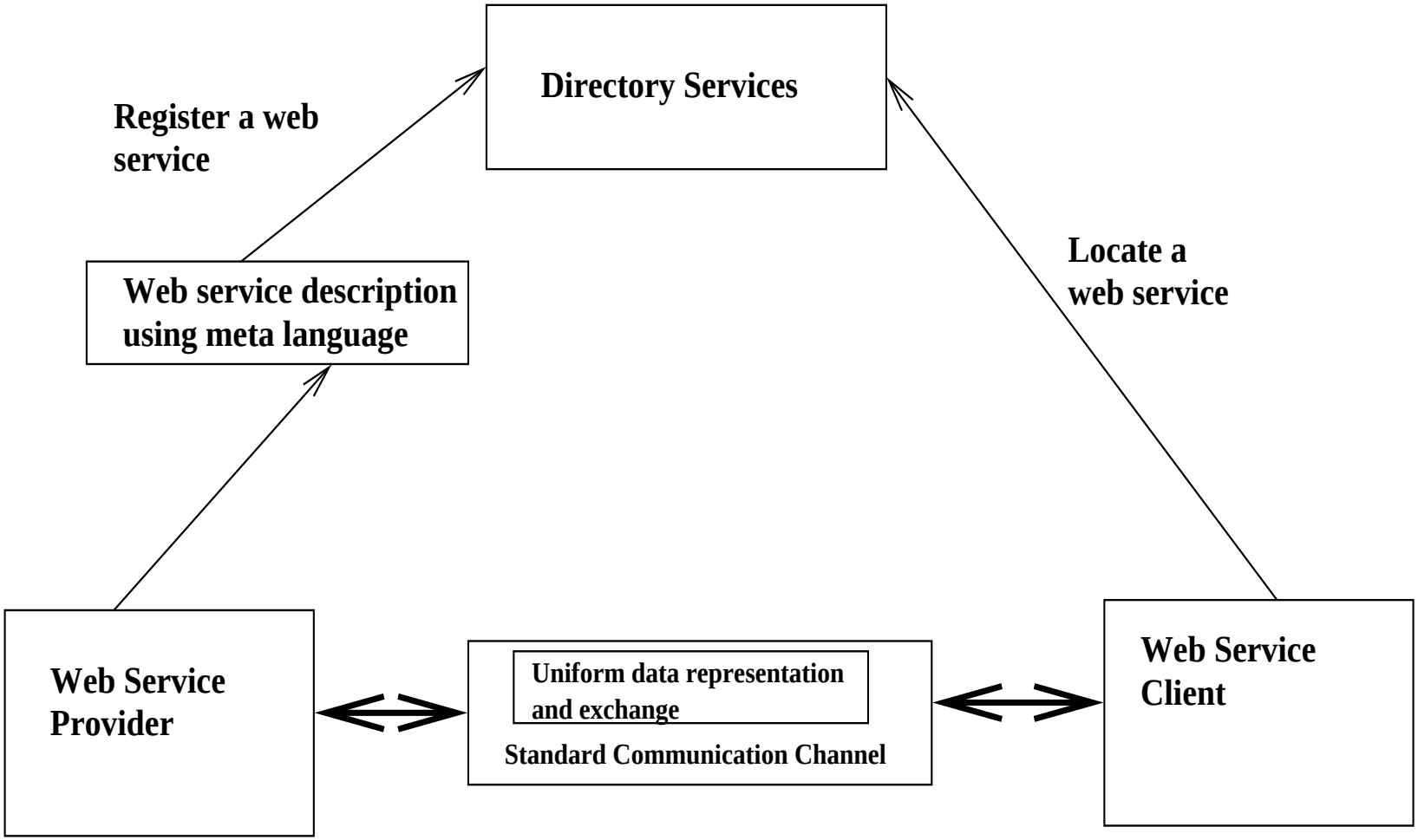
4.3: Web Services

Web services are software functions that can be invoked by clients across internet.

Web services support integration of applications at different network locations, enabling these applications to function as if they were part of a single large software system.

Web services are example of services in service-oriented architecture (SOA), and can be used to provide functionalities offered by public or private clouds in cloud computing.



Web service architecture

Web Services

- **Web service host** – the machine on which web service executable resides
- **Publisher of a web service** – makes web service available for use
- **Consumer of a web service** – someone who uses a web service.

There are several ways in which an application can make data and services available to other applications over internet:

Raw HTML Most basic way a client program can extract data from server is by downloading web pages and then parsing them. Advantage: does not depend on software at server, beyond support of HTTP. But analysis of data depends on format of the web pages – which can change at any time.

CSV A server may make its data available as *comma-separated value* files, a text format for database tables. Eg:

```
Type, Price, Bedrooms, Area
Flat, 208000, 2, SW11
Terraced house, 450000, 3, SW19
Flat, 550000, 4, SE1
```

The yahoo.co.uk finance site adopts this approach, providing CSV files of FTSE 100 and other financial data.

FTP *File transfer protocol* provides a means to access files stored on a remote computer connected to internet.

SOAP More sophisticated approach is to use protocol designed for application-to-application communication across web, such as SOAP (Simple Object Access Protocol) http://www.w3.org/TR/SOAP, an XML-based protocol for exchanging messages, including descriptions of remote procedure calls. A SOAP message is an XML document, an *envelope*, which describes method call that message concerns. Body of message can either be request or response.

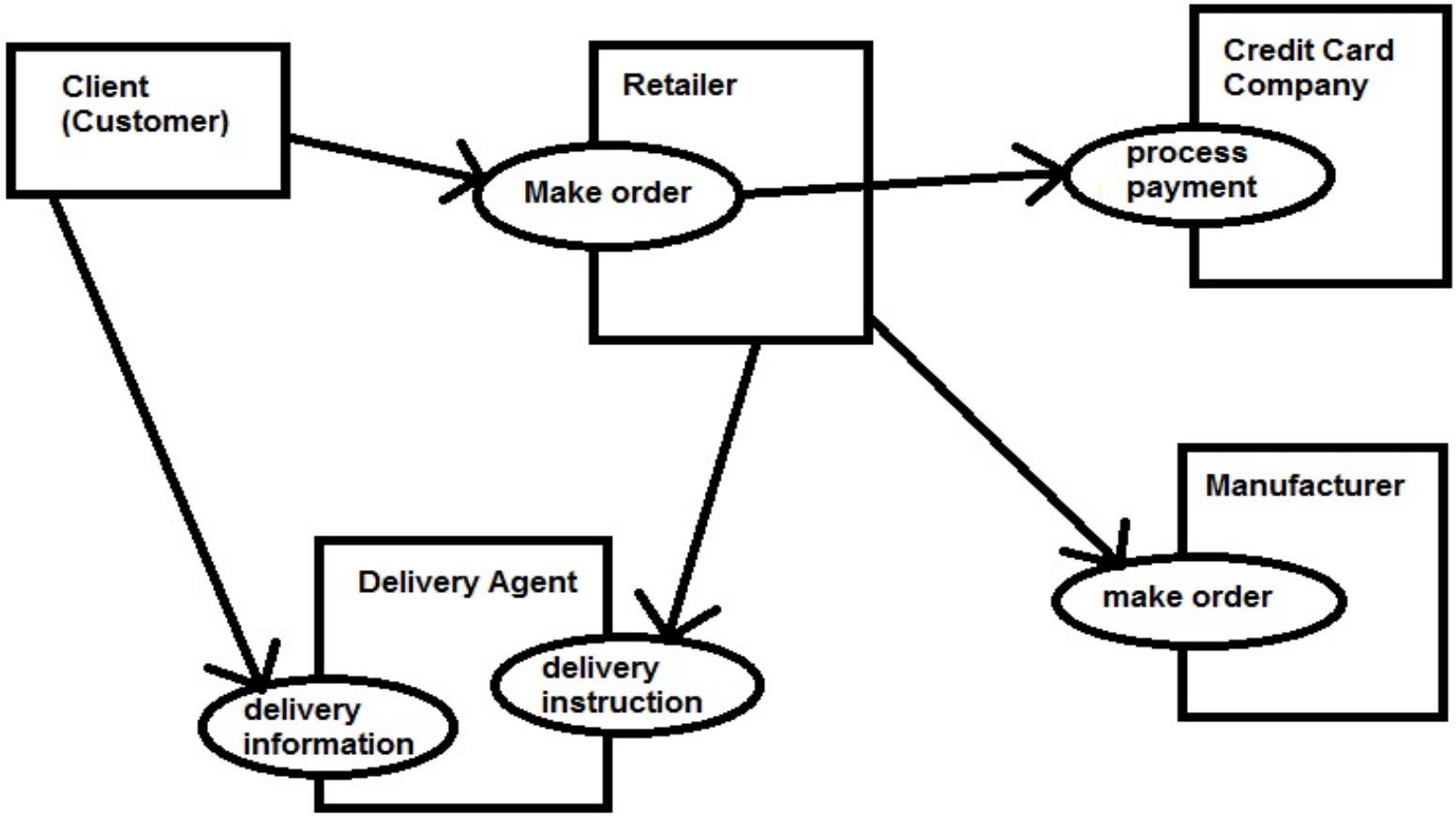
WSDL Web Services Definition Language is also XML-based (http://www.w3.org/TR/wsdl). It supports description of network services operating on messages with document or procedural content.

Web services: restrictions

A task may be made into a web service if:

- It involves access to remote data, or other business-to-business (B2B) interaction.
- It represents a common subtask in several business processes.
- If it does not require fine-grain interchange of data.
- If it is not performance-critical.

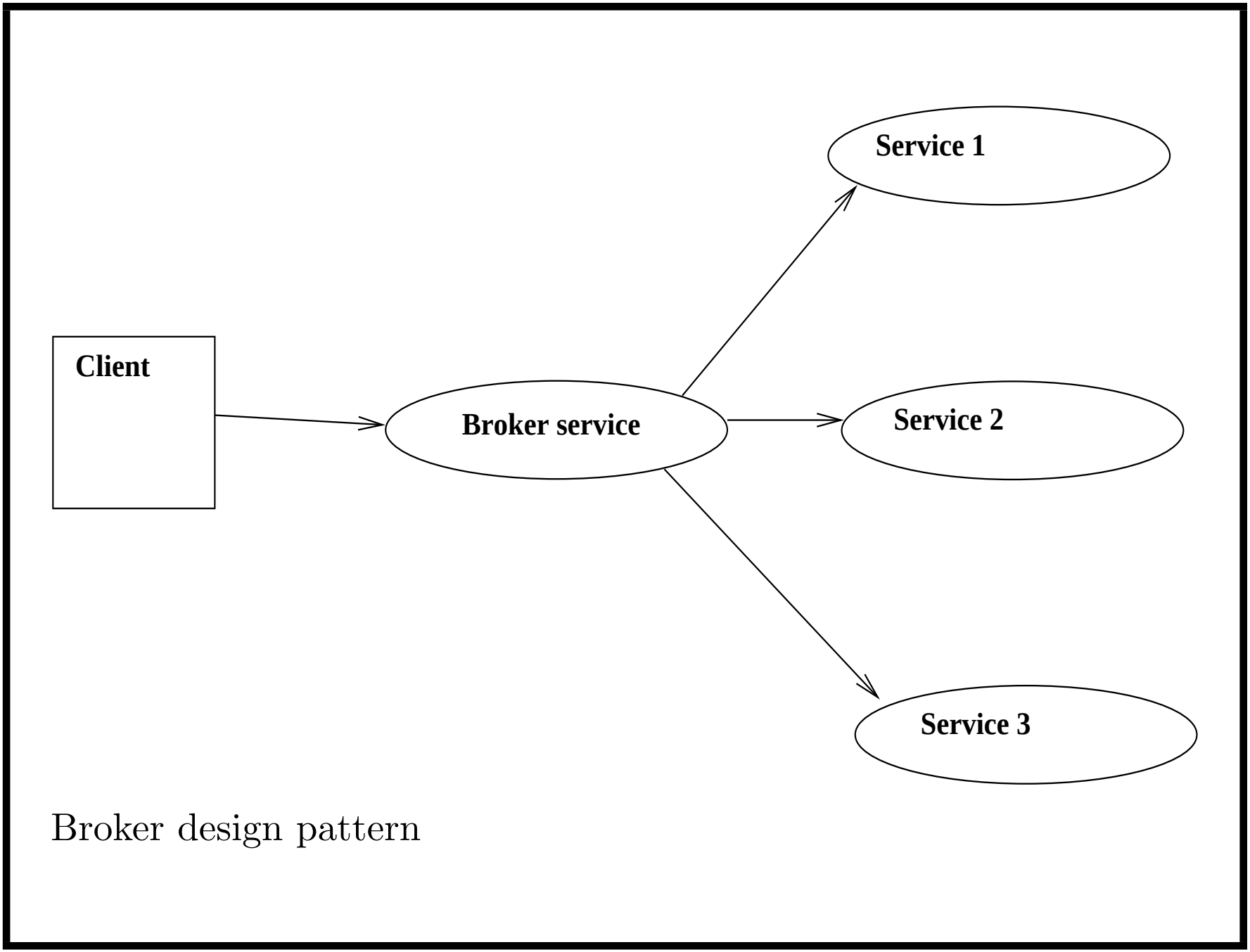
Web service invocation is relatively slow because it uses data transmission over the internet, and packaging of call data.



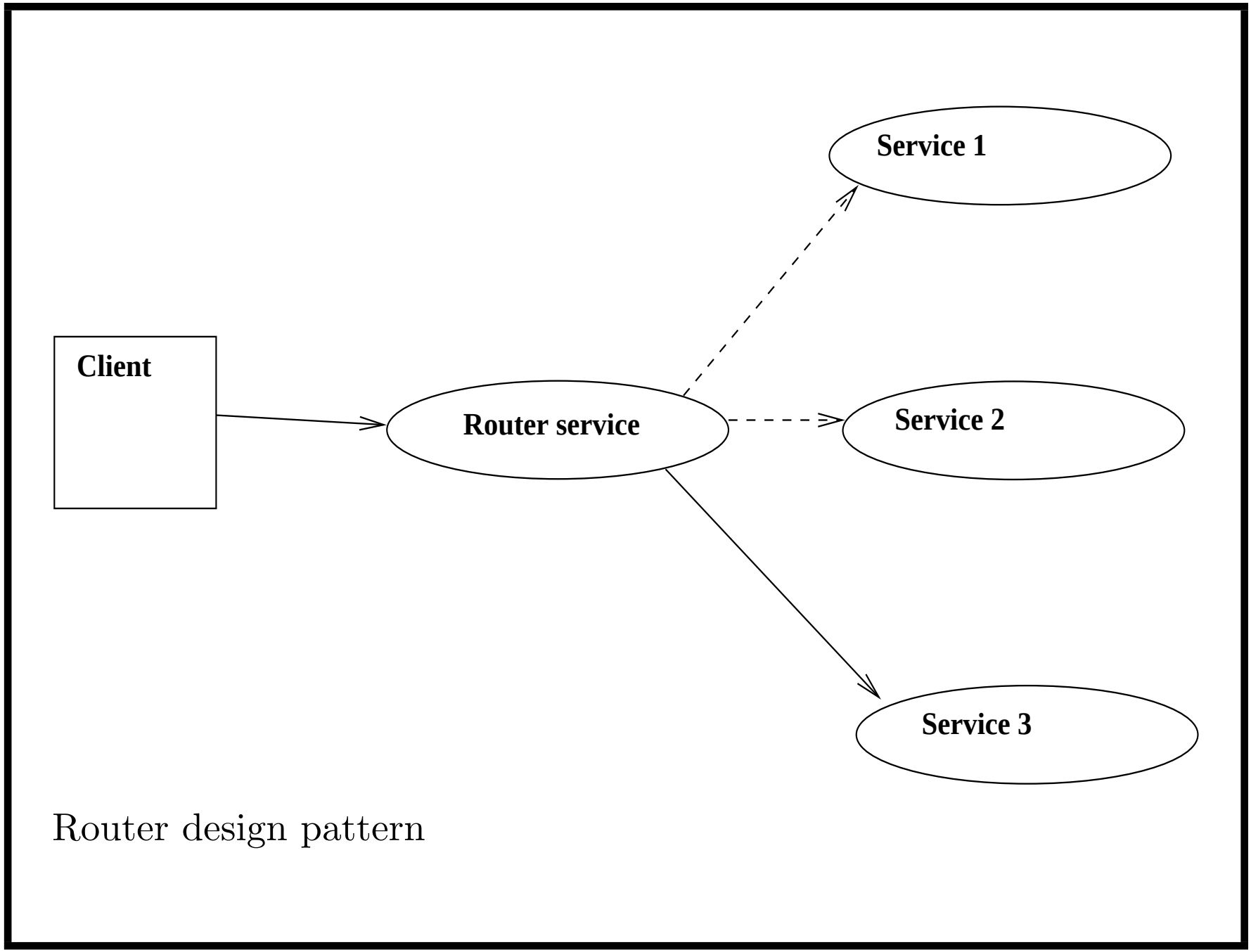
Typical Business-to-business web service architecture

Web Service Design Patterns

- **Broker design pattern**: source application needs to call multiple target services (eg., to find price of an item supplied by alternative suppliers). Pattern introduces a broker service to perform this distributed call.
- **Router design pattern**: source application needs to call one specific service, depending on various criteria/rules. Pattern introduces router service which applies these rules to select correct target service.



Broker design pattern



Router design pattern

Implementing web services using J2EE

J2EE provides the JAX-RPC API to program web services that communicate using an XML-based protocol such as SOAP.

JAX-RPC hides details of SOAP message formats and construction, and is similar to the Java RMI (Remote Method Invocation) interface.

Unlike RMI, web clients and services do not have to run on Java platforms, since HTTP, SOAP and WSDL are used to support client-server communication, independent of particular programming languages.

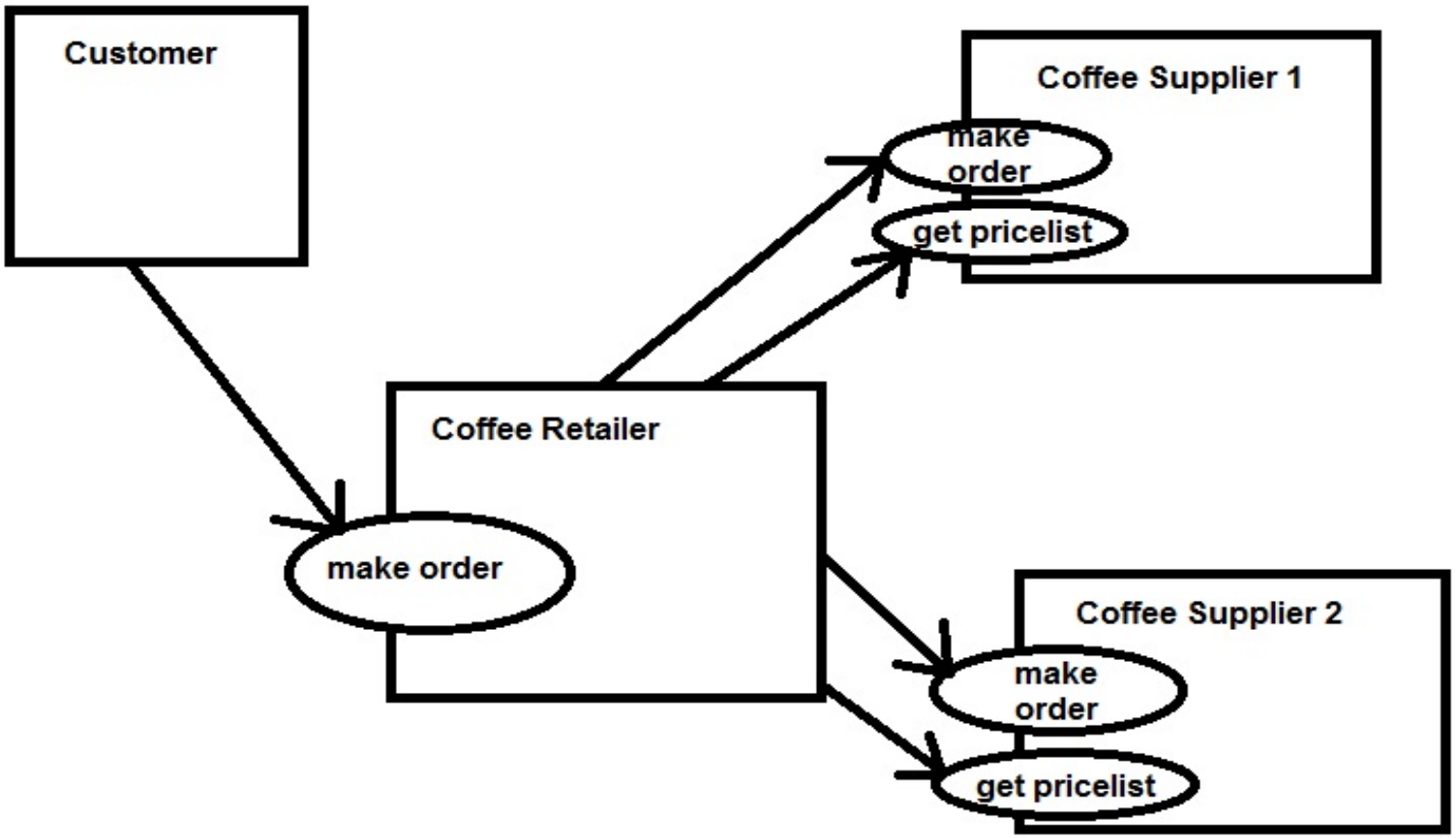
J2EE also provides means to directly construct SOAP messages and interact with web services by sending such messages. The SAAJ (SOAP with Attachments API for Java) API supports construction of SOAP messages, and transmission of these over a *SOAPConnection*.

Web service example: coffee break system

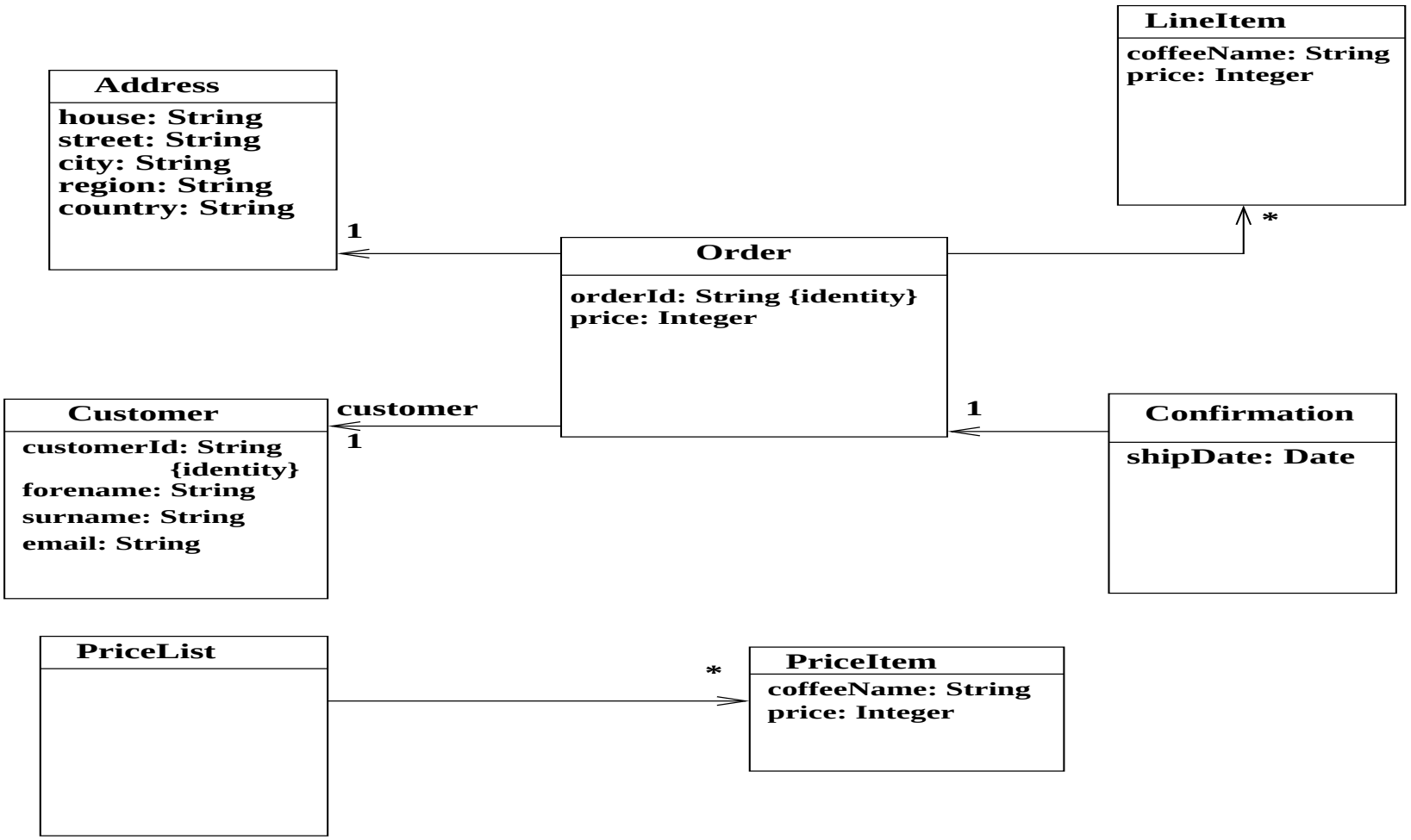
This example from J2EE web service tutorial illustrates JAX-RPC and SAAJ web service mechanisms. The application receives orders from customers, and sends orders to coffee suppliers, using web services provided by the suppliers to supply price lists and order coffee.

One supplier uses SAAJ with pre-defined XML message formats (DTDs), the other uses JAX-RPC.

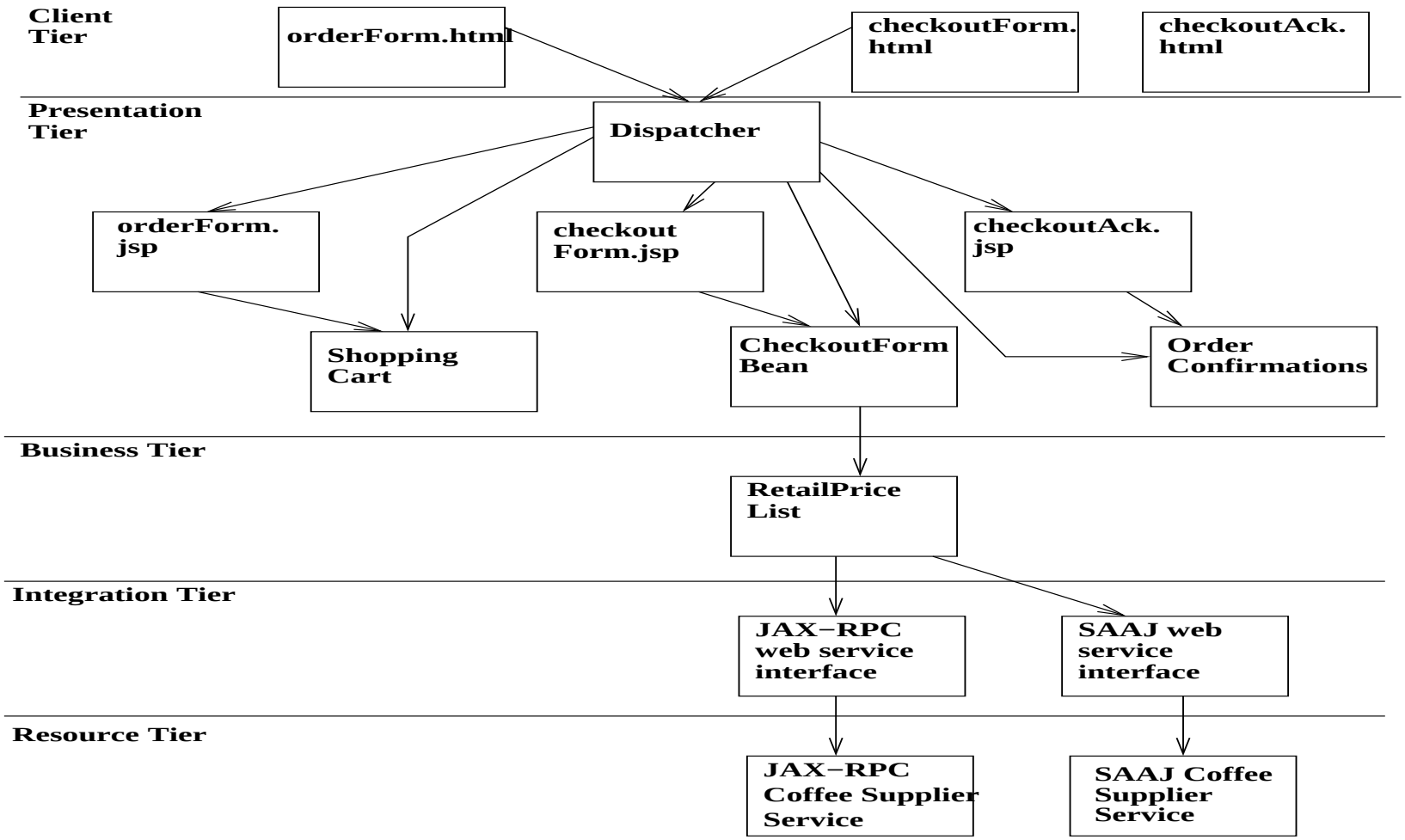
System could use Broker pattern to find best supplier, and Router to order from this supplier.



Architecture of coffee system



Class diagram of coffee system



Coffee system implementation architecture

4.4: REST and RESTful Web Services

Representational State Transfer (REST): data and functionality are accessed using Uniform Resource Identifiers (URIs), eg., URLs. REST is a client/server architecture for a stateless communication protocol, eg. HTTP.

- Resources are identified by URIs, global addresses for resources and services.
- Resources are manipulated using PUT, GET, POST, and DELETE. PUT creates a new resource, which can be then deleted by using DELETE. GET retrieves current state of a resource. POST transfers a new state onto a resource.
- Resources are decoupled from their representation so their content can be accessed in a variety of formats, such as HTML, XML, plain text, PDF, JPEG, JSON, and others.
- Stateful interactions through hyperlinks: Every interaction

with a resource is stateless: request messages are self-contained. Stateful interactions are based on the concept of explicit state transfer, eg., by using URI rewriting, cookies, and hidden form fields. State can be embedded in response messages to point to valid future states of the interaction.

JAX-RS in Java EE6 permits direct definition of RESTful web services in Java.

Compared to SOAP/WSDL approach, REST is simpler (no special languages needed) but less flexible: clients must know precise URL of service.

Summary

In this part we have described the Java J2EE and Java EE platforms for enterprise systems, and introduced technologies that can be used for EIS development on these platforms.

We have introduced the concept of web services and web service patterns.