Workload Characterization of the SPECjms2007 Benchmark

Kai Sachs[1], Samuel Kounev[1,2], Jean Bacon[2], and Alejandro Buchmann[1]

[1] Databases and Distributed Systems Group, TU Darmstadt, Germany
[2] Computer Laboratory, University of Cambridge, UK

Abstract. Message-oriented middleware (MOM) is at the core of a vast number of financial services and telco applications, and is gaining increasing traction in other industries, such as manufacturing, transportation, health-care and supply chain management. There is a strong interest in the end user and analyst communities for a standardized benchmark suite for evaluating the performance and scalability of MOM. In this paper, we present a workload characterization of the SPECjms2007 benchmark which is the world's first industry-standard benchmark specialized for MOM. In addition to providing standard workload and metrics for MOM performance, the benchmark provides a flexible performance analysis framework that allows users to customize the workload according to their requirements. The workload characterization presented in this paper serves two purposes i) to help users understand the internal components of the SPECjms2007 workload and the way they are scaled, ii) to show how the workload can be customized to exercise and evaluate selected aspects of MOM performance. We discuss how the various features supported by the benchmark can be exploited for in-depth performance analysis of MOM infrastructures.

1 Introduction

Message-oriented middleware (MOM) is increasingly adopted as an enabling technology for modern event-driven applications like stock trading, event-based supply chain management, air traffic control and online auctions to name just a few. Novel messaging applications, however, pose some serious performance and scalability challenges. For example, the next generation of event-driven supply chain management based on RFID technology [6] (for instance SAP's AutoID infrastructure [3]) will be highly reliant on scalable and efficient backend systems to support the processing of acquired real-time data and its integration with enterprise applications and business processes [12]. Large retailers, like Wal-Mart, Metro or Tesco, are expected to have throughput rates of about 60 billion messages per annum [2]. The performance and scalability of the underlying MOM platforms used to process these messages will be of crucial importance for the successful adoption of such applications in the industry.

To guarantee that applications meet their Quality of Service (QoS) requirements, it is essential that the platforms on which they are built are tested using

benchmarks to measure and validate their performance and scalability. However, if a benchmark is to be useful and reliable, it must fulfill several fundamental requirements [9]. First of all, it must be designed to stress platforms in a manner representative of real-world messaging applications. It must exercise all critical services provided by platforms and must provide a level playing field for performance comparisons. Finally, to be reliable, a benchmark must generate reproducible results and must not have any inherent scalability limitations. While a number of proprietary benchmarks for MOM servers (for example [14, 7, 1, 8]) have been developed and used in the industry for performance testing and product comparisons (see [5, 11, 4]), these benchmarks do not meet the above requirements. The reason is that most of them use artificial workloads that do not reflect any real-world application scenario. Furthermore, they typically concentrate on stressing individual MOM features in isolation and do not provide a comprehensive and representative workload for evaluating the overall MOM server performance. To address these concerns, in September 2005 we launched a project at the Standard Performance Evaluation Corporation (SPEC) with the goal to develop a standard benchmark for evaluating the performance and scalability of MOM products. The new benchmark was called SPECjms2007 and it was developed at SPEC's OSG-Java Subcommittee with the participation of TU-Darmstadt, IBM, Sun, BEA, Sybase, Apache, Oracle and JBoss. SPECjms2007 exercises messaging products through the JMS (Java Message Service) [15] standard interface which is supported by all major MOM vendors.

In this paper, we introduce the SPECjms2007 benchmark and provide a comprehensive characterization of its workload. We start with a brief overview of the benchmark goals and then present the business scenario it models and discuss the way it was implemented. An important advantage of SPECjms2007 is that it allows users to customize the workload to their needs by configuring it to stress selected features of the MOM infrastructure in a way that resembles a given target customer workload. However, in order to exploit this, users need to understand the way the workload is decomposed into components and which performance aspects are exercised by these components. To this end, after discussing the benchmark scenario and its implementation, we present a detailed characterization of the benchmark workload. This characterization, on the one hand, aims to help users gain an in-depth understanding of the SPECjms2007 workload, so that they can interpret the benchmark results correctly. On the other hand, it provides the information needed to enable users to tailor the workload to their own requirements.

The rest of the paper is organized as follows. In Section 2, we briefly discuss the goals of SPECjms2007 and then introduce the business scenario and interactions it models. Following this, in Section 3, we present an in-depth characterization of the SPECjms2007 workload in terms of the number and types of destinations, the interaction mix, the message types, the message sizes and the message delivery modes. We show how the workload can be customized to stress selected performance aspects and discuss two standard strategies for scaling the workload. The paper is wrapped up in Section 4.

2 The SPECjms2007 Benchmark

2.1 Requirements and Goals

The aim of the SPECjms2007 benchmark is to provide a standard workload and metrics for measuring and evaluating the performance and scalability of MOM platforms. To achieve this the SPECjms2007 workload must fulfill several important requirements. First of all, it must be based on a representative workload scenario that reflects the way platform services are exercised in real-life systems. The goal is to allow users to relate the observed behavior to their own applications and environments. Second, the workload should be comprehensive in that it should exercise all platform features typically used in MOM applications including both point-to-point (P2P) and publish/subscribe (pub/sub) messaging. The features and services stressed should be weighted according to their usage in real-life systems. The third requirement is that the workload should be focused on measuring the performance and scalability of the MOM server's software and hardware components. It should minimize the impact of other components and services that are typically used in the chosen application scenario. For example, if a database would be used to store business data and manage the application state, it could easily become the limiting factor of the benchmark as experience with other benchmarks shows [10]. Finally, the SPECjms2007 workload must not have any inherent scalability limitations. The user should be able to scale the workload both by increasing the number of destinations (queues and topics) as well as the message traffic pushed through a destination.

Producing and publishing standard results for marketing purposes will be just one usage scenario for SPECjms2007. Many users will be interested in using the benchmark to tune and optimize their platforms or to analyze the performance of certain specific MOM features. Others could use the benchmark for research purposes in academic environments where, for example, one might be interested in evaluating the performance and scalability of novel methods and techniques for building high-performance MOM servers. All these usage scenarios require that the benchmark framework allows the user to precisely configure the workload and transaction mix to be generated. Providing this configurability is a great challenge because it requires that interactions are designed and implemented in such a way that one could run them in different combinations depending on the desired transaction mix.

2.2 Workload Scenario

The workload scenario chosen for SPECjms2007 models the supply chain of a supermarket company. The participants involved are the supermarket company, its stores, its distribution centers and its suppliers. The scenario offers an excellent basis for defining interactions that stress different subsets of the functionality offered by MOM servers. Moreover, it offers a natural way to scale the workload. The participants involved in the scenario can be grouped into the following four roles:

Company Headquarters (HQ) The company's corporate headquarters are responsible for managing the accounting of the company, managing information about the goods and products offered in the supermarket stores, managing selling prices and monitoring the flow of goods and money in the supply chain.

Distribution Centers (DCs) The distribution centers supply the supermarket stores. Every distribution center is responsible for a set of stores in a given area. The distribution centers in turn are supplied by external suppliers. The distribution centers are involved in the following activities: taking orders from supermarkets, ordering goods from suppliers, delivering goods to supermarkets and providing sales statistics to the HQ (e.g. for data mining).

Supermarkets (SMs) The supermarkets sell goods to end customers. The scenario focuses on the management of the inventory of supermarkets including their warehouses. Some supermarkets are smaller than others, so that they do not have enough room for all products, others may be specialized for some product groups like certain types of food. We assume that every supermarket is supplied by exactly one of the distribution centers.

Suppliers (SPs) The suppliers deliver goods to the distribution centers of the supermarket company. Different suppliers are specialized for different sets of products and they deliver goods on demand, i.e. they must receive an order from the supermarket company to send a shipment.

2.3 Modeled Interactions

SPECjms2007 implements seven interactions between the participants in the supermarket supply chain.

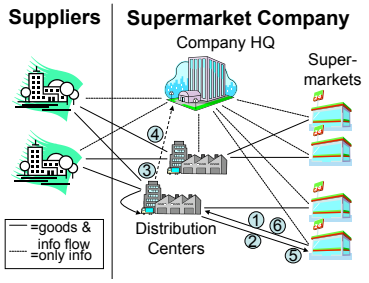

Fig. 1. Interaction 1 - Communication between SM and DC

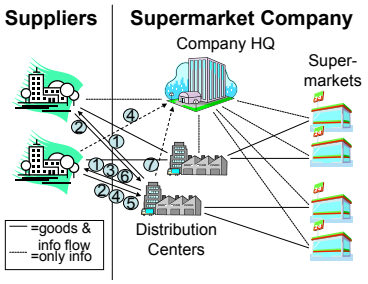

Fig. 2. Interaction 2 - Communication between SP and DC

Interaction 1: Order/Shipment Handling between SM and DC

This interaction exercises persistent P2P messaging between the SMs and DCs. The interaction is triggered when goods in the warehouse of a SM are depleted and the SM has to order from its DC to refill stock. The following steps are followed as illustrated in Figure 1:

1. A SM sends an order to its DC.
2. The DC sends a confirmation to the SM and ships the ordered goods.
3. Goods are registered by RFID readers upon leaving the DC warehouse.
4. The DC sends information about the transaction to the HQ (sales statistics).
5. The shipment arrives at the SM and is registered by RFID readers upon entering the SM warehouse.
6. A confirmation is sent to the DC.

Interaction 2: Order/Shipment Handling between DC and SP

This interaction exercises persistent P2P and pub/sub (durable) messaging between the DCs and SPs. The interaction is triggered when goods in a DC are depleted and the DC has to order from a SP to refill stock. The following steps are followed as illustrated in Figure 2:

1. A DC sends a call for offers to all SPs that supply the types of goods that need to be ordered.
2. SPs that can deliver the goods send offers to the DC.
3. Based on the offers, the DC selects a SP and sends a purchase order to it.
4. The SP sends a confirmation to the DC and an invoice to the HQ. It then ships the ordered goods.
5. The shipment arrives at the DC and is registered by RFID readers upon entering the DC's warehouse.
6. The DC sends a delivery confirmation to the SP.
7. The DC sends transaction statistics to the HQ.

Interaction 3: Price Updates

This interaction exercises persistent, durable pub/sub messaging between the HQ and the SMs. The interaction is triggered when selling prices are changed by the company administration. To communicate this, the company HQ sends messages with pricing information to the SMs.

Interaction 4: SM Inventory Management

This interaction exercises persistent P2P messaging inside the SMs. The interaction is triggered when goods leave the warehouse of a SM (to refill a shelf). Goods are registered by RFID readers and the local warehouse application is notified so that inventory can be updated.

Interaction 5: Sales Statistics Collection

This interaction exercises non-persistent P2P messaging between the SMs and the HQ. The interaction is triggered when a SM sends sales statistics to the HQ. HQ can use this data as a basis for data mining in order to study customer behavior and provide useful information to marketing.

Interaction 6: New Product Announcements

This interaction exercises non-persistent, non-durable pub/sub messaging between the HQ and the SMs. The interaction is triggered when new products are announced by the company administration. To communicate this, the HQ sends messages with product information to the SMs selling the respective product types.

Interaction 7: Credit Card Hot Lists

This interaction exercises non-persistent, non-durable pub/sub messaging between the HQ and the SMs. The interaction is triggered when the HQ sends credit card hot lists to the SMs (complete list once every hour and incremental updates as required).

2.4 Benchmark Implementation

Event Handlers and Agents SPECjms2007 is implemented as a Java application comprising multiple JVMs and threads distributed across a set of *client nodes*. For every destination (queue or topic), there is a separate Java class called *Event Handler (EH)* that encapsulates the application logic executed to process messages sent to that destination. Event handlers register as listeners for the queue/topic and receive call backs from the messaging infrastructure as new messages arrive. For maximal performance and scalability, multiple instances of each event handler executed in separate threads can exist and they can be distributed over multiple physical nodes. Event handlers can be grouped according to the physical location (e.g. HQ, SM, DC or SP) they pertain to in the business scenario. In addition to the event handlers, for every physical location, a set of threads is launched to drive the benchmark interactions that are logically started at that location. These are called *driver threads*. The set of all event handlers and driver threads pertaining to a given physical location is referred to as *agent*. For example, each DC agent is comprised of a set of event handlers for the various destinations inside the DC and a set of driver threads used to drive Interaction 2, which is the only interaction with logical starting point at DCs.

Workload Configurability An important goal of SPECjms2007 that we discussed in Section 2.1 was to provide a flexible framework for performance analysis of MOM servers that allows users to configure and customize the workload according to their requirements. To achieve this goal, the interactions have been implemented in such a way that one could run them in different combinations depending on the desired transaction mix. SPECjms2007 offers three different ways of structuring the workload: horizontal, vertical and freeform. The latter are referred to as *workload topologies* and they correspond to three different modes of running the benchmark offering different level of configurability. The horizontal topology is meant to exercise the ability of the system to handle increasing number of destinations. To this end, the workload is scaled by increasing the number of physical locations (SMs, DCs, etc.) while keeping the traffic per location constant. The vertical topology, on the other hand, is meant to exercise the ability of the system to handle increasing message traffic through a fixed set of destinations. Therefore, a fixed set of physical locations is used and the workload is scaled by increasing the rate at which interactions are run. Finally, the freeform topology allows the user to use the seven SPECjms2007 interactions as building blocks to design his own workload scenario which can be scaled in

an arbitrary manner by increasing the number of physical locations and/or the rates at which interactions are run. The user can configure the number of physical locations emulated, the number of message producers and consumers, the message size disributions, the message delivery modes, etc. Most importantly, the user can selectively turn off interactions or change the rate at which they are run to shape the workload according to his requirements. At the same time, when running the horizontal or vertical topology, the benchmark behaves as if the interactions were interrelated according to their dependencies in the real-life application scenario. For further details on the benchmark implementation, the reader is referred to [13].

3 SPECjms2007 Workload Characterization

3.1 Message Traffic Analysis

We start with a detailed analysis of the message traffic produced by the benchmark workload in terms of the number and type of messages generated and their sizes. We consider the workload parameters that can be configured in the most general freeform topology and show how they affect the resulting message traffic. The different types of messages and destinations used in the various interactions are detailed in Table 1.

Messages Sizes The sizes of the messages generated as part of each interaction can be configured by setting an interaction-specific message sizing parameter (for example, "number of order lines sent to DC" for Interaction 1). Each sizing parameter can be assigned three possible values with respective probabilities (discrete probability distribution). The message sizing parameters used for the different interactions are listed in Table 2, along with some data that can be used to compute the resulting message sizes in KBytes. This data is based on measurements we took using a deployment of SPECjms2007 on a major JMS server platform[3]. The exact message sizes may be slightly different on different platforms, as MOM servers add their own platform-specific message headers. The measurements provided here were compared against measurements on a second popular JMS server and the differences were negligible. Based on the data in Table 2, the message sizes in KBytes for Interactions 1, 2, 4, 6 and 7 can be computed as $\vartheta = m_1 \cdot x + b$ where x is the interaction's message sizing parameter and m_1 and b are set to their respective values from Table 2. The `priceUpdate` messages of Interaction 3 have constant size that cannot be changed by the user. The size of the `statInfoSM` messages used in Interaction 5 is configured using two sizing parameters as follows $\vartheta = x \cdot (m_1 + m_2 \cdot y) + b$ where x and y are the two sizing parameters (i.e. "number of SM cash desks" and "number of sales lines") and m_1, m_2 and b are set to their respective values from Table 2. Based on the above two formulas and the data in Table 2, the user can configure the benchmark to use message sizes that match the user's own target workload.

[3] Due to product license restrictions, the specific configuration used cannot be disclosed.

Intr.	Message	Destination	Type	Prop.	Description
1	order	Queue (DC)	ObjectMsg	P, T	Order sent from SM to DC.
	orderConf	Queue (SM)	ObjectMsg	P, T	Order confirmation sent from DC to SM.
	shipDep	Queue (DC)	TextMsg	P, T	Shipment registered by RFID readers upon leaving DC.
	statInfo-OrderDC	Queue (HQ)	StreamMsg	NP, NT	Sales statistics sent from DC to HQ.
	shipInfo	Queue (SM)	TextMsg	P, T	Shipment from DC registered by RFID readers upon arrival at SM.
	shipConf	Queue (DC)	ObjectMsg	P, T	Shipment confirmation sent from SM to DC.
2	callForOffers	Topic (HQ)	TextMsg	P, T, D	Call for offers sent from DC to SPs (XML).
	offer	Queue (DC)	TextMsg	P, T	Offer sent from SP to DC (XML).
	pOrder	Queue (SP)	TextMsg	P, T	Order sent from DC to SP (XML).
	pOrderConf	Queue (DC)	TextMsg	P, T	Order confirmation sent from SP to DC (XML).
	invoice	Queue (HQ)	TextMsg	P, T	Order invoice sent from SP to HQ (XML).
	pShipInfo	Queue (DC)	TextMsg	P, T	Shipment from SP registered by RFID readers upon arrival at DC.
	pShipConf	Queue (SP)	TextMsg	P, T	Shipment confirmation sent from DC to SP (XML).
	statInfo-ShipDC	Queue (HQ)	StreamMsg	NP, NT	Purchase statistics sent from DC to HQ.
3	priceUpdate	Topic (HQ)	MapMsg	P, T, D	Price update sent from HQ to SMs.
4	inventoryInfo	Queue (SM)	TextMsg	P, T	Item movement registered by RFID readers in the warehouse of SM.
5	statInfoSM	Queue (HQ)	ObjectMsg	NP, NT	Sales statistics sent from SM to HQ.
6	product-Announcement	Topic (HQ)	StreamMsg	NP, NT, ND	New product announcements sent from HQ to SMs.
7	creditCardHL	Topic (HQ)	StreamMsg	NP, NT, ND	Credit card hotlist sent from HQ to SMs.

Table 1. Message Types Used in The Interactions - (N)P=(Non-)Persistent; (N)T=(Non-)Transactional; (N)D=(Non-)Durable

Message Throughput We now characterize the message throughput first on a per interaction basis and then on a per location basis. The two most important sets of workload parameters that determine the message throughput are the number of locations of each type and the interaction rates. We denote the sets of physical locations as follows:

$$\Psi_{SM} = \{SM_1, SM_2, \ldots, SM_{|\Psi_{SM}|}\} \qquad \Psi_{DC} = \{DC_1, DC_2, \ldots, DC_{|\Psi_{DC}|}\}$$
$$\Psi_{SP} = \{SP_1, SP_2, \ldots, SP_{|\Psi_{SP}|}\} \qquad \Psi_{HQ} = \{HQ_1, HQ_2, \ldots, HQ_{|\Psi_{HQ}|}\}$$

Note that although the modeled scenario has a single physical HQ location, the benchmark allows multiple HQ instances to exist each with its own set of queues. The goal is to avoid the HQ queues becoming a bottleneck when scaling the number of SMs, DCs and SPs. It is assumed that messages sent to the HQ are distributed evenly among the HQ instances. Multiple HQ instances are considered as separate servers within the same physical location.

For each interaction, the *interaction rate* specifies the rate at which the interaction is initiated by every physical instance of its initiating location, SM for Interaction 1, DC for Interaction 2, etc. We denote the interaction rates as $\lambda_i, 1 \leq i \leq 7$. Since multiple HQ instances are not considered as separate physical locations, it follows that the rates of Interactions 3, 6 and 7 which are initiated

Intr.	Message Sizing Parameters	Message	m_1	m_2	b
1	No of order lines sent to DC	orderConf	0.0565	*na*	1.7374
		statInfoOrderDC	0.0153	*na*	0.1463
		shipInfo	0.0787	*na*	0.8912
		shipDep	0.0787	*na*	0.7222
		order	0.0565	*na*	1.4534
		shipConf	0.0202	*na*	0.7140
2	No of purchase order lines sent to SP	callForOffers	0.1785	*na*	0.8094
		offer	0.2489	*na*	0.9414
		pOrder	0.2498	*na*	1.1076
		pShipConf	0.0827	*na*	0.7612
		statInfoShipDC	0.0831	*na*	0.7681
		pOrderConf	0.2410	*na*	1.3494
		invoice	0.1942	*na*	1.1211
		pShipInfo	0.0827	*na*	0.7279
3	*Message has fixed size*	priceUpdate	*na*	*na*	0.2310
4	No of registered items leaving warehouse	inventoryInfo	0.0970	*na*	0.5137
5	No of cash desks & sales lines	statInfoSM	0.0139	0.3650	0.9813
6	No of new products announced	productAnnouncement	0.0103	*na*	0.1754
7	No of credit cards in hot list	creditCardHL	0.0166	*na*	0.1846

Table 2. Parameters for Message Size Calculation

by the HQ are interpreted as rates over all HQ instances as opposed to rates per HQ instance. Interaction 2 uses a set of topics representing the different product families offered by suppliers. These topics help to distribute the `callForOffers` messages sent by DCs. Suppliers subscribe to all topics corresponding to groups of products they offer so that they receive all relevant `callForOffers` messages. We denote the set of product families as $\Pi = \{PF_1, PF_2, PF_3, \ldots, PF_{|\Pi|}\}$.

The probability that a SP offers products from a given product family $PF_i \in \Pi$ is a configurable workload parameter and will be denoted as ρ. Every SP subscribes to $\rho \cdot |\Pi|$ product families and thus $|\Psi_{SP}| \cdot \rho \cdot |\Pi|$ subscriptions exist overall. The number of subscribers that subscribe to a given product family is denoted as $\zeta = |\Psi_{SP}| \cdot \rho$.

Group	a	b	c	d
Type	Pub/Sub	Pub/Sub	P2P	P2P
Properties	NP NT ND	P T D	NP NT	P T

Table 3. Message Groups

In the following, we show how the message throughput, in terms of the number of messages sent and received per unit of time, can be broken down according to the type of messaging (P2P vs. pub/sub) and the message delivery mode (persistent vs. non-persistent, transactional vs. non-transactional, durable vs. non-durable). To this end, we group messages as shown in Table 3. Further, we define the following sets:

$\Gamma = \{a, b, c, d\}$: Message groups as defined in Table 3.
$\Omega = \{se, re\}$: Messages sent vs. messages received.
$\Lambda = \{SM, SP, DC, HQ\}$: Types of physical locations.

3.1a). Message Throughput per Interaction

We first analyze the message throughput on a per interaction basis. We will use the following notation:

$\xi^j_{i,k}$ for $j \in \Omega, 1 \leq i \leq 7$ and $k \in \Gamma$

No of messages of group k sent/received per sec as part of Interaction i.

$\xi_i^j = \sum_{k \in \Gamma} \xi_{i,k}^j$ for $1 \leq i \leq 7, j \in \Omega$
Total no of messages sent/received per sec as part of Interaction i.

$\xi^j = \sum_{i=1}^{7} \xi_i^j$ for $j \in \Omega$
Total no of messages sent/received per sec over all interactions.

Based on the information provided in the previous sections and analysis of the benchmark design, the following equations are derived characterizing the message throughput of each interaction:

Interaction 1: $\xi_{1,c}^{se} = \xi_{1,c}^{re} = \lambda_1 \cdot |\Psi_{SM}|$ $\quad \xi_{1,d}^{se} = \xi_{1,d}^{re} = 5 \cdot \lambda_1 \cdot |\Psi_{SM}|$
$\xi_{1,k}^{j} = 0, \ \forall k \in \{a, b\} \wedge j \in \Omega$

Interaction 2: $\xi_{2,a}^{j} = 0, \ \forall j \in \Omega$ $\quad \xi_{2,c}^{se} = \xi_{2,c}^{re} = \lambda_2 \cdot |\Psi_{DC}|$
$\xi_{2,b}^{se} = \lambda_2 \cdot |\Psi_{DC}$ $\quad \xi_{2,d}^{se} = \xi_{2,d}^{re} = (\zeta + 5) \cdot \lambda_2 \cdot |\Psi_{DC}|$
$\xi_{2,b}^{re} = \zeta \cdot \lambda_2 \cdot |\Psi_{DC}|$

Interaction 3: $\xi_{3,b}^{se} = \lambda_3$ $\quad \xi_{3,k}^{j} = 0, \ \forall k \in \Gamma, k \neq b \wedge j \in \Omega$
$\xi_{3,b}^{re} = \lambda_3 \cdot |\Psi_{SM}|$

Interaction 4: $\xi_{4,d}^{se} = \xi_{4,d}^{re} = \lambda_4 \cdot |\Psi_{SM}|$ $\quad \xi_{4,k}^{j} = 0, \ \forall k \in \Gamma, k \neq d \wedge j \in \Omega$

Interaction 5: $\xi_{5,d}^{se} = \xi_{5,d}^{re} = \lambda_5 \cdot |\Psi_{SM}|$ $\quad \xi_{5,k}^{j} = 0, \ \forall k \in \Gamma, k \neq d \wedge j \in \Omega$

Interaction 6: $\xi_{6,a}^{se} = \lambda_6$ $\quad \xi_{6,k}^{j} = 0, \ \forall k \in \Gamma, k \neq a \wedge j \in \Omega$
$\xi_{6,a}^{re} = \lambda_6 \cdot |\Psi_{SM}|$

Interaction 7: $\xi_{7,a}^{se} = \lambda_7$ $\quad \xi_{7,k}^{j} = 0, \ \forall k \in \Gamma, k \neq a \wedge j \in \Omega$
$\xi_{7,a}^{re} = \lambda_7 \cdot |\Psi_{SM}|$

3.1b). Message Throughput per Location

We now analyze the message throughput on a per location basis. The following notation will be used:

$\chi_{l,k}^{j}$ for $j \in \Omega, l \in \Lambda, k \in \Gamma$
No of messages of group k sent/received per sec by a location of type l.

$\chi_l^j = \sum_{k \in \Gamma} \xi_{l,k}^j$ for $j \in \Omega, l \in \Lambda$
Total no of messages sent/received per sec by a location of type l.

SMs participate in all interactions apart from Interaction 2. The following equations characterize the message throughput of each SM:

$$\chi^{se}_{SM,a} = \chi^{se}_{SM,b} = \chi^{re}_{SM,c} = 0 \qquad \chi^{se}_{SM,c} = \lambda_5$$
$$\chi^{re}_{SM,a} = \lambda_6 + \lambda_7 \qquad \chi^{se}_{SM,d} = 2\lambda_1 + \lambda_4$$
$$\chi^{re}_{SM,b} = \lambda_3 \qquad \chi^{re}_{SM,d} = 2\lambda_1 + \lambda_4$$

SPs participate only in Interaction 2. Overall $\lambda_2 \cdot |\Psi_{DC}|$ `callForOffers` messages are sent by the DCs per sec. Therefore, every SP receives $\rho \cdot \lambda_2 \cdot |\Psi_{DC}|$ messages and for each of them it sends an offer to the respective DC. The probability that an offer is accepted is $\frac{1}{\zeta}$ and hence the number of SP offers accepted per sec is given by:

$$\frac{\rho \cdot \lambda_2 \cdot |\Psi_{DC}|}{\zeta} = \frac{\lambda_2 \cdot |\Psi_{DC}|}{|\Psi_{SP}|}$$

The following equations characterize the message throughput of each SP:

$$\chi^{se}_{SP,a} = \chi^{re}_{SP,a} = \chi^{se}_{SP,b} = \chi^{se}_{SP,c} = \chi^{re}_{SP,c} = 0$$
$$\chi^{re}_{SP,b} = \rho \cdot \lambda_2 \cdot |\Psi_{DC}|$$
$$\chi^{se}_{SP,d} = \rho \cdot \lambda_2 \cdot |\Psi_{DC}| + \frac{3\lambda_2 \cdot |\Psi_{DC}|}{|\Psi_{SP}|}$$
$$\chi^{re}_{SP,d} = \frac{2\lambda_2 \cdot |\Psi_{DC}|}{|\Psi_{SP}|}$$

DCs participate in Interactions 1 and 2 both as producers and consumers of messages. The number of SMs supplied by each DC is given by $\delta = \frac{|\Psi_{SM}|}{|\Psi_{DC}|}$.

The following equations characterize the message throughput of each DC:

$$\chi^{se}_{DC,a} = \chi^{re}_{DC,a} = \chi^{re}_{DC,b} = \chi^{re}_{DC,c} = 0$$
$$\chi^{se}_{DC,b} = \lambda_2$$
$$\chi^{se}_{DC,c} = \delta \cdot \lambda_1 + \lambda_2$$
$$\chi^{se}_{DC,d} = 3\lambda_1 \cdot \delta + 2\lambda_2$$
$$\chi^{re}_{DC,d} = 3\lambda_1 \cdot \delta + \lambda_2(\zeta + 2)$$

The HQ participate in Interactions 1, 2, and 5 as message consumer and in Interactions 3, 6, and 7 as message producer. The following equations characterize the message throughput of the HQ:

$$\chi^{re}_{HQ,a} = \chi^{re}_{HQ,b} = \chi^{se}_{HQ,c} = \chi^{se}_{HQ,d} = 0$$
$$\chi^{se}_{HQ,a} = \lambda_6 + \lambda_7$$
$$\chi^{se}_{HQ,b} = \lambda_3$$
$$\chi^{re}_{HQ,c} = \lambda_1 \cdot |\Psi_{SM}| + \lambda_2 \cdot |\Psi_{DC}| + \lambda_5 \cdot |\Psi_{SM}|$$
$$\chi^{re}_{HQ,d} = \lambda_2 \cdot |\Psi_{DC}|$$

The detailed message throughput analysis presented above serves two main purposes. First, using the throughput equations, the user can assemble a workload configuration (in terms of number of locations and interaction rates) that

stresses specific types of messaging under given scaling conditions. As a very basic example, the user might be interested in evaluating the performance and scalability of non-persistent pub/sub messaging under increasing number of subscribers. In this case, a mix of Interactions 6 and 7 can be used with increasing number of SMs. Second, the characterization of the message traffic on a per location basis can help users to find optimal deployment topology of the agents representing the different locations such that the load is evenly distributed among client nodes and there are no client-side bottlenecks. This is especially important for a messaging benchmark where the server acts as mediator in interactions and significant amount of processing is executed on the client side.

3.2 Horizontal Topology

As mentioned earlier, the goal of the horizontal topology is to exercise the ability of the system to handle increasing number of destinations. To achieve this, the workload is scaled by increasing the number of physical locations (SMs, DCs, etc) while keeping the traffic per location constant. A scaling parameter `BASE` is introduced and the following rules are enforced:

1. $|\Psi_{SM}| = \texttt{BASE}$
2. $|\Psi_{DC}| = \lceil \frac{|\Psi_{SM}|}{5} \rceil$
3. $|\Psi_{SP}| = \lceil 0.4 \cdot |\Psi_{SM}| \rceil$
4. $|\Psi_{HQ}| = \lceil \frac{|\Psi_{SM}|}{20} \rceil$
5. $|\Pi| = |\Psi_{SM}|$
6. $\rho = \frac{5}{|\Pi|}$
7. λ_i, $1 \leq i \leq 7$ are fixed

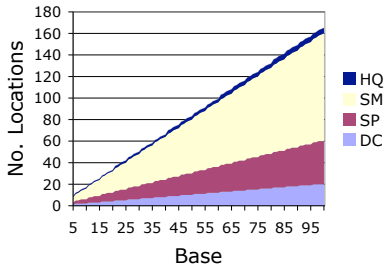

Fig. 3. # Locations for Horiz. Topology

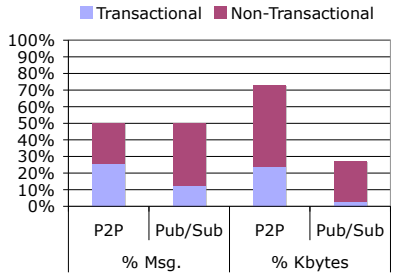

Fig. 4. Horiz. Topology Message Mix

Figure 3 shows how the number of locations of each type is scaled as the `BASE` parameter is increased. The rates λ_i at which interactions are initiated by participants are fixed so that the traffic per location (and therefore also per destination) remains constant. The relative weights of the interactions are set based on a detailed business model of the supermarket supply chain which captures the interaction interdependencies. This model has several input parameters (e.g. total number of product types, size of supermarkets, average number of items sold per week) whose values are chosen in such a way that the following overall target messaging mix is achieved as close as possible:

– 50% P2P messages and 50% pub/sub

– 50% of P2P messages persistent, 50% non-persistent
– 25% of pub/sub messages persistent, 75% non-persistent

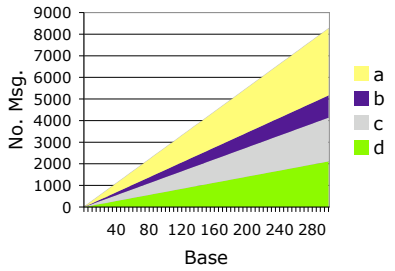

Fig. 5. Horizontal Topology: # msg.

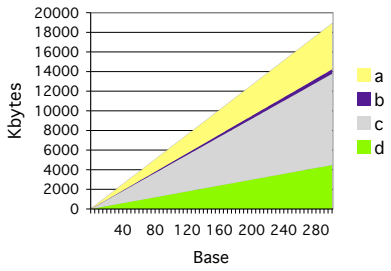

Fig. 6. Message traffic in Kbytes

The goal is to put equal weight on P2P and pub/sub messaging. Within each group the target relative weights of persistent vs. non-persistent messaging have been set according to the relative usage of these messaging styles in real-life applications. Table 4(a) shows the achieved message mix in the horizontal topology. Figure 4 presents the same data in graphical form. Figures 5 and 6 show how the number of messages of each type and the bandwidth they use are scaled as a function of the `BASE` parameter. As evident from the figure, when scaling the workload the proportions of the different types of messages remain constant. This is expected since the relative weights of the various messaging styles used by the workload should not depend on the scaling factor.

Intr.	**Message** *Probability*	**Size 1** *95 %*	**Size 2** *4 %*	**Size 3** *1 %*	**Avg. Size**
1	orderConf	2.02	7.39	41.29	2.63
	statInfoOrderDC	0.22	1.67	10.83	0.39
	shipInfo	1.28	8.76	55.95	2.13
	shipDep	1.12	8.59	55.79	1.96
	order	1.74	7.10	41.01	2.34
	shipConf	0.81	2.73	14.83	1.03
2	callForOffers	1.35	7.06	36.52	1.93
	offer	1.69	9.65	50.71	2.50
	pOrder	1.86	9.85	51.07	2.67
	pShipConf	1.01	3.65	17.29	1.28
	statInfoShipDC	1.02	3.68	17.38	1.29
	pOrderConf	2.07	9.79	49.56	2.86
	invoice	1.70	7.92	39.95	2.33
	pShipInfo	0.98	3.62	17.26	1.24
3	priceUpdate	0.24	0.24	0.24	0.24
4	inventoryInfo	1.48	10.22	49.03	2.31
5	statInfoSM	*na*			5.27
6	productAnnouncement	1.21	0.28	10.51	1.26
7	creditCardHL	1.01	8.49	50.00	1.80

Table 4. Message Sizes in KByte

The sizes of the messages used in the various interactions have been chosen to reflect typical message sizes in real-life MOM applications. Pub/sub messages are generally much smaller than P2P messages due to the decoupled nature

of the delivery mechanism. For every type of message, SPECjms2007 generates messages with sizes chosen from a discrete distribution with three possible values as shown in Table 4. There are two exceptions, the `priceUpdate` message used in Interaction 4 and the `statInfoSM` message used in Interaction 5. The former has a fixed size, while the latter has size between 4.7 and 24.78 KB with an average of 5.27 KB. Since `statInfoSM` messages contain sales statistics, their size is determined by the rate at which items are sold in supermarkets which depends on the number of customers visiting a supermarket per day and the average number of items sold per customer.

(a) Horizontal

Message Group	Message Count Target	Achieved	Bandwidth Used
a	37.50%	37.46%	24.66%
b	12.50%	12.45%	2.41%
c	25.00%	24.55%	49.19%
d	25.00%	25.55%	23.74%

(b) Vertical

Message Group	Message Count Target	Achieved	Bandwidth Used
a	15.00%	14.19%	7.19%
b	5.00%	5.99%	2.25%
c	40.00%	39.09%	61.03%
d	40.00%	40.74%	29.52%

Table 5. Topology Message Mix

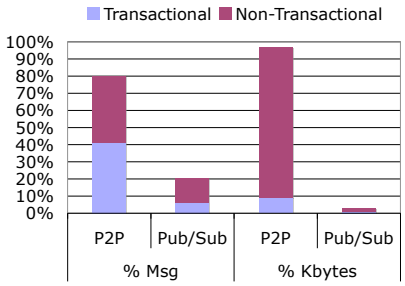

Fig. 7. Vert. Topology Message Mix

3.3 Vertical Topology

The goal of the vertical topology is to exercise the ability of the system to handle increasing message traffic through a fixed set of destinations. Therefore, a fixed set of physical locations is used and the workload is scaled by increasing the rate at which interactions are executed. Similar to the horizontal case, a single parameter `BASE` is used as a scaling factor. The following rules are enforced:

1. $|\Psi_{SM}| = 10$
2. $|\Psi_{DC}| = 2$
3. $|\Psi_{SP}| = 5$
4. $|\Psi_{HQ}| = 1$
5. $|\Pi| = 100$
6. $\rho = 50\%$
7. $\lambda_i = c_i \cdot$ `BASE`, where c_i is a fixed factor and $1 \leq i \leq 7$

Again, the relative weights of the interactions are set based on the business model of the supply chain scenario. Unlike the horizontal topology, however, the vertical topology places the emphasis on P2P messaging which accounts for 80% of the total message traffic. The aim is to exercise the ability of the system to handle increasing traffic through a destination by processing messages in parallel. This aspect of MOM server performance is more relevant for P2P messaging (queues) than for pub/sub messaging where the message throughput is inherently limited by the speed at which subscribers can process incoming messages.

Table 4(b) shows the achieved message mix in the vertical topology. Figure 7 presents the same data in graphical form. Figures 8 and 9 shows how the number

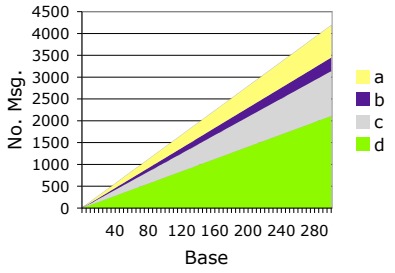

Fig. 8. Vertical Topology: # msg.

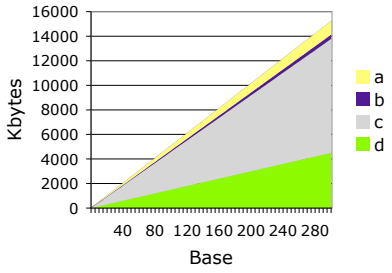

Fig. 9. Message traffic in Kbytes

of messages of each type and the bandwidth they use are scaled as a function of the `BASE` parameter. Again, when scaling the workload the message mix remains constant which is the expected behavior. The sizes of the messages used in the various interactions are computed in the same way as for the horizontal topology (see Table 4).

4 Concluding Remarks

We presented a comprehensive workload characterization of the new SPECjms2007 benchmark which is the world's first industry standard benchmark specialized for MOM. SPECjms2007 provides a flexible and robust tool that can be used for in-depth performance evaluation of MOM servers. However, in order to take advantage of this, users need to understand the way the workload is decomposed into components and which performance aspects are exercised by these components. The workload characterization presented in this paper is meant to help users gain an in-depth understanding of the SPECjms2007 workload and how it can be configured and customized. Our extensive analysis of the message traffic produced by the benchmark considered the following dimensions, i) message types and destinations, ii) message sizes, iii) message throughput and iv) message delivery modes. We characterized the message traffic both on a per interaction and location basis. The results we presented can be used to define a workload configuration that stresses selected features of the MOM infrastructure in a way that resembles a given target customer workload. Moreover, the traffic equations are essential for finding an optimal deployment topology with a uniform load distribution and no client-side bottlenecks. After considering the general freeform topology, we looked at the more specific horizontal and vertical topologies. We discussed their goals and characterized the interaction and message mixes they are based on and the way they are scaled. Our analysis not only helps to better understand and interpret official benchmark results, but also provides an example of how to define a scalable workload configuration for evaluating selected performance and scalability aspects of MOM.

5 Acknowledgments

This work was partially funded by the German Research Foundation. We acknowledge the contributions of the members of the SPECjms Working Group to the specification and development of SPECjms2007, in particular Marc Carter and Tim Dunn from IBM, George Tharakan from Sun Microsystems, Tom Barnes and Russell Raymundo from BEA, Evan Ireland from Sybase, and Adrian Co from Apache. We are also especially thankful to Lawrence Cullen, Robert Berry, Alan Adamson and John Stecher from IBM, Steve Realmuto from BEA and Ricardo Morin from Intel for their continued support of the SPECjms project.

References

1. ActiveMQ. JMeter performance test. http://incubator.apache.org/activemq/jmeter-performance-tests.html, 2006.
2. K. Alexander, T. Gillian, K. Gramling, M. Kindy, D. Moogimane, M. Schultz, and M. Woods. IBM Business Consulting Services - Focus on the Supply Chain: Applying Auto-ID within the Distribution Center. White paper IBM-AUTOID-BC-002, 2003.
3. C. Bornhövd, T. Lin, S. Haller, and J. Schaper. Integrating Automatic Data Acquisition with Business Processes - Experiences with SAP's Auto-ID Infrastructure. In *Proceedings of VLDB'04*, 2004.
4. M. Carter. JMS Performance with WebSphere MQ for Windows V6.0. http://www-1.ibm.com/support/docview.wss?rs=171& uid=swg24010028, 2005.
5. Crimson Consulting Group. High-Performance JMS Messaging - A Benchmark Comparison of Sun Java System Message Queue and IBM WebSphere MQ, 2003.
6. K. Finkenzeller. *RFID Handbook : Fundamentals and Applications in Contactless Smart Cards and Identification*. John Wiley & Sons, 2nd edition, may 2003.
7. IBM Hursley. Performance Harness for Java Message Service. http://www.alphaworks.ibm.com/tech/perfharness, 2005.
8. JBoss. JBoss JMS New Performance Benchmark. http://wiki.jboss.org/wiki/Wiki.jsp?page=JBossJMSNewPerformanceBenchmark, 2006.
9. S. Kounev. *Performance Engineering of Distributed Component-Based Systems - Benchmarking, Modeling and Performance Prediction*. Shaker Verlag, Dec. 2005. ISBN: 3832247130.
10. S. Kounev and A. Buchmann. Improving Data Access of J2EE Applications by Exploiting Asynchronous Processing and Caching Services. In *Proceedings of VLDB'02*, 2002.
11. Krissoft Solutions. JMS Performance Comparison. http://www.fiorano.com/comp-analysis/jms_perf_report.htm, 2006.
12. K. Sachs. Evaluation of Performance Aspects of the SAP Auto-ID Infrastructure. Master's thesis, Department of Computer Science, Darmstadt University of Technology, 2004.
13. K. Sachs, S. Kounev, M. Carter, and A. Buchmann. Designing a Workload Scenario for Benchmarking Message-Oriented Middleware. In *Proceedings of the 2007 SPEC Benchmark Workshop*. SPEC, January 2007.
14. Sonic Software Corporation. SonicMQ Test Harness, 2005.
15. Sun Microsystems Inc. Java Message Service (JMS) Specification Version 1.1. http://java.sun.com/products/jms/docs.html, 2002.