI
BenchFlow Framework

- Servers
- Containers
- Web Service
- DBMS
- Faban Drivers
- harness
- Performance Metrics
- Performance KPIs
- Analysers
- Data Cleaners
- WES
- Instance Database
- Faban
- docker
BenchFlow Framework

- Servers
- Containers

Test Execution

Faban Drivers
BenchFlow Framework

Servers

Containers

Faban Drivers

Test Execution

BenchFlow Framework
BenchFlow Framework

- Servers
- Containers
- Web Service
- WES
- DBMS
- Faban Drivers
- Context
- Test Execution
- Adapters
- MONITOR
BenchFlow Framework

- Servers
- Containers

- Web Service
- DBMS
- WES
- Instance Database

- Faban Drivers
- harness

- Test Execution

- Adapters

- Data Mappers

- MONITOR

- Analyses
  - Performance Metrics
  - Performance KPIs

- ANALYSERS

- DATA CLEANERS

- COLLECTORS
Performance Metrics and KPIs

TEST PROCESS
Performance Metrics and KPIs

**TEST PROCESS**

- Empty Script Task
- Wait 2 seconds

**LOAD FUNCTION**

Users

- 0
- 6
- 12
- 18
- 24
- 30

Context » WfMS Components » WfMSs Diversification » BenchFlow » Requirements » BenchFlow Framework » Experiments » Future Work...
Performance Metrics and KPIs

TEST PROCESS

Load Drivers

CPU  64 Cores @ 1400 MHz

RAM  128 GB

TEST ENVIRONMENT
Performance Metrics and KPIs

TEST PROCESS

Empty Script
Task

Wait 2 seconds

LOAD FUNCTION

Users

0 20 60 120 180 240 300

来看看吧

Load Drivers

WES

CPU

64 Cores
@ 1400 MHz

12 Cores
@ 800 MHz

RAM

128 GB

64 GB

TEST ENVIRONMENT

Context » WiMS Components » WiMSs Diversification » BenchFlow » Requirements » BenchFlow Framework » Experiments » Future Work » …
Performance Metrics and KPIs

**TEST PROCESS**

Empty Script
Task
Wait 2 seconds

**LOAD FUNCTION**

<table>
<thead>
<tr>
<th>Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>18</td>
</tr>
<tr>
<td>24</td>
</tr>
<tr>
<td>30</td>
</tr>
</tbody>
</table>

**TEST ENVIRONMENT**

**Load Drivers**

- **CPU**: 64 Cores @ 1400 MHz
- **RAM**: 128 GB

**WES**

- **CPU**: 12 Cores @ 800 MHz
- **RAM**: 64 GB

**DBMS**

- **CPU**: 64 Cores @ 2300 MHz
- **RAM**: 128 GB
Performance Metrics and KPIs

TEST PROCESS

Load Drivers
- CPU: 64 Cores @ 1400 MHz
- RAM: 128 GB

WES
- CPU: 12 Cores @ 800 MHz
- RAM: 64 GB

DBMS
- CPU: 64 Cores @ 2300 MHz
- RAM: 128 GB
- 10 Gbit/s

Load Drivers

TEST ENVIRONMENT

LOAD FUNCTION

Empty Script Task
Wait 2 seconds

TEST PROCESS

CPU
64 Cores @ 1400 MHz
RAM
128 GB

WES
CPU
12 Cores @ 800 MHz
RAM
64 GB

DBMS
CPU
64 Cores @ 2300 MHz
RAM
128 GB
10 Gbit/s
Throughput

\[ \text{Throughput} = \frac{\#BPInstances(bp)}{Time(s)} \]
Instance Duration Time

- Empty Script Task
- Wait 2 seconds

Instance Duration Time
The instance duration is the time difference between the start and the completion of a BP instance. It is presented in the box and whisker plot in Fig. 4(a) for Engine A and Fig. 4(b) for Engine B. This type of plot displays the analyzed data into quartiles where the box contains the second and third quartile, while the median is the line inside the box. The lines outside of the box, called whiskers, show the minimum and maximum value of the data [10].

The measurements show that Engine A scales better since it starts having an unexpected behaviour after 125 concurrent users, while the first execution performance problems of Engine B appear at 50 users, as evident from the instance duration increase of one order of magnitude. In Fig. 5 we report Engine A’s CPU utilization for each of the tests. It is interesting to notice that while the instance duration increases substantially starting from 135 concurrent users (Fig. 4(a)), the CPU utilization decreases, indicating that the slowdown of the WfE is not caused by lack of resources. The same has been verified by checking the CPU/RAM utilization of the DBMS.

After noticing a bottleneck in performance scaling, we investigate the causes. Since only two constructs, a Script task and a Timer event, are used in the experiment BP model, we test the WfE performance in handling each of them individually. The test processes used consist of a Start event, the tested construct and an End event. As per the previously gathered information we focus on the critical number of users (125/135 for Engine A and 25/50 for Engine B). We use the delay metric which compares the expected to the actual duration of the context.
Instance Duration Time and CPU Utilisation

Fig. 4: Aggregated Process Instance Duration Comparison

Fig. 5: Aggregated CPU Usage (Engine A)

Concurrent Users

Fig. 6: Script Task (a) and Timer Event (b) feature comparison

Construct Instance Delay (s)

Fig. 7: Construct Instance Delay (Engine A)

Fig. 8: Construct Instance Delay (Engine B)

5 Conclusion and Future Work

The BenchFlow framework greatly simplifies the performance benchmarking of BPMN2 WfMSs, by abstracting the heterogeneity of their interfaces and automating their deployment, the data collection and the metrics and Key Performance Indicators (KPIs) computation. It does so by relying on Faban and Docker, and by verifying the absence of noise in the performance measurements.

While the complexity of BPMN2 makes it challenging to benchmark the performance of the WfMSs implementing it, the benefits of doing so are evident. The first experimental results obtained with a simple BP model running on two popular open-source WfMSs have uncovered important scalability issues. We have discussed the identified performance bottlenecks with the WfMS vendors who have clarified the probable cause. Namely, in Engine A we have used a different DBMS configuration in the setup of the system. In Engine B the goal of the
Concurrent Users

Fig. 5: Aggregated CPU Usage (Engine A)

Construct Instance Delay (s)

Fig. 6: Script Task and Timer Event feature comparison

The delay measurements (Fig. 6) show that both WfEs handle the Script task efficiently with an average delay below 10ms. The same does not hold for the Timer. For Engine A, the average delay of the Timer at 135 users is by three orders of magnitude greater than at 125 users.

For Engine B, the delay increases by two orders of magnitude between 25 and 50 users. The observed system behaviour could be due to an excessive overhead introduced by concurrently handling many Timers, which could cause growth in the Timers queue thus postponing their execution and increasing their delay.

5 Conclusion and Future Work

The BenchFlow framework greatly simplifies the performance benchmarking of BPMN2 WfMSs, by abstracting the heterogeneity of their interfaces and automating their deployment, the data collection and the metrics and Key Performance Indicators (KPIs) computation. It does so by relying on Faban and Docker, and by verifying the absence of noise in the performance measurements.

While the complexity of BPMN2 makes it challenging to benchmark the performance of the WfMSs implementing it, the benefits of doing so are evident. The first experimental results obtained with a simple BP model running on two popular open-source WfMSs have uncovered important scalability issues. We have discussed the identified performance bottlenecks with the WfMS vendors who have clarified the probable cause.
Future Work

Experiments

• Perform the first *real-world* experiments
• Increase the number of supported WfMSs
• Simplify and automate the execution of common performance tests: Load Test, Spike Test, Scalability Test, …

BenchFlow Framework

• Release a development version on GitHub
  benchflow
Highlights
Highlights

Workflow Management System
Highlights

Workflow Management System

BenchFlow Project

"Design the first benchmark to assess and compare the performance of WfMSs that are compliant with Business Process Model and Notation 2.0 standard."

WfMS Components » WfMSs Diversification » BenchFlow » Requirements » BenchFlow Framework » Experiments » Future Work » Highlights
Highlights

- **Workflow Management System**
- **BenchFlow Project**
- **BenchFlow Framework**
Highlights

Workflow Management System

BenchFlow Project

BenchFlow Framework

Proof of Concept

"Design the first benchmark to assess and compare the performance of WfMSs that are compliant with Business Process Model and Notation 2.0 standard."

Performance Metrics and KPIs

Load Drivers
- CPU: 64 Cores @ 400 MHz
- DBMS: 64 Cores @ 2300 MHz

WfMS Components
- WfMSs Diversification
- BenchFlow
- Requirements
- BenchFlow Framework
- Experiments
- Future Work
Call for Action

Process Models
• We want to characterise the Workload using Real-World process models
• Send us your executable BPMN process models, even anonymised!

Execution Logs
• We want to characterise the Workload using Real-World behaviours
• Send us your execution logs, even anonymised!
BENCHFLOW

A FRAMEWORK FOR BENCHMARKING BPMN 2.0 WORKFLOW MANAGEMENT SYSTEMS

Vincenzo Ferme (@VincenzoFerme), Ana Ivanchikj, Cesare Pautasso
Faculty of Informatics
University of Lugano (USI)
Switzerland

vincenzo.ferme@usi.ch
Join Us @ ICWE 2016 in Lugano!

6-9 June 2016, USI Lugano, Switzerland

http://icwe2016.inf.usi.ch