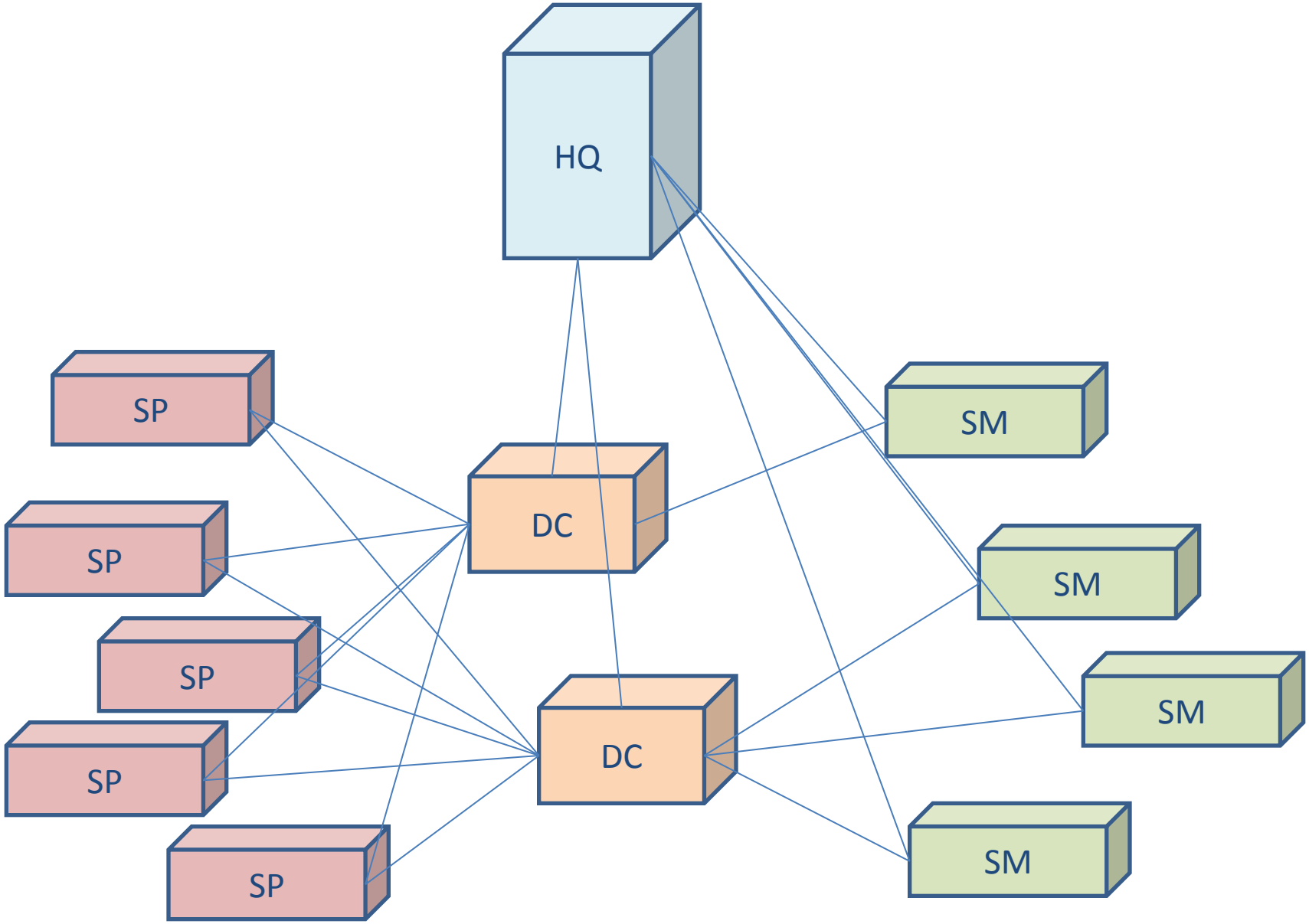
Lecture 15

Use case of message-oriented middleware

Supply chain of a supermarket company

- HQ company headquarters – manages the accounting of the company, manages information about the goods and products offered in the supermarket stores, manages selling prices and monitors the flow of goods and money in the supply chain.
- DC distribution centres - supply the supermarket stores. Every distribution centre is responsible for a set of stores in a given area. The distribution centres in turn are supplied by external suppliers. They take orders from supermarkets, order goods from suppliers, deliver goods to supermarkets and provide sales statistics to the HQ (e.g. for data mining).
- SM supermarkets sell goods to end customers. The scenario focuses on the management of the inventory of supermarkets including their warehouses.
- SP suppliers deliver goods to the distribution centres of the supermarket company.


HQ
SP
SP
SP
SP
SP
DC
DC
SM
SM
SM
SM

Data flow 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 :
  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.

Data flow 2: Order/Shipment Handling between DC and SP

- This interaction consists of 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 executed:
  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.

Data flow 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.

Data flow 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.

Data flow 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 behaviour and provide useful information to marketing.

Data flow 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.

Event handlers and agents

- 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.
- The set of all event handlers belonging to a given physical location is called *agent*.

Goal

- Configure/design the system so that all performance requirements are met irrespective of the workload variations.

- Experimental models:
  - 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.

Workload parameters

- # physical locations (HQs, SMs, DCs, SPs);
- rates at which data flow (messages) occur;
- message size distribution for each message type;
- # agents for each physical location;
- distribution of agents across client nodes;
- # JVMs run on each client node;
- distribution of agents among JVMs;
- # event handlers for each message type;
- # JMS Connections shared amongst event handlers;
- acknowledgment mode for non-transactional sessions;
- optional connection sharing by multiple sessions.

Data flow 1

Message.	Destination	Type	Prop	Description
order	Queue(DC)	ObjMsg	P,T	Order sent from SM to DC.
orderConf	Queue(SM)	ObjMsg	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)	ObjMsg	P,T	Shipment confirmation sent from SM to DC.

Analysis

- Step 1: determine messages size (KB)
- Step 2: message throughput, first on a per data flow 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 data flow rates.
- The detailed message throughput analysis serves two main purposes. First, using the throughput equations, the user can assemble a workload configuration (in terms of number of locations and data flow 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 data flow 6 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.

Suggested exercises

1. Consider the type of message destinations (queue or topic) for all data flows in the previous use case.

2. How would a queue or topic be physically implemented?

3. What additional mechanisms would be required for providing persistency and durability?

Reference

- The use case was taken from “Performance Evaluation of Message-oriented Middleware using the SPECjms2007 Benchmark” by K. Sachs, S. Kounev, J. Bacon and A. Buchmann, published in *Performance Evaluation* 66 (2009) 410-434, Elsevier.