Al Sham Private University
Faculty of Informatics Engineering

جامعة الشام الخاصة
كلية الهندسة المعلوماتية

نظم معلومات موزعة
Distributed Information Systems

Lecture 6: Web Services

اعداد: أ. غاندي هسام

Introduction

- The growth of the Web in the last two decades proves the effectiveness of using simple protocols over the Internet as the basis for a large number of wide-area services and applications.
- In particular, the HTTP request-reply protocol, allows general-purpose clients, called browsers, to view web pages and other resources with reference to their Uniform Resource Locators (URLs – see the box below for a note on URIs, URLs and URNs).

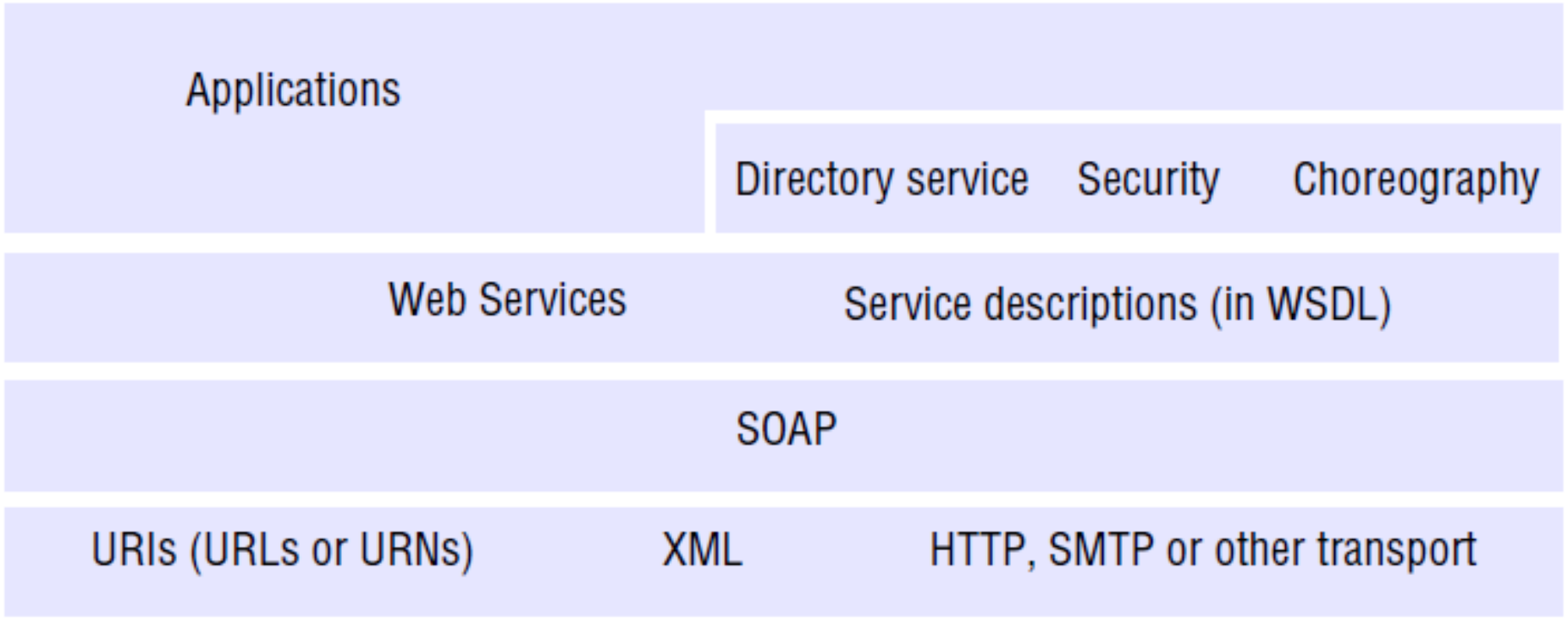
URI, URL and URN • The Uniform Resource Identifier (URI) is a general resource identifier, whose value may be either a URL or a URN. URLs, which include resource location information such as the domain name of the server of a resource being named, are well known to all web users. Uniform Resource Names (URNs) are location-independent – they rely on a lookup service to map them onto the URLs of resources. URNs are discussed in more detail in Section 13.1.

- In the original client-server model, both client and server were functionally specialized.
- **Web services** return to this model, in which an application-specific client interacts with a service with a functionally specialized interface over the Internet.
- Provide an infrastructure for maintaining a richer and more structured form the ability to work together between clients and servers.
- They provide a basis whereby a client program in one organization may interact with a server in another organization without human supervision.

- In particular, web services allow complex applications to be developed by providing services that integrate several other services.
- The provision of web services as an addition to web servers is based on the ability to use an HTTP request to cause the execution of a program.
- The terms ‘web server’ and ‘web services’ should not be confused: a web server provides a basic HTTP service, whereas a web service provides a service based on the operations defined in its interface.
- External data representation and marshalling of messages exchanged between clients and web services is done in **XML**.

- The **SOAP** protocol specifies the rules for using XML to package messages, for example to support a request-reply protocol.
- SOAP is used to encapsulate these messages and transmit them over HTTP or another protocol, for example, TCP or SMTP.
- A web service deploys service descriptions to specify the **interface** and other aspects of the service for the benefit of potential clients.
- Web services provide access to resources for remote clients, but they do not provide a means for coordinating their operations with one another.
- A web service interface generally consists of a collection of operations that can be used by a client over the Internet.

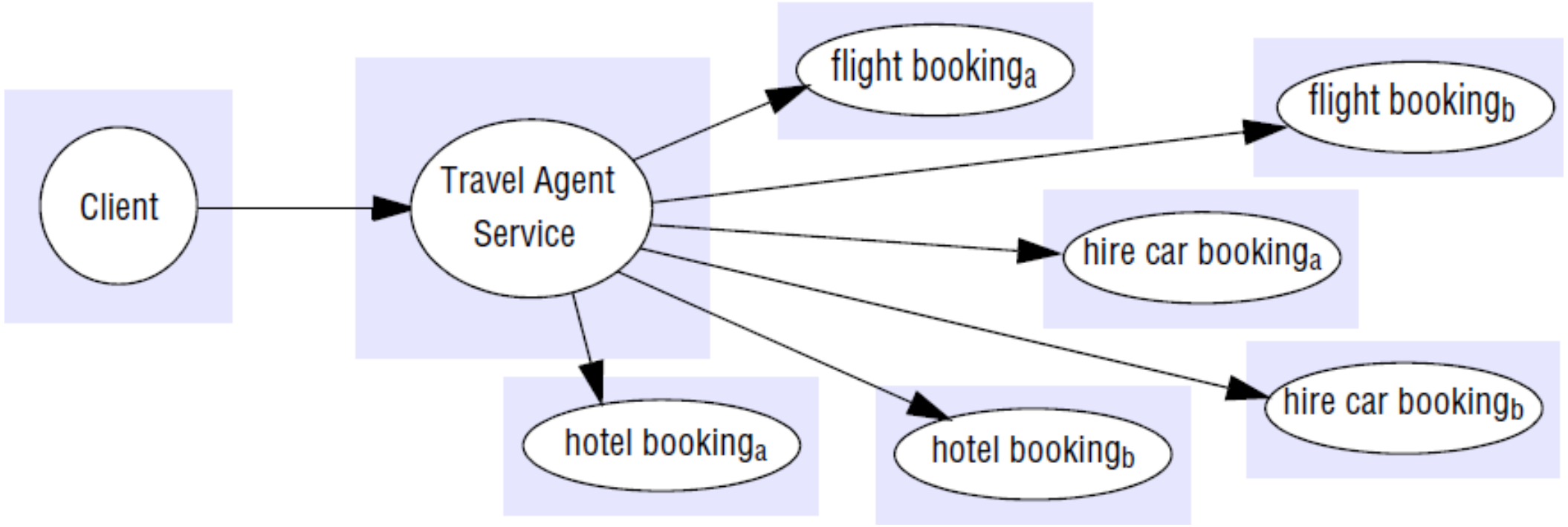
Web services infrastructure and components



Combination of web services

- Many well-known commercial web servers including Amazon, Yahoo, Google and eBay, offer web service interfaces that allow clients to manipulate their web resources.
- As an example, the web service offered by Amazon.com provides operations to allow clients to get information about products, to add an item to a shopping cart or to check the status of a transaction.
- Providing an interface for a web service allows its operations to be combined with those of other services to provide new functionality.
- As another example of the benefits of combining several services, consider the fact that many people book flights, hotels and rental cars for trips online using a variety of different web sites.

The ‘travel agent service’ combines other web services



REST (Representational State Transfer) • REST [Fielding 2000] is an approach with a very constrained style of operation, in which clients use URLs and the HTTP operations *GET*, *PUT*, *DELETE* and *POST* to manipulate resources that are represented in XML. The emphasis is on the manipulation of data resources rather than on interfaces. When a new resource is created, it has a new URL by which it can be accessed or updated. Clients are supplied with the entire state of a resource instead of calling an operation to get some part of it. Fielding argues that in the context of the Internet, the proliferation of different service interfaces will not be as useful as a simple minimum uniform set of operations. It is interesting to note that, according to Greenfield and Dornan [2004], 80% of the requests to the web services at Amazon.com are via the REST interface, with the remaining 20% using SOAP.

SOAP

- SOAP is designed to enable both client-server and asynchronous interaction over the Internet.
- It defines a scheme for using XML to represent the contents of request and reply messages as well as a scheme for the communication of documents.
- Originally SOAP was based only on HTTP, but the current version is designed to use a variety of transport protocols including SMTP, TCP or UDP.
- However, the programmer does not normally need to be concerned with these details, since SOAP APIs have been implemented in many programming languages, including Java, JavaScript, Perl, Python..

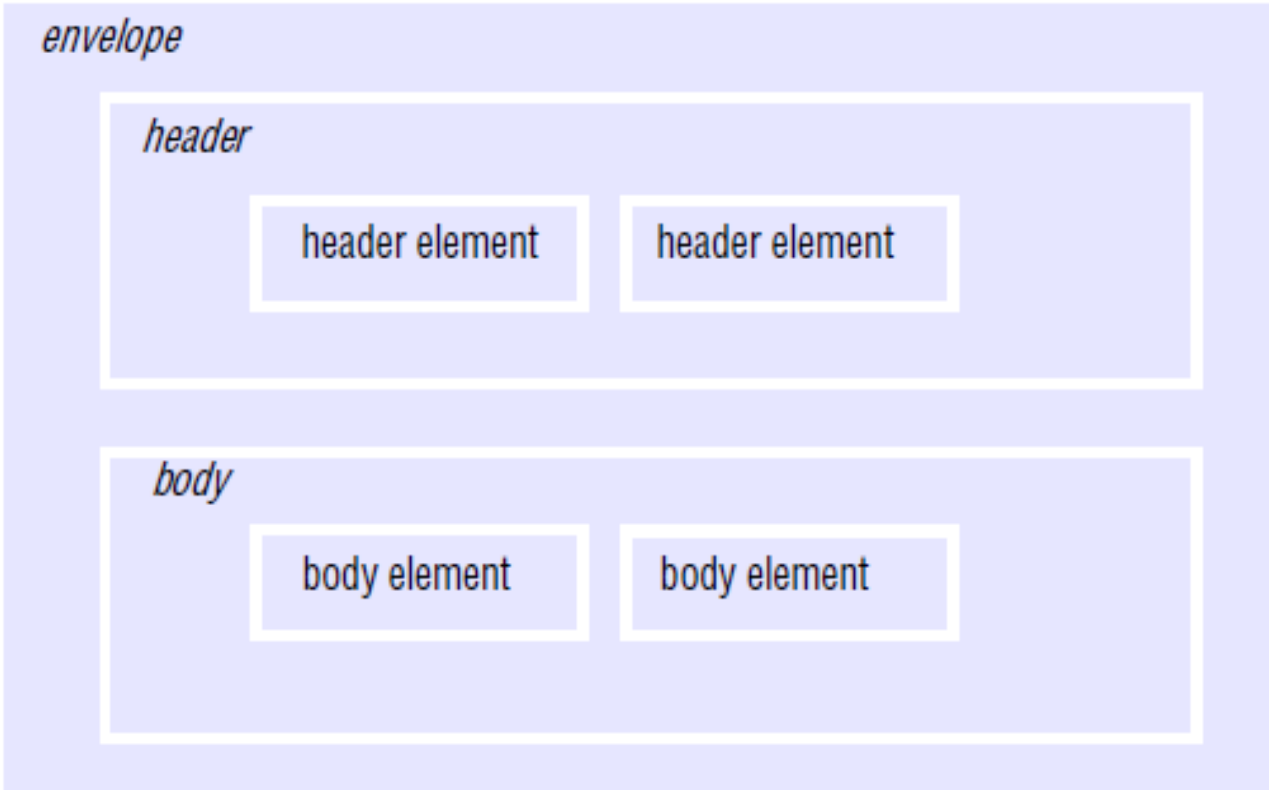
- To support client-server communication, SOAP specifies how to use the **HTTP POST** method for the request message and its response for the reply message.
- The combined use of XML and HTTP provides a standard protocol for client-server communication over the Internet.

❑ SOAP messages

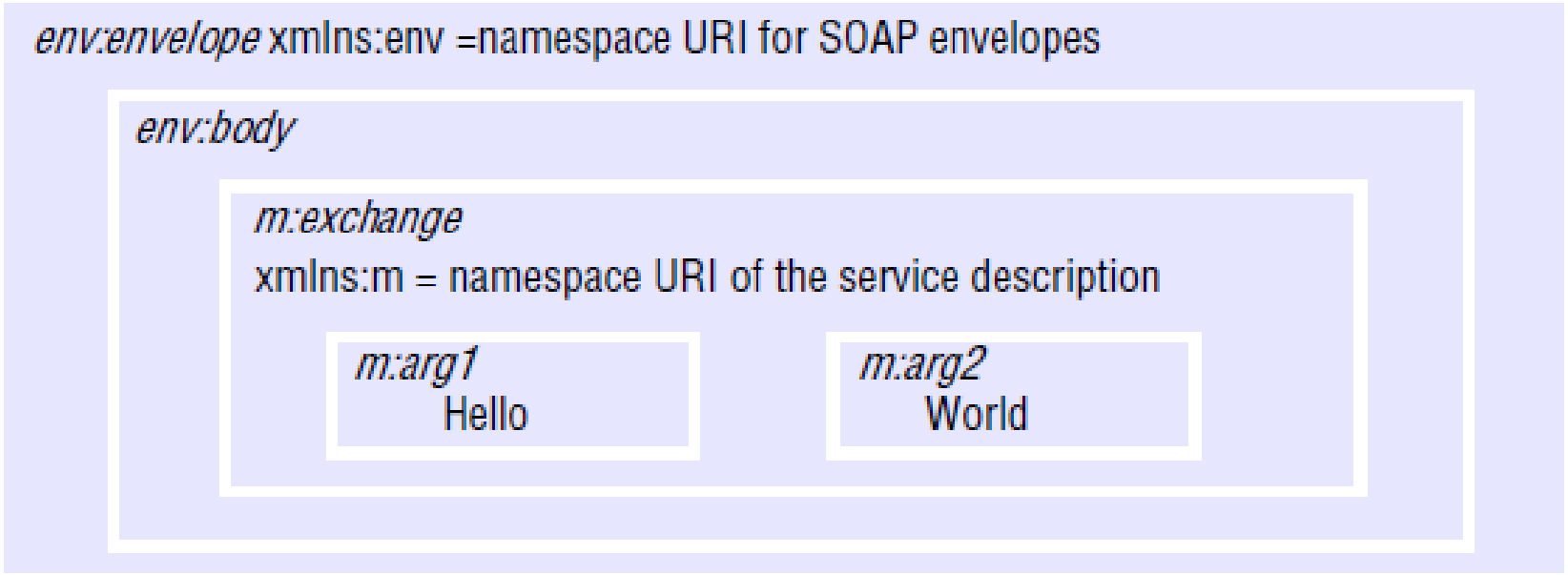
- is carried in an ‘envelope’. Inside the envelope there is an optional header and a body. Message headers can be used for establishing the necessary context for a service or for keeping a log or audit of operations.
- The message body carries an XML document for a particular web service.

- The XML elements envelope, header and body, together with other attributes and elements of SOAP messages, are defined as a schema in the SOAP XML namespace.
- Both the header and the body contain inner elements.

SOAP message in an envelope

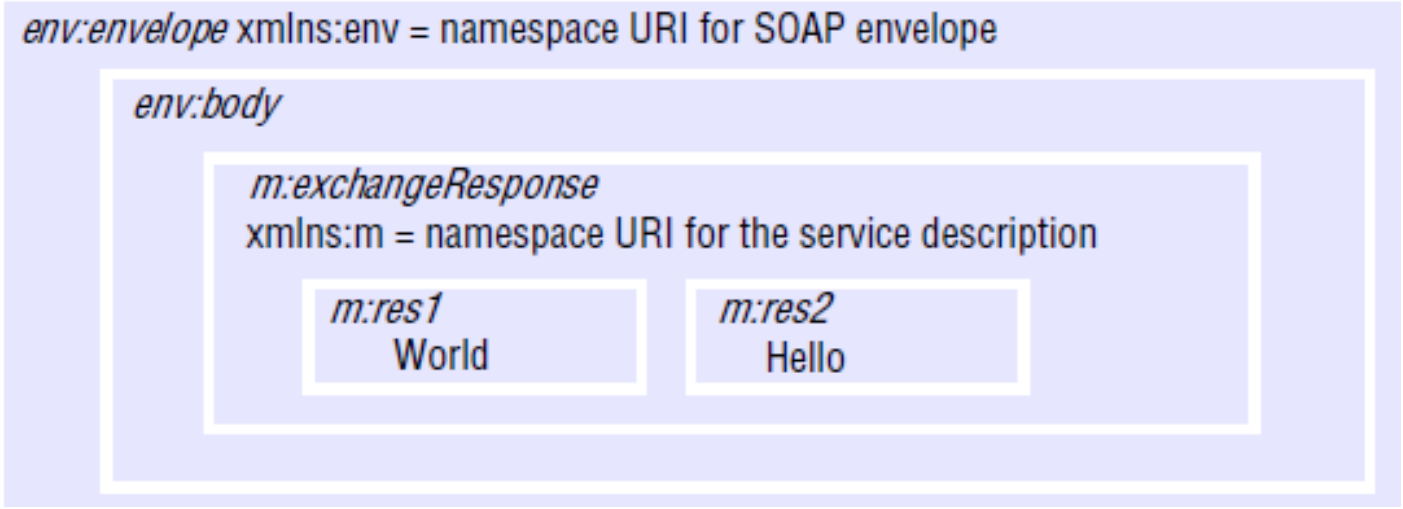


Example of a simple request without headers



- An example of a simple request message without a header.
- The body encloses an element containing the name of the procedure to be called and the URI of the namespace (the file containing the XML schema) for the relevant service description, which is denoted by **m**.
- The XML namespace denoted by env contains the SOAP definitions for an envelope.

Example of a reply corresponding to the request in Figure 9.4

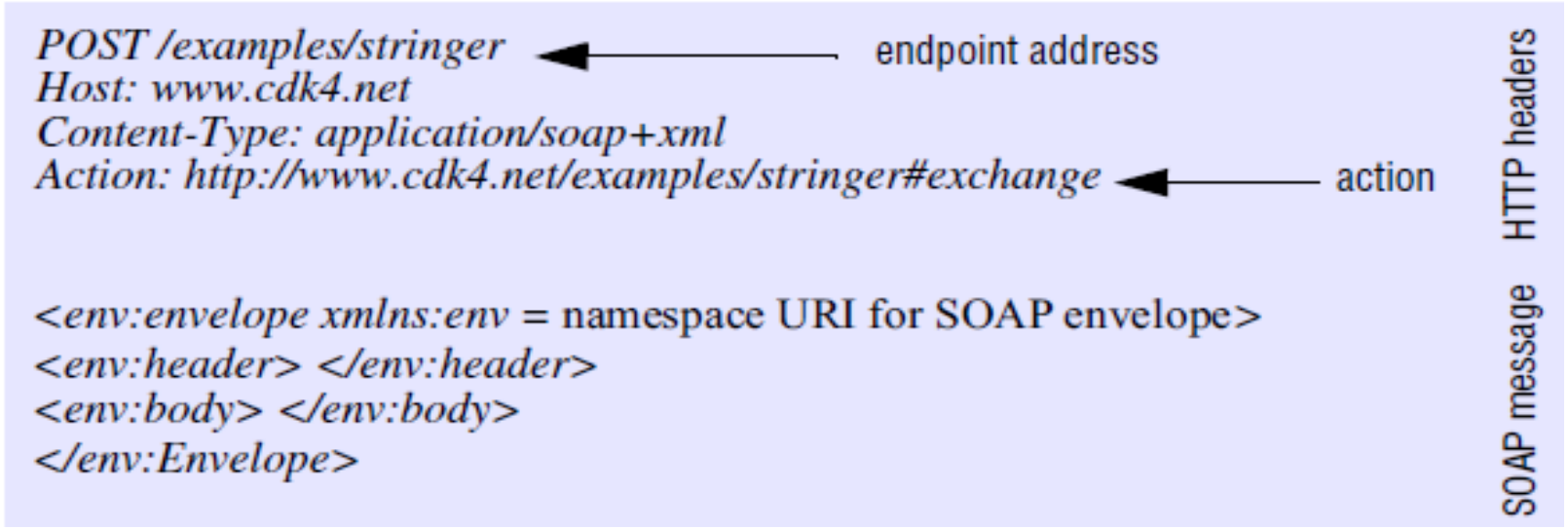


- The corresponding successful reply message, which contains the two output arguments.
- Note that the name of the procedure has ‘Response’ added to it.
- If a procedure has a return value, then it may be denoted as an element called rpc:result.
- The reply message uses the same two XML schemas as the request message, the first defining the SOAP envelope and the second the application-specific procedure and argument names.

❑ Transport of SOAP messages

- SOAP messages are independent of the type of transport used their envelopes contain no reference to the destination address. HTTP (or whatever protocol is used to transport a SOAP message) is left to specify the destination address.

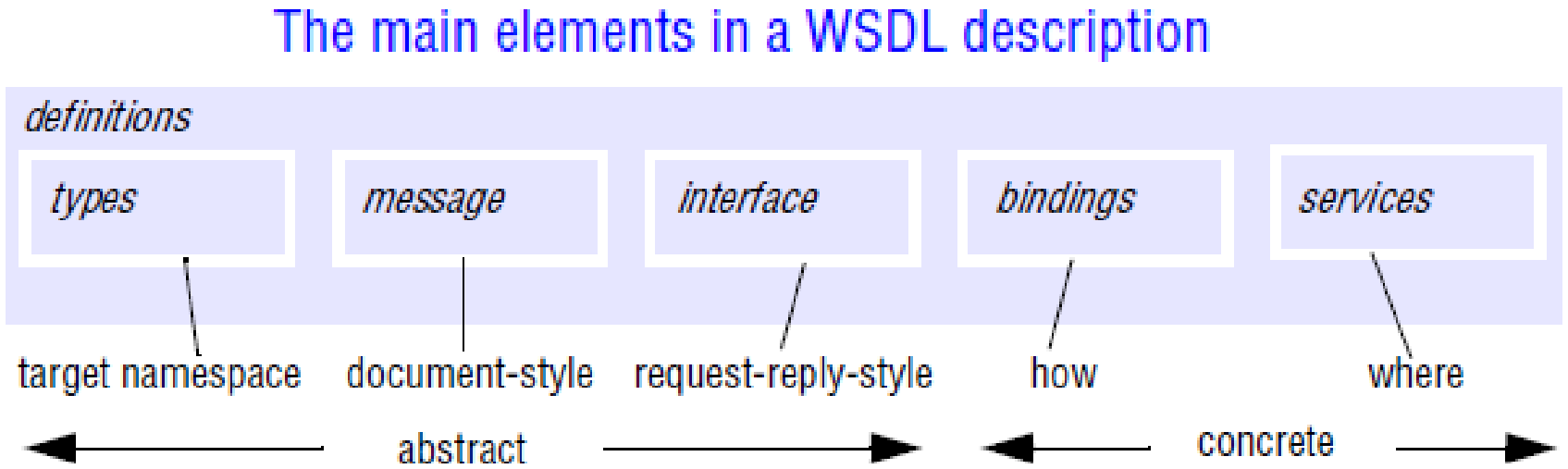
Use of HTTP POST Request in SOAP client-server communication



Service descriptions and IDL for web services: WSDL

- Interface definitions are needed to allow clients to communicate with services.
- For web services, interface definitions are provided as part of a more general service description, which specifies two other additional characteristics – how the messages are to be communicated (for example, by SOAP over HTTP) and the URI of the service.
- service descriptions are written in XML.
- The ability to specify the URI of a service as a part of the service description avoids the need for the separate binder or naming service used by most other middleware.

- the web services context, the Web Services Description Language (**WSDL**) is commonly used for service descriptions.
- It defines an XML schema for representing the components of a service description, which include, for example, the element names *definitions*, *types*, *message*, *interface*, *bindings* and *services*.



- The abstract part of the description includes a set of definitions of the types used by the service – in particular, the types of the values exchanged in messages.
- The set of names defined within the types section of a WSDL definition is called its *target namespace*.
- The message section of the abstract part contains a description of the set of messages exchanged.
- For the request-reply style of interaction, there are two messages for each operation, which are used to describe the operations in the *interface* section.
- The concrete part specifies how and where the service may be contacted.

❑ Messages or operations

- In web services, all that the client and the server need is to have a common idea about the messages to be exchanged.
- For a service based on the exchange of a small number of different types of document, WSDL just describes the types of the different messages to be exchanged.
- When a client sends one of these messages to a web service, the latter decides what operation to perform and what type of message to send back to the client on the basis of the message type received.
- However, in WSDL an operation is a construct for relating request and reply messages, in contrast to the definition of an operation in a service interface.

- For example, the request and reply messages for the *newShape* operation, which has a single input argument of type *GraphicalObject* and a single output argument of type int.

WSDL request and reply messages for the *newShape* operation

```
message name = "ShapeList_newShape"
    part name="GraphicalObject_1"
        type = "ns:GraphicalObject"
```

```
message name = "ShapeList_newShapeResponse"
    part name="result"
        type = "xsd:int"
```

tns – target namespace

xsd – XML schema definitions

End of Lecture 6