Designing a Workload Scenario for Benchmarking Message-Oriented Middleware

Kai Sachs*,
Samuel Kounev*†,
Marc Carter ‡,
Alejandro Buchmann*

**** Databases and Distributed Systems Group, TU Darmstadt / Germany***
† Computer Laboratory, University of Cambridge / UK
‡ IBM Hursley Labs, Hursley Park, Winchester / UK

TECHNISCHE UNIVERSITÄT DARMSTADT

DVS

Overview

I. Introduction
II. Workload Requirements and goals of the SPECjms benchmark
III. Application Scenario for SPECjms
IV. Implementation Details
V. Summary

Message Oriented Middleware (MOM)

- Used in many business domains
  - Financial services and enterprise applications
  - Health care
  - Supply chain
  - ...
- And in many technologies
  - Enterprise Service Bus (ESB)
  - Service Oriented Architecture (SOA)
  - Enterprise Application Integration (EAI)
  - ...
- Increasing importance → **Need for benchmark**

Requirements of a MOM benchmark

- Scenario **representative** of real-world applications.
- Exercise **all critical** services provided by platforms.
- **Not** optimized for **a specific product**.
- **Reproducible results**.
- No inherent **scalability** limitations.

Current State of MOM Benchmarking

- **Many proprietary benchmarks** for MOM servers
  - Used for performance testing and product comparisons

However:

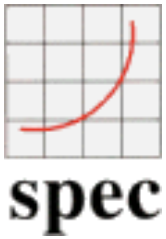
- These benchmarks do not meet all of the defined requirements

  Typically they...
  - concentrate on stressing individual MOM features, and
  - do not provide a comprehensive and representative workload for evaluating the overall MOM performance
- Currently no **industry-standard benchmark** for MOM Benchmarking → **SPECjms 2007**

What is SPECjms 2007?

- **World's first industry standard benchmark for MOM** products supporting Java Message Service (JMS)

- Developed by the SPEC OSG-Java subcommittee with the participation of:
  - IBM
  - TU Darmstadt
  - Sun
  - Sybase
  - BEA
  - Apache
  - Oracle
  - JBoss



Goals of SPECjms 2007

I. Provide a **standard workload and metrics** for measuring and evaluating JMS-based platforms

II. Provide a **flexible framework** for JMS performance analysis

Overview

I. Introduction
II. Workload Requirements and goals of the SPECjms benchmark
III. Application Scenario for SPECjms
IV. Implementation Details
V. Summary

Categories of Workload Requirements

- Representativeness
- Comprehensiveness
- Focus
- Scalability
- Configurability

Categories of Workload Requirements

- **Representativeness**
- Comprehensiveness
- Focus
- **Scalability**
- **Configurability**

Representativeness

The goal:

- Allow users to relate the observed behavior to their own applications and environments.
- Should simulate the way platform services are exercised in real-life systems.

Therefore:

- It should be based on a representative workload scenario:
  - Communication style and the types of messages should represent a *typical transaction mix*.

Scalability

- Dimensions of scaling the workload :
  - **Horizontal scaling**:
    - De/Increase the number of destinations (queues and topics)
    - Keep the traffic per destination constant
  - **Vertical scaling:**
    - De/Increase traffic per destination
    - Keep the number of destinations fixed
- Preserve real-life relationships in modeled scenario

- **Additionally:** Support for freeform scaling,
  e.g. user defined traffic per destination and number of destinations

Configurability I

- Provide a flexible performance analysis tool:
  - Allows users to configure and customize the workload, e.g. for research purposes
- Produce and publish standard results e.g for marketing purposes

Therefore:

- Need for a framework which supports
  - tuning,
  - analyzing and
  - optimizing

  performance of certain features / platforms

Configurability II

- A benchmark framework should allow:
  - precise **configuration of workload and transaction mix**
  - to **switch off business interactions**
    (implies that interactions should be decoupled)

- Providing such a configurability is a great challenge:
  - ***Freeform mode:***
    Design and implement interactions so that they can be run in different combinations depending on the desired transaction mix
  - ***Standard mode:***
    It has to be ensured, that the interactions always behave like defined in the application scenario

Overview

I. Introduction
II. Workload Requirements and goals of the SPECjms benchmark
III. Application Scenario for SPECjms
IV. Implementation Framework
V. Summary

The Application Scenario

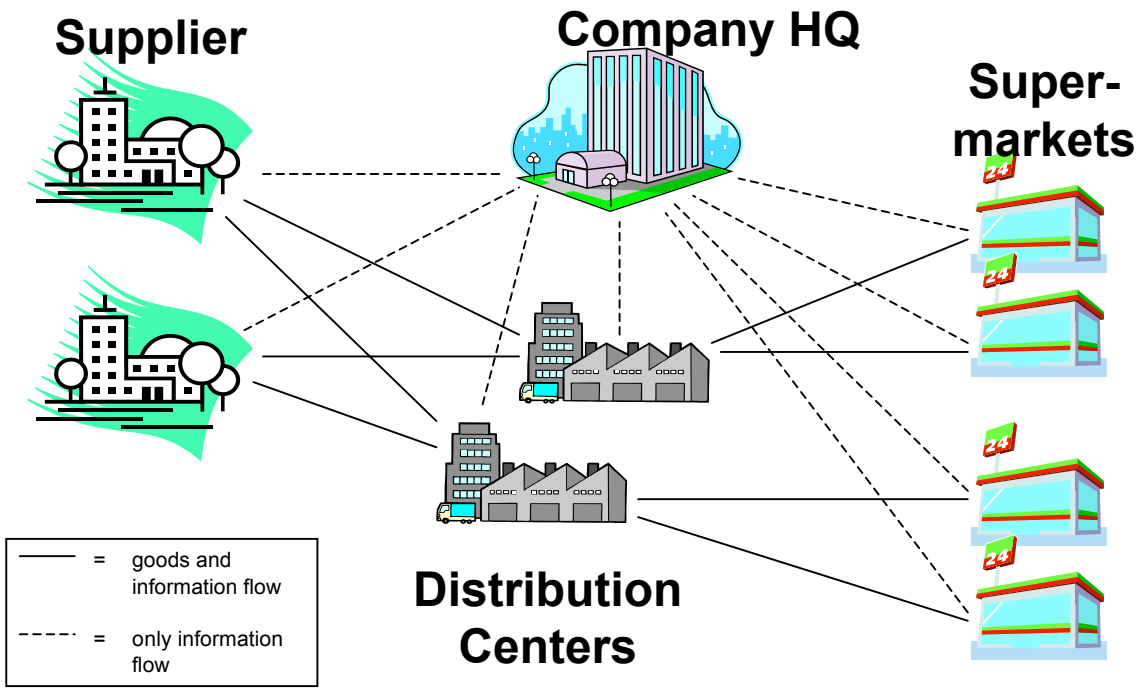
- Represents a **supply chain** of a supermarket company.
- **Participants:**
  - Headquarters (HQ)
  - Supermarkets (SM)
  - Distribution Centers (DC)
  - Suppliers (SP).
- Based on the previously discussed requirements.

The Application Scenario

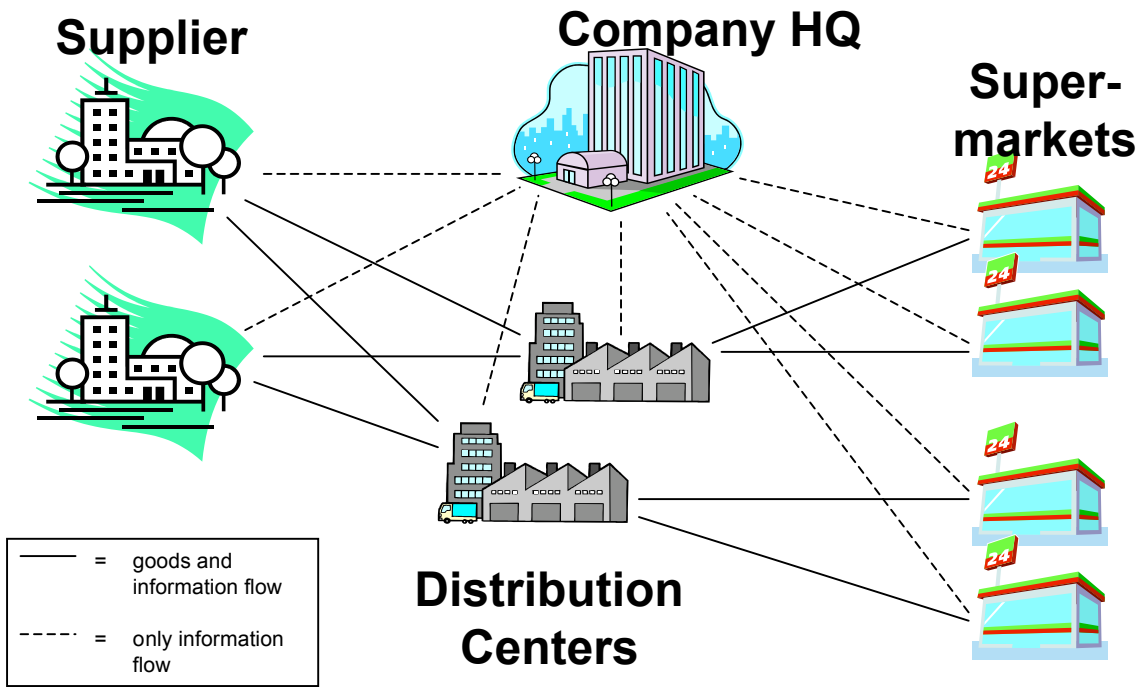
Why again a Supply Chain Scenario?

- Excellent basis for defining different interactions:
  Many destinations, use cases, ...
- Typical real word application
- Importance of performance (RFID!)
- Allows scaling the workload in a natural way:
  - *Horizontal:* e.g. scale the number of SMs
  - *Vertical:* e.g. scale amount of products sold per SM

Participants



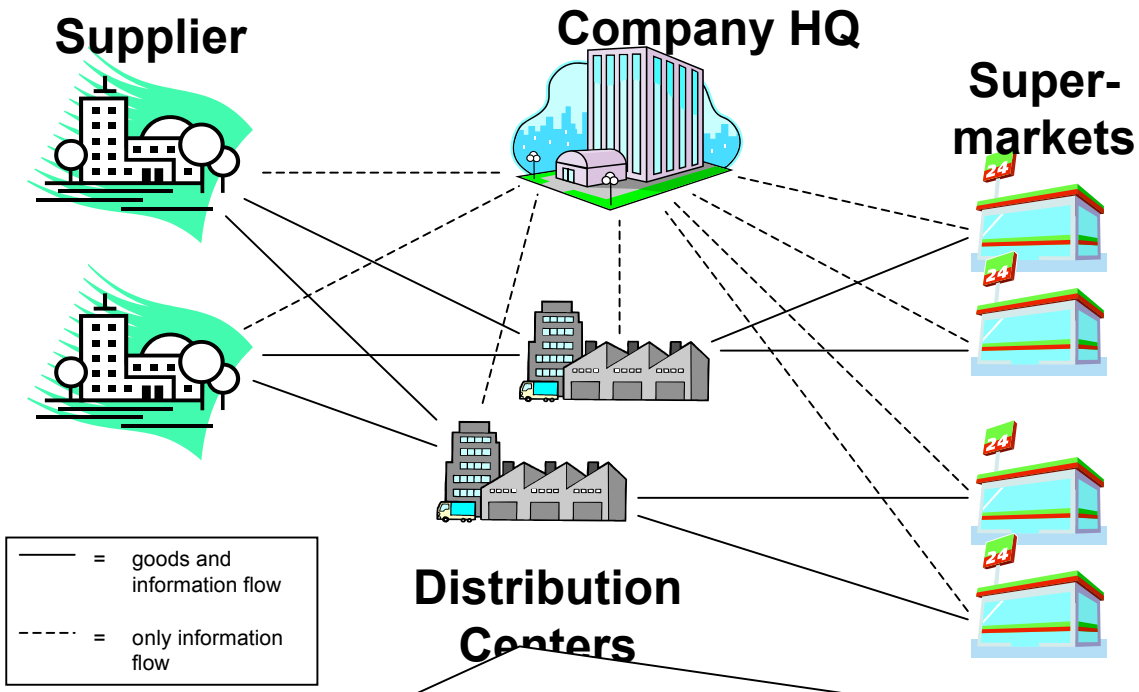
Participants - Supermarkets



Supermarket (SM)

- sells goods to end customers.
- manages its inventory.
- every supermarket offers different products.
- every supermarket is supplied by exactly one of the distribution centers.

Participants - Distribution Center



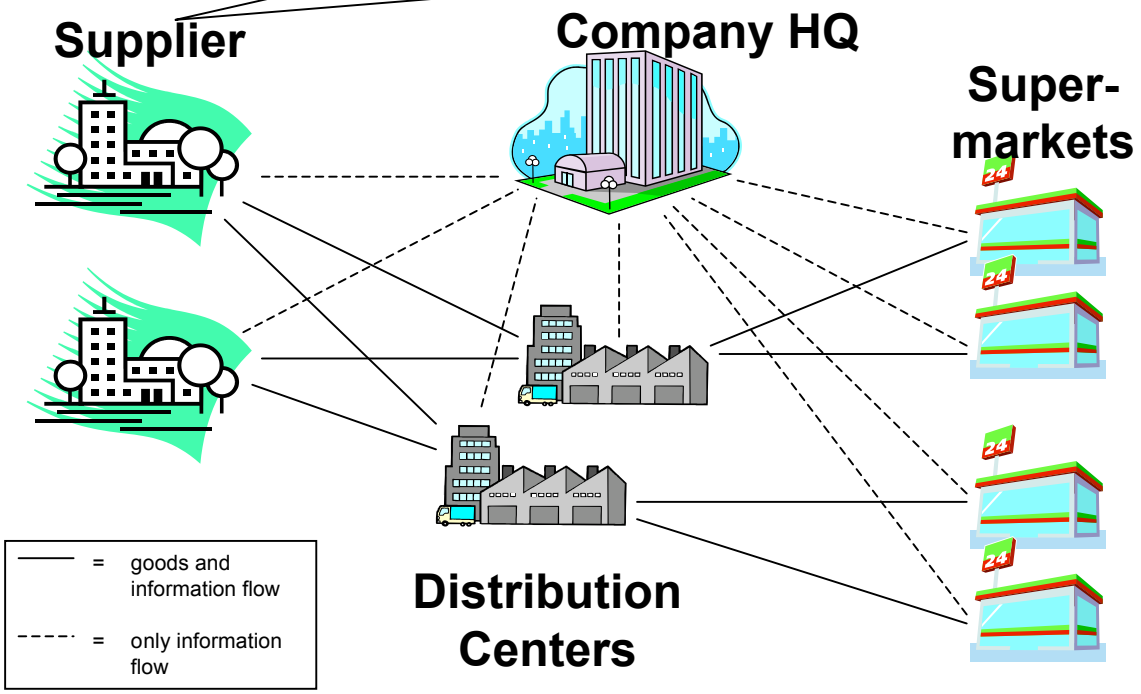
Distribution Center (DC)

- supplies the supermarket stores which sell goods to end customers.
- responsible for a set of stores in a given area.
- is supplied by external suppliers.

Participants – Suppliers

Supplier (SP)

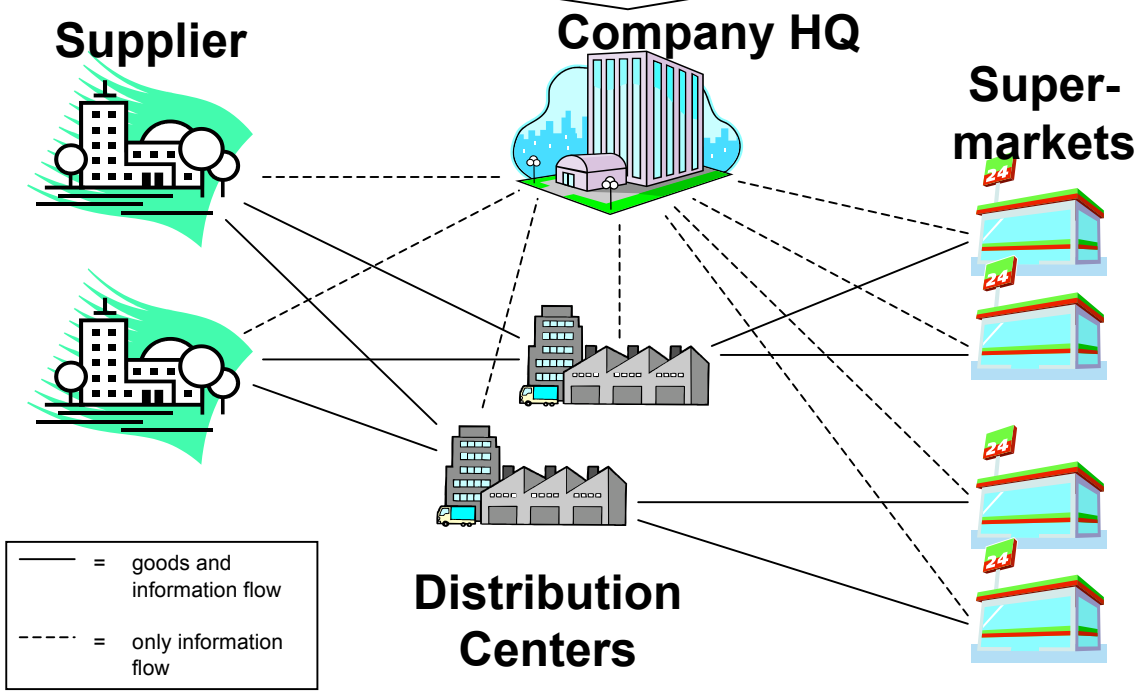
- deliver goods to distribution centers (based on an offer of the supplier).
- not every supplier offers the same products.
- offers either all products of a given product family or none of them.



Pa

Company HQ

- manages the accounting of the company.
- manages information about the goods and products.
- manages selling prices.
- monitors the flow of goods and money in the supply chain.



Business Interactions

The following interactions are part of the scenario:

1. Order / Shipment Handling (SM / DC)
2. (Purchase) Order / Shipment Handling (DC / SP)
3. Price Updates
4. Inventory Management
5. Sales Statistics Collection
6. Product Announcements
7. Credit Card Hotlists

Business Interactions

The following interactions are part of the scenario:

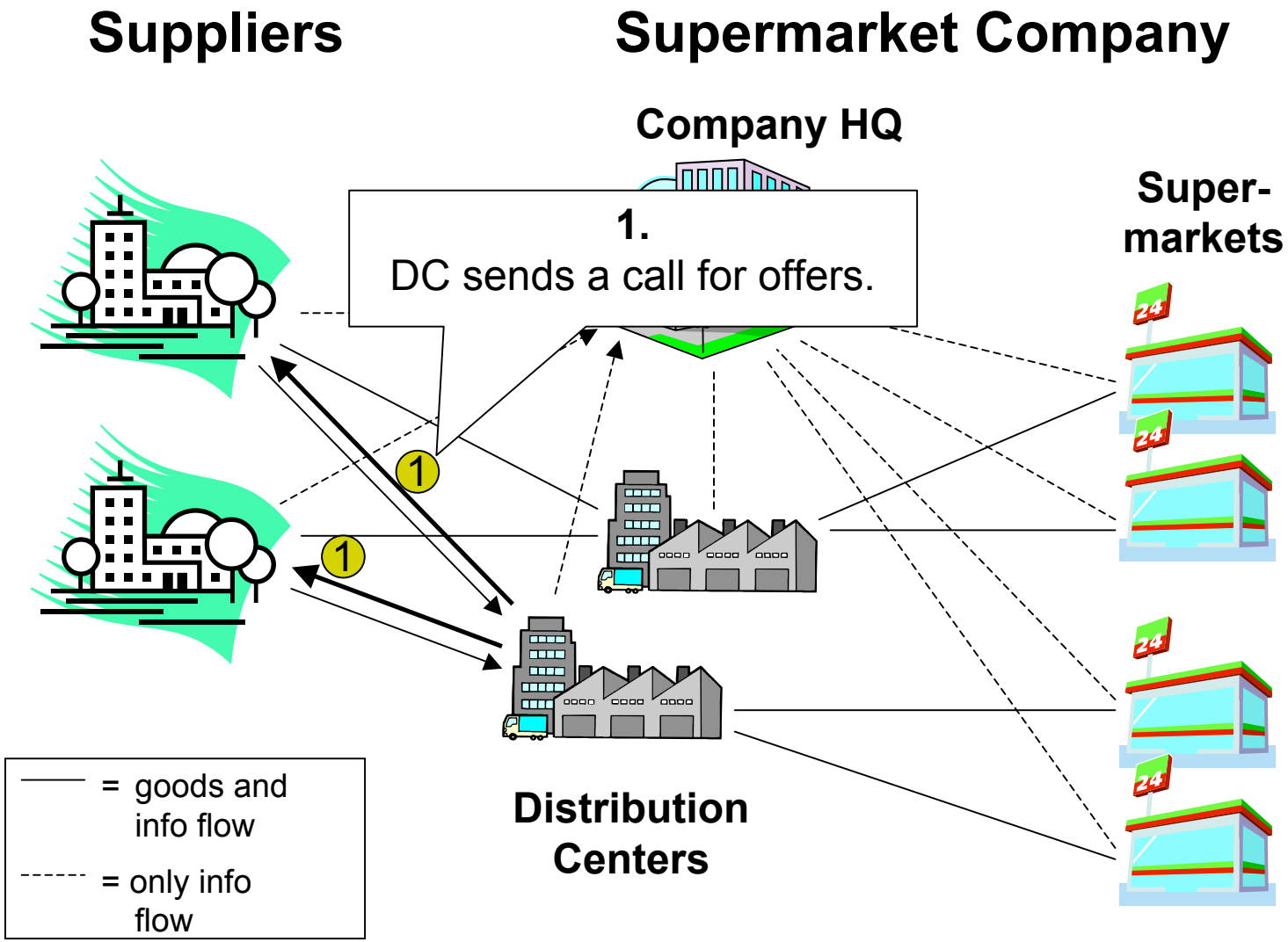
1. Order / Shipment Handling (SM / DC)
2. **(Purchase) Order / Shipment Handling (DC / SP)**
3. Price Updates
4. Inventory Management
5. Sales Statistics Collection
6. Product Announcements
7. Credit Card Hotlists

Example: Interaction 2

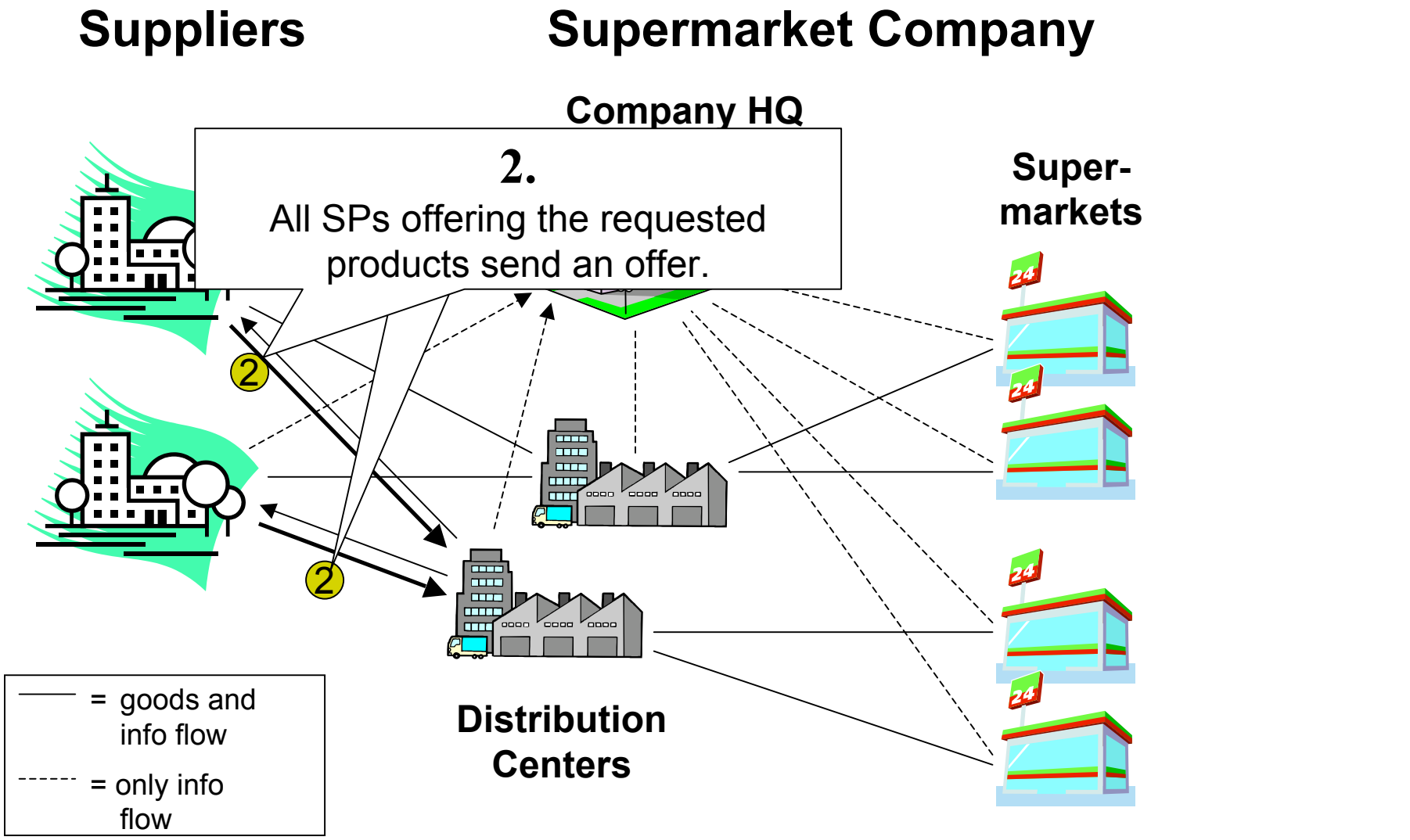
Purchase Order / Shipment Handling (DC & SPs)

- Point-to-Point **and** Publish/Subscribe communication.
- Inter company communication.
- Includes **six steps**

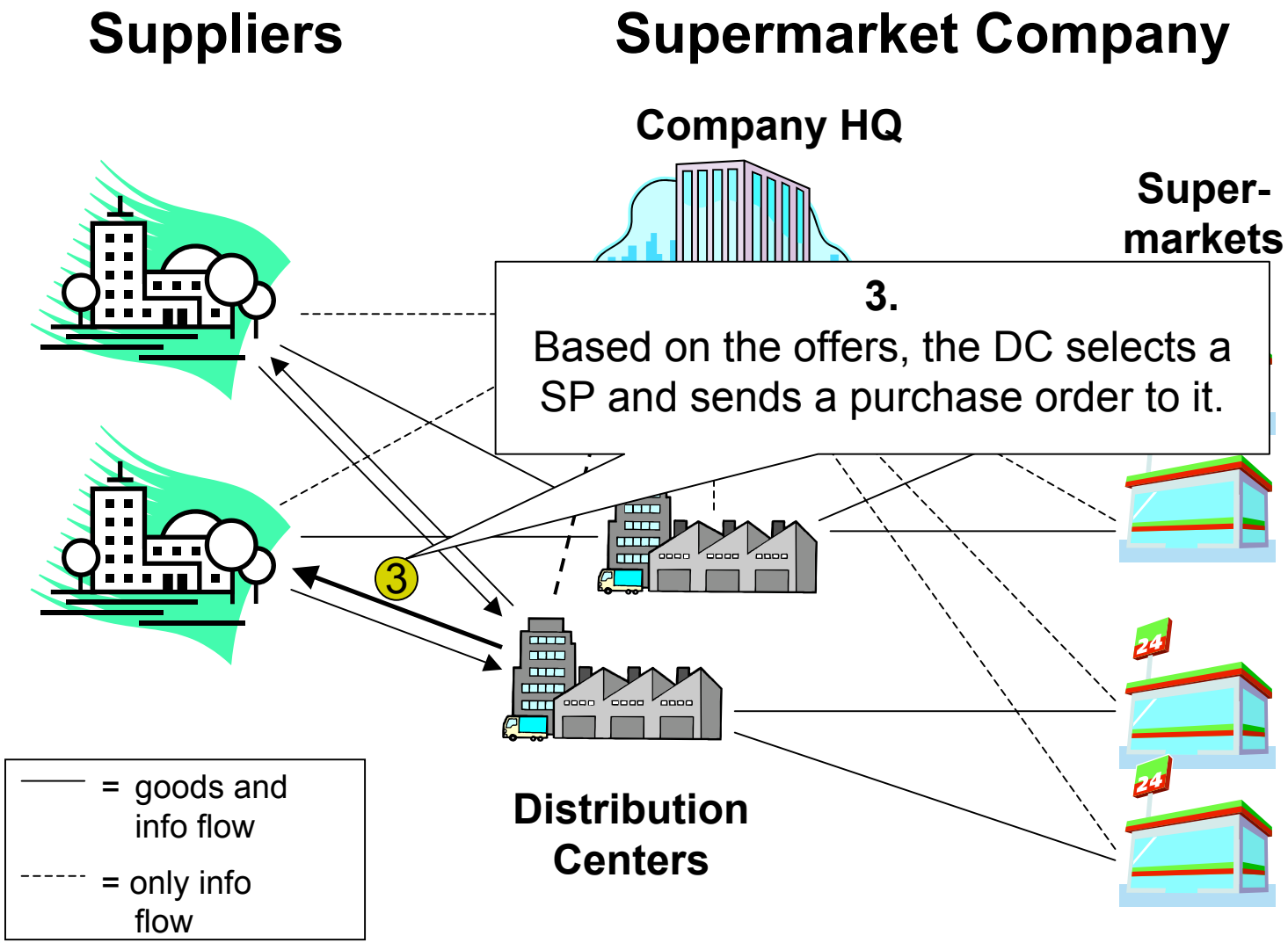
Interaction 2
Purchase Order / Shipment Handling



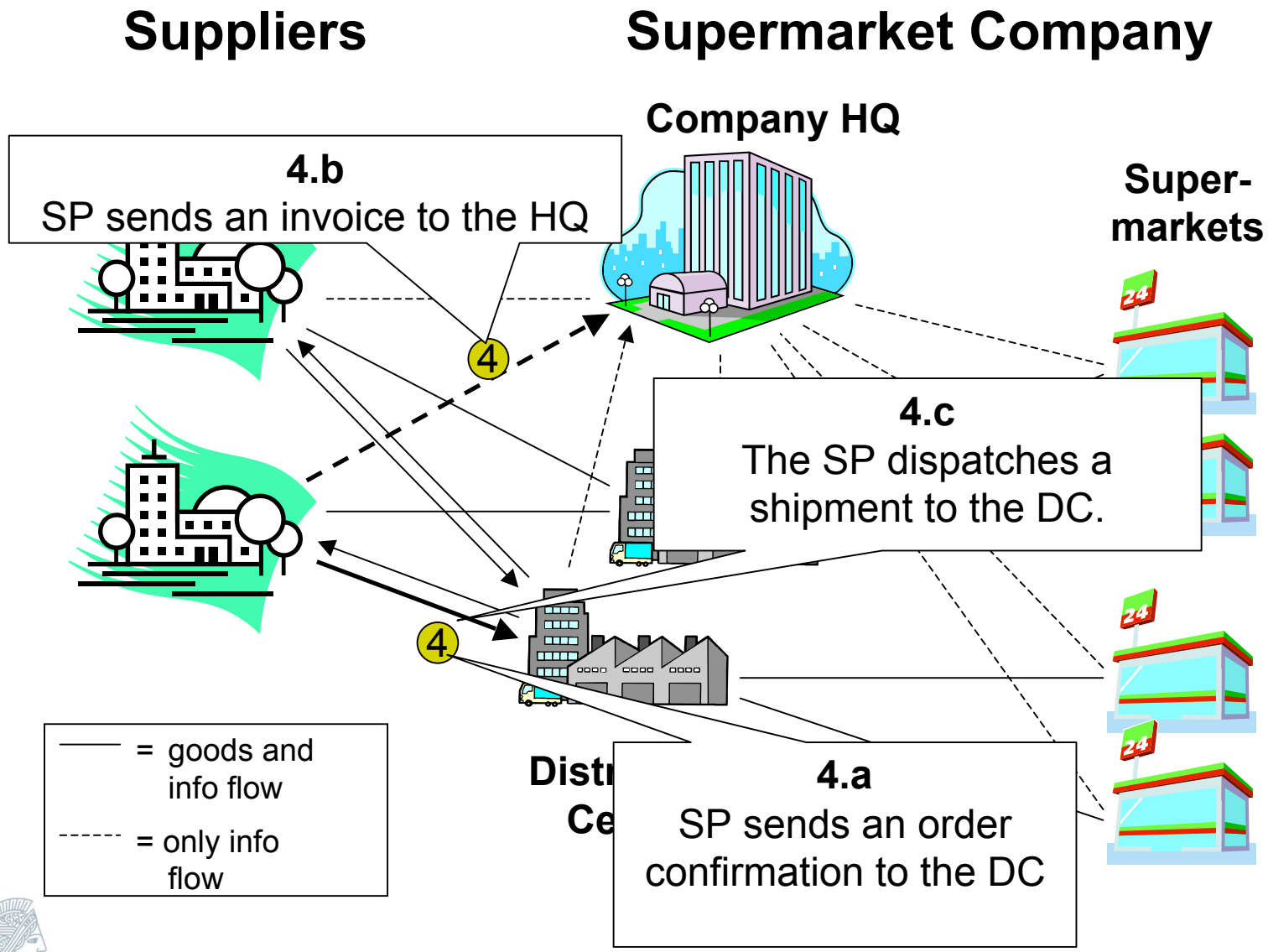
Interaction 2
Purchase Order / Shipment Handling



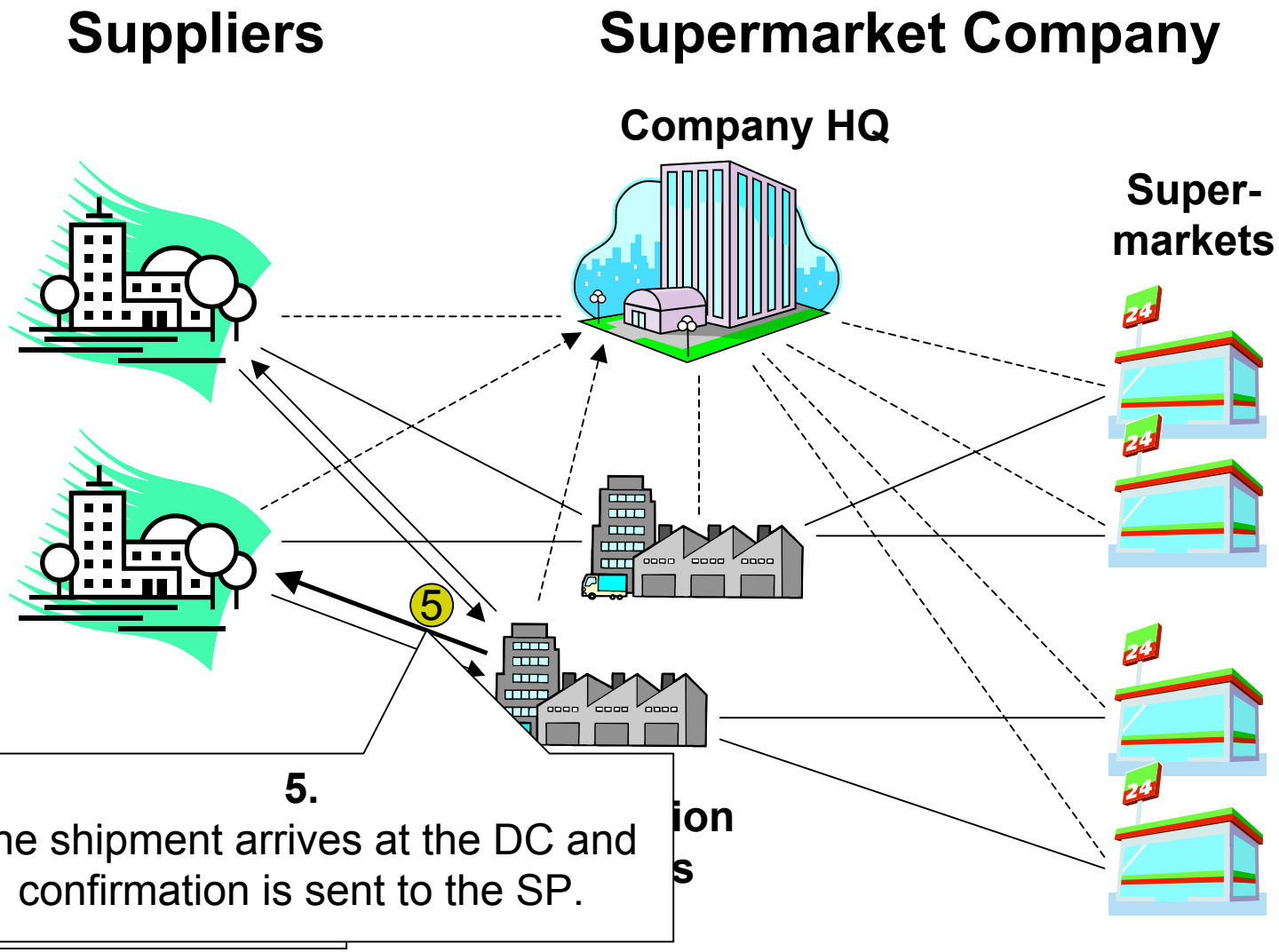
Interaction 2
Purchase Order / Shipment Handling



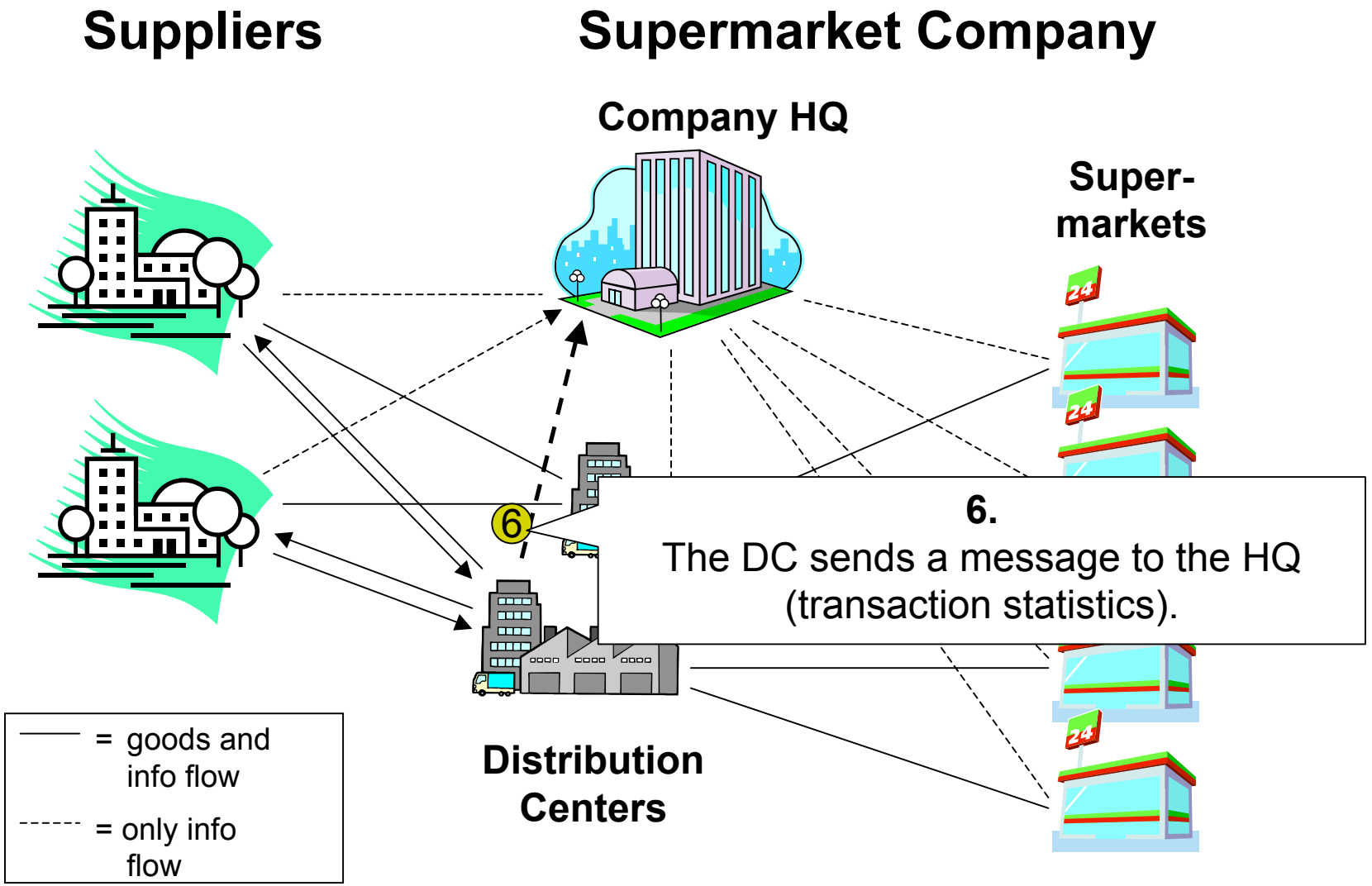
Interaction 2
Purchase Order / Shipment Handling



Interaction 2
Purchase Order / Shipment Handling



Interaction 2
Purchase Order / Shipment Handling



Overview

I. Introduction
II. Workload Requirements and goals of the SPECjms benchmark
III. Application Scenario for SPECjms
IV. Implementation Details
V. Summary

Message Types and Destinations

19 different messages are defined:

- Three different sizes per message (small, medium, large) with a certain probability
- Acknowledgment mode:
  - Standard: AUTO_ACKNOWLEDGMENT
    (can be changed in several interactions)
- All messages types supported by the JMS Specification excepted ByteMessages
- (Non-)Persistent, (Non-)Transactional, Durable, ...

Message Types and Destinations

- Number of queues per location instance:

Location	No. of queues
SM	3
SP	2
HQ	4
DC	6

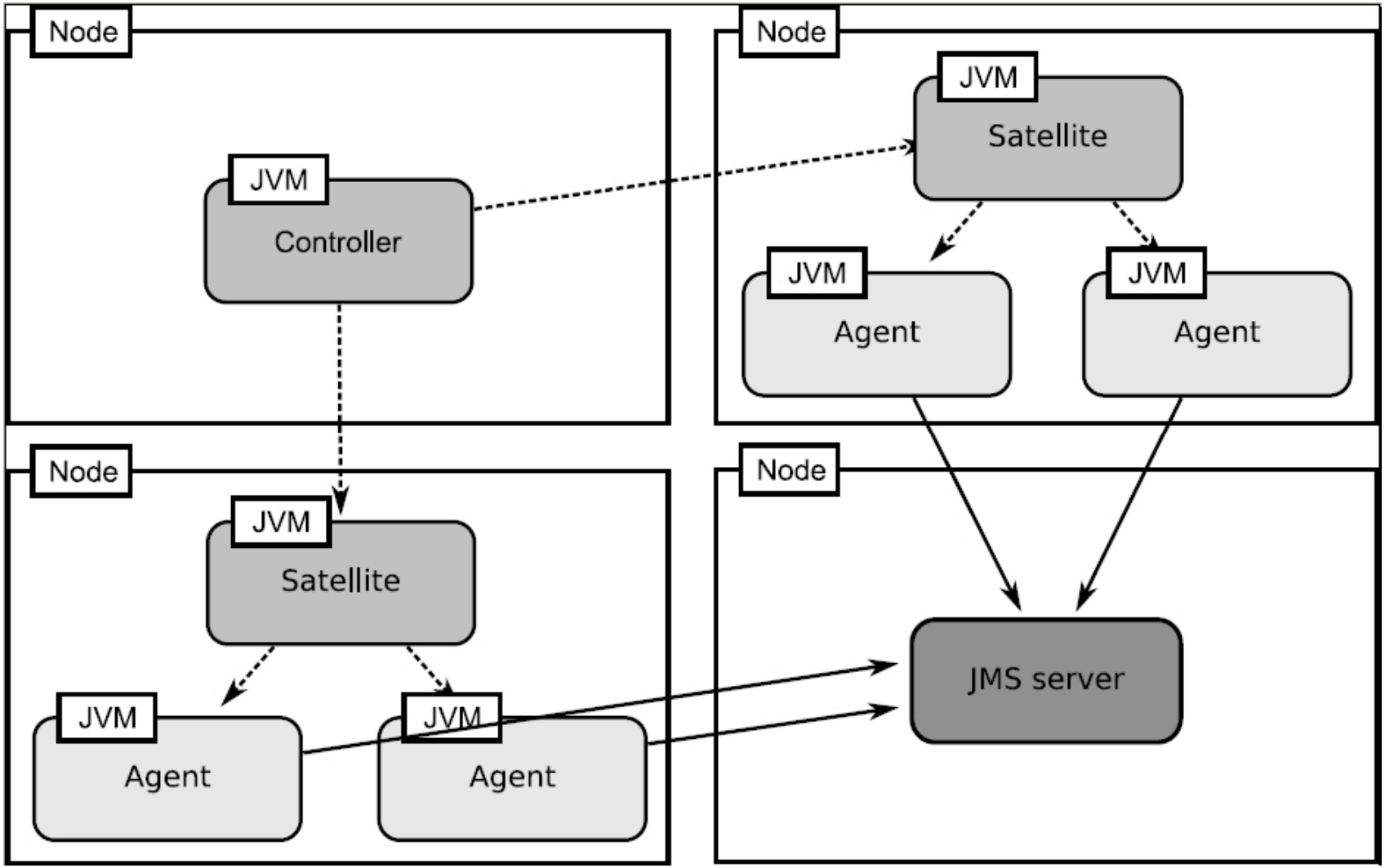
- Number of topics:
  3 + *one for every product family*

Driver Framework

- Many locations represented by many event handlers (message consumers)
- Event handlers may be distributed across many physical machines.

- **Reusable driver framework** addresses this issues without any inherent scalability limitations.
- **Plain Java**
- **Maximum choice** in laying out workload to achieve maximum performance.

Driver Framework



A Flexible Framework for Performance Analysis

- Allows to **configure and customize** the workload / transaction mixes
- Provides **three different topologies** corresponding to three different modes in which the benchmark can be run:
  - Vertical
  - Horizontal
  - Freeform
- **Many features**

A Flexible Framework for Performance Analysis

Some features:

- Number of physical locations (HQ, SM, DC, SP) emulated.
- Number of agents representing a single physical location.
- Number of event handlers in an agent of each type.
- Number of driver instances for each interaction.
- Total number of invocations of each interaction (as an alternative to specifying a rate).
- Message size distributions for each interaction.
- The driver nodes on which agents are run.
- Number of JVMs run on each node and the way agents are distributed among them.
- Number of javax.jms.Connection objects shared amongst event handler classes within a single agent.
- ....

Overview

I. Introduction
II. Workload Requirements and goals of the SPECjms benchmark
III. Application Scenario for SPECjms
IV. Implementation Details
V. Summary

Summary

- The presented scenario models a set of interactions in the **supply chain** of a supermarket company.
- These interactions are used as a basis in SPEC's new SPECjms benchmark.
- SPECjms will be the **world's first industry-standard benchmark for MOM products.**
- SPECjms can be used to **stress and evaluate** the different aspects of JMS performance.
- SPECjms is **more than a benchmark:**
  Offers also a performance analysis tool for JMS-based infrastructures.

Thanks for your attention