A CONTAINER-CENTRIC METHODOLOGY FOR BENCHMARKING WORKFLOW MANAGEMENT SYSTEMS

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

Marigianna Skouradaki, Prof. Frank Leymann
Institute of Architecture of Application Systems
University of Stuttgart, Germany
The BenchFlow Project

“Design and implement the first benchmark to assess and compare the performance of WfMSs that are compliant with Business Process Model and Notation 2.0 standard.”
What is a Workflow Management System?
What is a Workflow Management System?
What is a Workflow Management System?
What is a Workflow Management System?

**Diagram:**
- **WfMS**
- **Application Server**
- **Instance Database**
- **DBMS**
- **Users**
- **Applications**
- **Web Service**

**Process Diagram:**
- **A** → **B** → **D**
- **A** → **C**

**Context:**
- Benchmarking Requirements
- Methodology Overview
- Methodology Details
- Advantage of Containers
- 1st Application
- Future Work

**Vincenzo Ferme**
Many Vendors of BPMN 2.0 WfMSs

Number of BPMN 2.0 WfMSs

Year of the First Version Supporting BPMN 2.0

https://en.wikipedia.org/wiki/List_of_BPMN_2.0_engines

BETA
BPMN 2.0
Aug 2009

BPMN 2.0
Jan 2011

ISO/IEC 19510
BPMN 2.0.2
Jan 2014
Benchmarking Requirements

- Relevant
- Representative
- Portable
- Scalable
- Simple
- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable

- S. E. Sim, S. Easterbrook et al., *Using benchmarking to advance research: A challenge to software engineering*, 2003
Why a new Methodology?

No available methodologies involving vendors for both defining a standard benchmark and benchmarking production systems.
Why a new Methodology?

No available methodologies involving vendors for both defining a standard benchmark and benchmarking production systems

Already standard benchmarks
Why a new Methodology?

No available methodologies involving vendors for both defining a standard benchmark and benchmarking production systems

Already standard benchmarks

No interaction with Vendors
Why a Container-Centric Methodology?

- Relevant
- Representative
- Portable
- Scalable
- Simple

- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable

Emerging Technology
Why a Container-Centric Methodology?

- Relevant
- Representative
- Portable
- Scalable
- Simple
- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable

Emerging Technology
Lightweight
Why a Container-Centric Methodology?

- Relevant
- Representative
- Portable
- Scalable
- Simple
- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable

Emerging Technology
Lightweight
Negligible Performance Impact
Why a Container-Centric Methodology?

- Relevant
- Representative
- **Portable**
- Scalable
- Simple
- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable

Emerging Technology

Lightweight

Negligible Performance Impact
Why a Container-Centric Methodology?

- Relevant
- Representative
- Portable
- Scalable
- Simple
- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable

Emerging Technology
Lightweight
Negligible Performance Impact
Why a Container-Centric Methodology?

- Relevant
- Representative
- Portable
- Scalable
- Simple

- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable

Emerging Technology
Lightweight
Negligible Performance Impact
Why a Container-Centric Methodology?

- Relevant
- Representative
- Portable
- Scalable
- Simple
- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable

Emerging Technology
Lightweight
Negligible Performance Impact
Why a Container-Centric Methodology?

- Relevant
- Representative
- Portable
- Scalable
- Simple
- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable

Emerging Technology
Lightweight
Negligible Performance Impact
What about the other Requirements?

- Relevant
- Representative
- Portable
- Scalable
- Simple

- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable
Benchmarking Choreography

Benchmarking Methodology

Agreement Proposal

Containerized WfMS Request

Vendor

Vendor

Vendor

Verified Benchmark Results

Results Verification Request

Draft Benchmark Results

BenchFlow

Provide Benchmarking Methodology

Agree on adding Vendor's WfMS to the Benchmark

Provide Containerized Distribution of WfMS

BenchFlow

BenchFlow

BenchFlow

Community

Vendor

Vendor

Vendor

Publish Benchmark Results

Are the Results Valid?

Verify the Benchmark Results

Provide Draft Benchmark Results

Valid

Not Valid

Results Verification Outcome
Benchmarking Methodology

provide the methodology

BenchFlow

Provide Benchmarking Methodology

Vendor

Agreement Proposal

BenchFlow

Agree on adding Vendor's WfMS to the Benchmark

Vendor

Signed Agreement

Verified Benchmark Results

Results Verification Report
The Benchmarking Process

Input/Process/Output Model

Test Types → Workload Model → Workload → Configurations → WfMS → Derive KPIs → Metrics KPIs
The Benchmarking Process

Input/Process/Output Model

Workload Mix

Test Types → Workload Model

20% → D

80% → B → C

Configurations

Workload → WfMS → Derive → Metrics KPIs

Context » Benchmarking Requirements » Methodology Overview » Methodology Details » Advantage of Containers » 1st Application » Future Work
The Benchmarking Process

Workload Model

Input/Process/Output Model

Workload Mix

Test Types

Test Data Load Functions

Configurations

Workload

WfMS

Derive

Metrics KPIs

Context » Benchmarking Requirements » Methodology Overview » Methodology Details » Advantage of Containers » 1st Application » Future Work

Vincenzo Ferme
The Benchmarking Process

*Input/Process/Output Model*

- **Configurations**
- **Workload**
- **WfMS**
- **Metrics KPIs**
- **Containers**

**Workload Mix**
- 80% A, B, C
- 20% D

**Test Types**

**Test Data Load Functions**

**Context**
- Benchmarking Requirements
- Methodology Overview
- Methodology Details
- Advantage of Containers
- 1st Application
- Future Work
The Benchmarking Process

*Input/Process/Output Model*

**Workload Mix**

- 80% A
- 20% D

**Test Types**

**Workload Model**

**Test Data**

**Load Functions**

**Context**

- Benchmarking Requirements
- Methodology Overview
- Methodology Details
- Advantage of Containers
- 1st Application
- Future Work

**WfMS Configurations**

- Derive Metrics KPIs
  - Throughput
  - Workflow Instance Duration

**Performance Data**

- Workload Instance Duration
- Throughput

**Containers**
#env-environment-variables

specifying the volumes to be exposed

containers must allow to issue the WfMS configurations
different configurations. To do that, the Con-

a "getting started" configuration, or production-grade

for example, the configuration provided to users as

mance of their WfMS with different configurations,

parameters, except the ones required in order to cor-

figuration, i.e., the configuration in which the WfMS

cluded in the benchmark, there must be a default con-

ity of the results. For each WfMS version to be in-

istries are a solution that can work with vendors of

Dockerfile (Turnbull, 2014, ch. 4). While private reg-

same applies to the Containers' definition file, i.e., the

registry

should be publicly available (e.g., at the Docker Hub

Container distributions. The containerized WfMS

pending on the WfMS's architecture. The DBMS

should be

Vincenzo Ferme

Main Agreement Points:

• Production Stable Release
• Provide defined APIs
• Share Containerised WfMS
• Authorise Publishing of Results

Agreement Proposal

BenchFlow

Agree on adding Vendor's WfMS to the Benchmark

Vendor

Signed Agreement

Verified Benchmark Results

Results Verification R

Context » Benchmarking Requirements » Methodology Overview » Methodology Details » Advantage of Containers » 1st Application » Future Work  📭 Vincenzo Ferme
Benchmarking Methodology

containerised WfMSs

Context » Benchmarking Requirements » Methodology Overview » Methodology Details » Advantage of Containers » 1st Application » Future Work

1) the used DB, i.e., DB driver, url, username and password, DB schema. Alternatively, the DBMS Container) has to create the WfMS’s DB schema depending on the WfMS’s architecture. The DBMS configuration is required.

The WES should be publicly available (e.g., at the Docker Hub (Turnbull, 2014, ch. 4)), or the Benchflow team should be granted access to a private registry used by the vendor. The logging level of the WES, and the application stack configurations for real-life usage, they can also profile the performance of their WfMS with different configurations.

However, if vendors want to benchmark the performance of their WfMS with different configurations, they may want to provide a Containerized WfMS. The Containerized WfMS has to be publicly available and may be provided as part of the deployment. The Benchflow team may request access to the Containerized WfMS for benchmarking.

The Containerized WfMS may be used with admin privileges for the DBMS. Alternatively, the Vendor may simply provide the files needed to create the Containerized WfMS. The Vendor’s WfMS may be added to the Benchmark, and the WfMS log files, and to access all the data useful to the Vendor, on the host operating system. Precisely, the WES should be able to start without modifying any configuration.
Benchmarking Methodology

requirements from the WfMS

CORE

NON-CORE
Benchmarking Methodology

requirements from the WfMS

Initialisation APIs

Deploy Process

Start Process

Instance

CORE

NON-CORE
Benchmarking Methodology
requirements from the WfMS

Initialisation APIs

Deploy Process
Start Process
Instance

User APIs

Create User
Create Group
Pending User Tasks
Claim Task
Complete Task

CORE
NON-CORE
Benchmarking Methodology
requirements from the WfMS

Initialisation APIs
- Deploy Process
- Start Process Instance

User APIs
- Create User
- Pending User Tasks
- Claim Task

Web Service APIs
- Create Group
- Invoke WS
- Complete Task

WfMS

CORE

NON-CORE

Configurations
WfMS
Benchmarking Methodology

requirements from the WfMS

Initialisation APIs

Deploy Process
Start Process Instance

User APIs
Create User
Pending User Tasks
Claim Task
Complete Task

Web Service APIs
Create Group
Invoke WS

Event APIs
Pending Event Tasks
Issue Event

CORE

NON-CORE
### Benchmarking Methodology

**Requirements from the WfMS**

**Functionality**

- **Initialisation**
  - Deploy a process
  - Start a process instance

- **Deployment**
  - Access pending tasks
  - Pending tasks IDs

- **User ID**
  - Create a user
  - Claim a task*
  - Access pending tasks

- **Process ID**
  - Create a group of users
  - Complete a task
  - Pending tasks IDs

- **Response Data**
  - MtIDs
  - UtIDs
  - CeIDs
  - PiIDs
  - PdIDs

---

<table>
<thead>
<tr>
<th><strong>Non-core APIs</strong></th>
<th><strong>Core APIs</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Optional depending on the WfMS implementation</em></td>
<td></td>
</tr>
<tr>
<td>Web service APIs</td>
<td>Core APIs</td>
</tr>
<tr>
<td>Event APIs</td>
<td>Initialisation</td>
</tr>
<tr>
<td>User APIs</td>
<td>Deploy a process</td>
</tr>
<tr>
<td>User ID</td>
<td>Start a process instance</td>
</tr>
<tr>
<td>Process ID</td>
<td></td>
</tr>
<tr>
<td>Pending events IDs</td>
<td>MtIDs</td>
</tr>
<tr>
<td>Access pending events</td>
<td>UtIDs</td>
</tr>
<tr>
<td>Issue events</td>
<td>CeIDs</td>
</tr>
<tr>
<td>Map tasks to Web service endpoints</td>
<td>PiIDs</td>
</tr>
<tr>
<td>Claim a task</td>
<td>PdIDs</td>
</tr>
<tr>
<td>Create a group of users</td>
<td>Process ID</td>
</tr>
<tr>
<td>Complete a task</td>
<td></td>
</tr>
<tr>
<td>Pending tasks IDs</td>
<td></td>
</tr>
</tbody>
</table>
### Benchmarking Methodology

*containerised WfMSs*

- At least two containers
- DBMS can refer to existing one publicly available
- Provide a ready to use default configuration (at least)
- Configurability of: DBMS, WfMS, Logging Level (at least)
Benchmarking Methodology

executing the benchmark and providing results
BenchFlow Framework

architecture

Test Execution

Servers  Containers

Web Service

WfMS

Web Service

Faban Drivers

Containers

Kafka

Spark

Instance Database

Minio

Performance Metrics

Performance KPIs

ANALYSERS

DATA TRANSFORMERS

harness

Faban

docker

Faban Drivers

[Context » Benchmarking Requirements » Methodology Overview » Methodology Details » Advantage of Containers » 1st Application » Future Work] 23

Vincenzo Ferme
Performance Metrics and KPIs

Load Driver

Application Server

WfMS

Instance Database

DBMS

Users

Web Service
Performance Metrics and KPIs

Load Driver

Metrics and KPIs

• Engine Level
Performance Metrics and KPIs

Metrics and KPIs

- Engine Level
- Process Level
Performance Metrics and KPIs

Metrics and KPIs

- Engine Level
- Process Level
- Feature Level
Performance Metrics and KPIs

**Metrics and KPIs**

- **Engine Level**
- **Process Level**
- **Feature Level**
- **Interactions**
Performance Metrics and KPIs

- Engine Level
- Process Level
- Feature Level
- Interactions
- Environments
Executing the Benchmark

*minimal data requirements*

**Accessibility of the Data**
Executing the Benchmark

*minimal data requirements*

### Accessibility of the Data

- **WfMS**
- **DBMS**

### Availability of Timing Data

- **Workflow & Construct:**
  - Start Time
  - End Time
  - [Duration]

**DBMS**

**WfMS**
Executing the Benchmark

minimal data requirements

Accessibility of the Data
Availability of Timing Data

- Workflow & Construct:
  - Start Time
  - End Time
  - [Duration]

Availability of Execution State
State of the workflow execution. E.g., running, completed, error
Benchmarking Methodology

mock example of benchmark results
Benchmarking Methodology

*mock example of benchmark results*

- **Workload Model**
  - MySQL
  - WfMS A

- **Hardware Configuration**
  - **O.S.**: Ubuntu 14.04.01
  - **J.V.M.**: Oracle Serv. 7u79
  - **App. Server**: Ap. Tomcat 7.0.62

- **MySQL**: Community Server 5.6.26

- **WfMS A**: v7.0.1
Benchmarking Methodology

mock example of benchmark results

Workload Model

MySQL

O.S.: Ubuntu 14.04.01
J.V.M.: Oracle Serv. 7u79

App. Server:

MySQL: Community Server 5.6.26
WfMS A: v7.0.1

Hardware Configuration

Metrics and KPIs
Benchmarking Methodology
executing the benchmark and providing results
Benchmarking Methodology

*publish benchmark results*

Verified Benchmark Results

BenchFlow

Publish Benchmark Results

Community

Are the Results Valid?

Valid

Invalid
Advantages of using Containers

- Accomplish some Benchmarking Requirement: Portability, Repeatability, Accessibility, Efficiency
Advantages of using Containers

• Accomplish some Benchmarking Requirement: Portability, Repeatability, Accessibility, Efficiency

• Common way to deploy systems provided by different vendors

Docker Compose

Docker Swarm
Advantages of using Containers

- Accomplish some Benchmarking Requirement: Portability, Repeatability, Accessibility, Efficiency
- Common way to deploy systems provided by different vendors
- Standard APIs to access Environment Metrics

Docker Compose  
Docker Swarm
First Application of the Methodology

[CAiSE ’16]
First Application of the Methodology

[CAiSE '16]

Workload
First Application of the Methodology

[CAiSE ’16]

![Diagram showing Workload and 3 WfMSs]
First Application of the Methodology

[CAiSE ’16]

Workload

3 WfMSs

Metrics

• Engine Level
• Process Level
• Environment
First Application of the Methodology

[CAiSE '16]

Results: relevant differences among WfMSs

• Engine Level
• Process Level
• Environment
Future Work

• Continue to Apply and Improve the Methodology
Future Work

- Continue to Apply and Improve the Methodology
- Involve more Vendors and Researchers as part of the Benchmarking Effort
Future Work

• Continue to Apply and Improve the Methodology

• Involve more Vendors and Researchers as part of the Benchmarking Effort
Future Work

• Continue to Apply and Improve the Methodology

• Involve more Vendors and Researchers as part of the Benchmarking Effort
Future Work

• Continue to Apply and Improve the Methodology

• Involve more Vendors and Researchers as part of the Benchmarking Effort
Future Work

- Continue to Apply and Improve the Methodology
- Involve more Vendors and Researchers as part of the Benchmarking Effort

1\textsuperscript{st} International Workshop on Performance and Conformance of Workflow Engines

Workshop
September 5\textsuperscript{th}, 2016

ESOCC 2016
Vienna, Austria

http://uniba-dsg.github.io/peace-ws/
Highlights
Highlights

Benchmarking Requirements

- Relevant
- Representative
- Portable
- Scalable
- Simple
- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable

S. E. Sim, S. Easterbrook et al, *Using benchmarking to advance research: A challenge to software engineering*, 2003
Highlights

Benchmarking Requirements

- Relevant
- Representative
- Portable
- Scalable
- Simple
- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable

- K. Huppler, The art of building a good benchmark, 2009
- S. E. Sim, S. Easterbrook et al., Using benchmarking to advance research: A challenge to software engineering, 2003

Benchmarking Methodology
Benchmarking Requirements

- Relevant
- Representative
- Portable
- Scalable
- Simple
- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable

• K. Huppler, *The art of building a good benchmark*, 2009
• S. E. Sm, S. Easterbrook et al., *Using benchmarking to advance research: A challenge to software engineering*, 2003

Advantages of using Containers

- Accomplish some Benchmarking Requirement:
  - Portability, Repeatability, Accessibility, Efficiency
- Common way to deploy systems provided by different vendors

- Standard APIs to access Environment Metrics

Docker Compose

Docker Swarm

Highlights

Benchmarking Requirements

Benchmarking Choreography

Advantages of Containers

Benchmarking Methodology
**Highlights**

**Benchmarking Requirements**
- Relevant
- Representative
- Portable
- Scalable
- Simple
- Repeatable
- Vendor-neutral
- Accessible
- Efficient
- Affordable

* S. E. Sim, S. Easterbrook et al., *Using benchmarking to advance research: A challenge to software engineering*, 2003

**Advantages of using Containers**
- Accomplish some Benchmarking Requirement: Portability, Repeatability, Accessibility, Efficiency
- Common way to deploy systems provided by different vendors
  - Docker Compose
  - Docker Swarm
- Standard APIs to access Environment Metrics

**Future Work**
- Continue to Apply and Improve the Methodology
- Involve more Vendors and Researchers as part of the Benchmarking Effort
A CONTAINER-CENTRIC METHODOLOGY FOR BENCHMARKING WORKFLOW MANAGEMENT SYSTEMS

benchflow

http://benchflow.inf.usi.ch

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

Marigianna Skouradaki, Prof. Frank Leymann
Institute of Architecture of Application Systems
University of Stuttgart, Germany
BACKUP SLIDES

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

Marigianna Skouradaki, Prof. Frank Leymann
Institute of Architecture of Application Systems
University of Stuttgart, Germany
Published Work

[SSP ’14]

[BTW ’15]

[ICPE ’15]
Published Work

[CLOSER ’15]

[SOSE ’15]

[BPM ’15]
Published Work

[BPMD ’15]

[ICPE ’16]

[CAiSE ’16]
Docker Performance

[IBM ’14]

“...Our results show that containers result in equal or better performance than VMs in almost all cases.

“...Although containers themselves have almost no overhead, Docker is not without performance gotchas. Docker volumes have noticeably better performance than files stored in AUFS. Docker’s NAT also introduces overhead for workloads with high packet rates. These features represent a tradeoff between ease of management and performance and should be considered on a case-by-case basis.

BenchFlow Configures Docker for Performance by Default