Web Services

TECHNOLOGY

Understanding Web Service

```
#include <iostream>
#include <cstdlib>
using namespace std;
int main()
{
    cout << "Hello, world!" << endl;
    return 0;
}
```

Learning Objectives

By the end of this lesson, you will be able to:

- Summarize Web Services
- Examine the different types, features, and components of the web servers
- Look at the main security issues of the web services
- Exemplify the style of software design

Learning Objectives

By the end of this lesson, you will be able to:

- Examine the SOA architecture
- Identify the advantages of SOAP-based Web Services
- Categorize the REST Web Services
- Interpret the different HTTP codes used in the Web Services

A Day in the Life of a Full Stack Developer

You are hired as a developer by an organization and have been asked to develop a web application for the organization. You are responsible for setting up a server and the client application so that messages can be exchanged between both using the request-response methods while maintaining the confidentiality and integrity of the application.

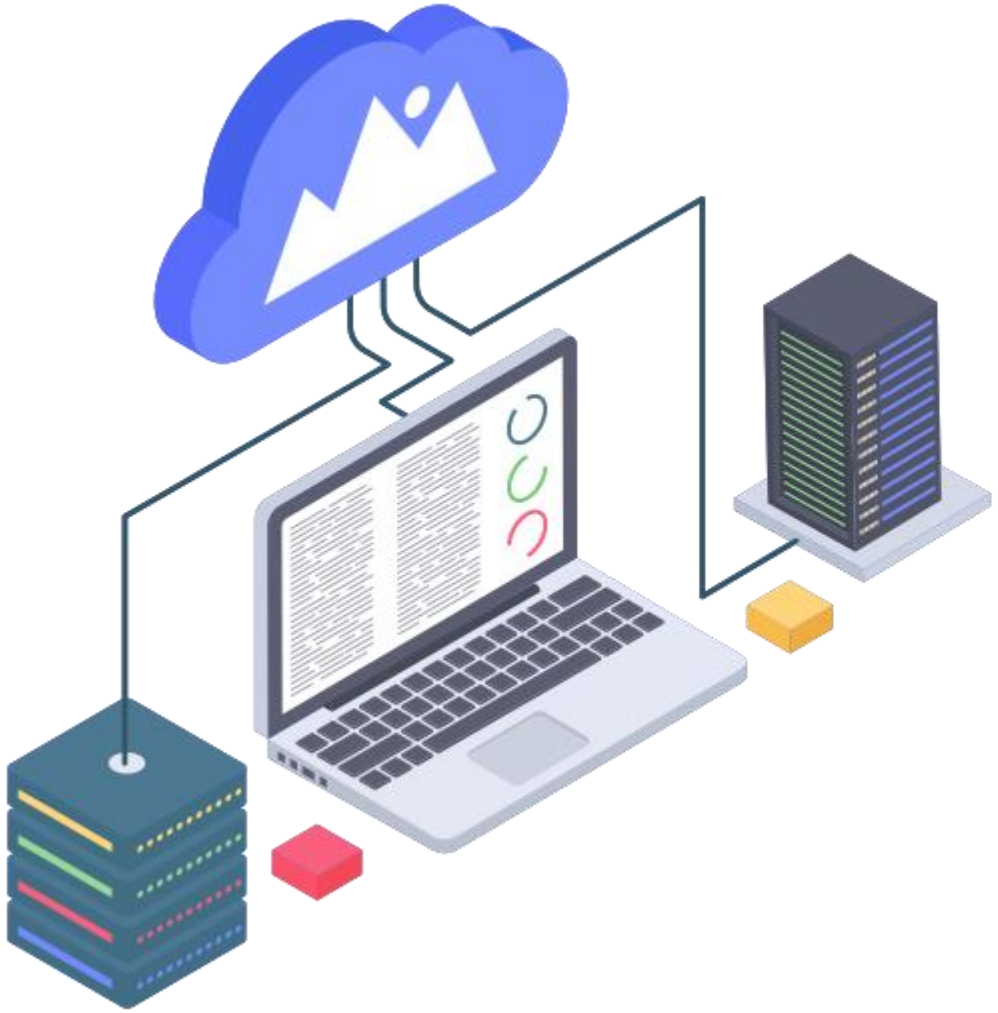
To do so, you use Web Services and SOAP to create a transport-independent messaging protocol between the server and the application.

For this, you must explore Web Services, SOAP, different HTTPS codes, and so on.

Web Services: Overview

Web Service

It refers to a method used to spread messages between the server and client applications.



Web Service

The following are the characteristics of Web Service:

- A Web Service is a type of software module.
- A Web Service is used to carry out a specific set of functions.
- A Web Service is called over the network in cloud computing.

Web Service

The following are the characteristics of Web Service:

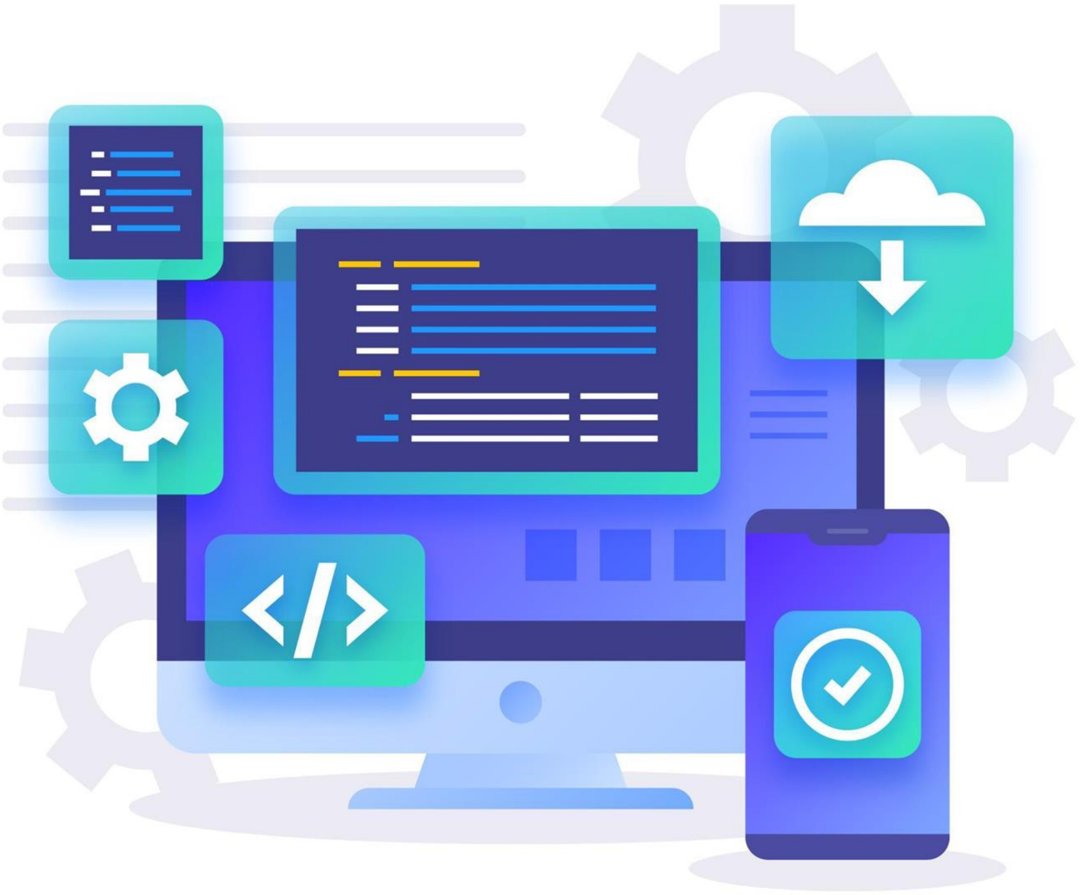
- Web Services are open, standard-based web applications.
- Web Services interact with other applications for the exchange of data.
- Web Services help to convert existing applications to web applications.

Web Service

Any application that uses web protocols for connecting, interoperating, and exchanging data messages is a Web Service.

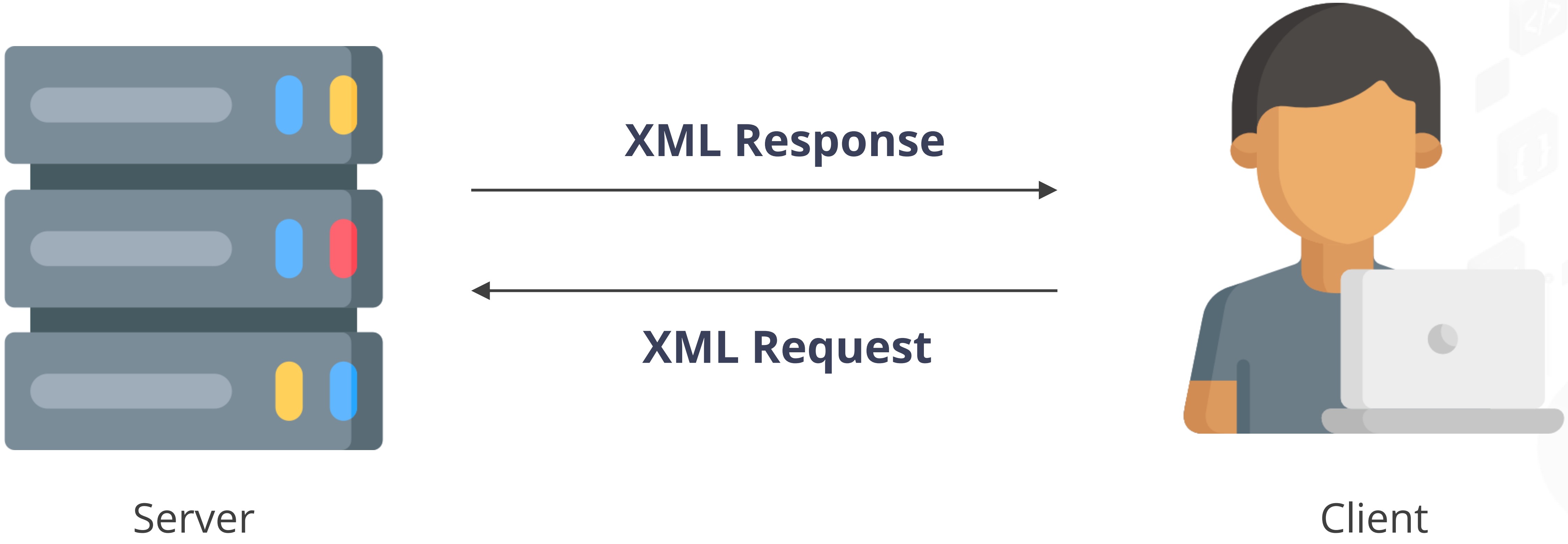
Web Service

It allows programs developed in different languages to connect with one another through the exchange of data over a Web Service between servers and clients.



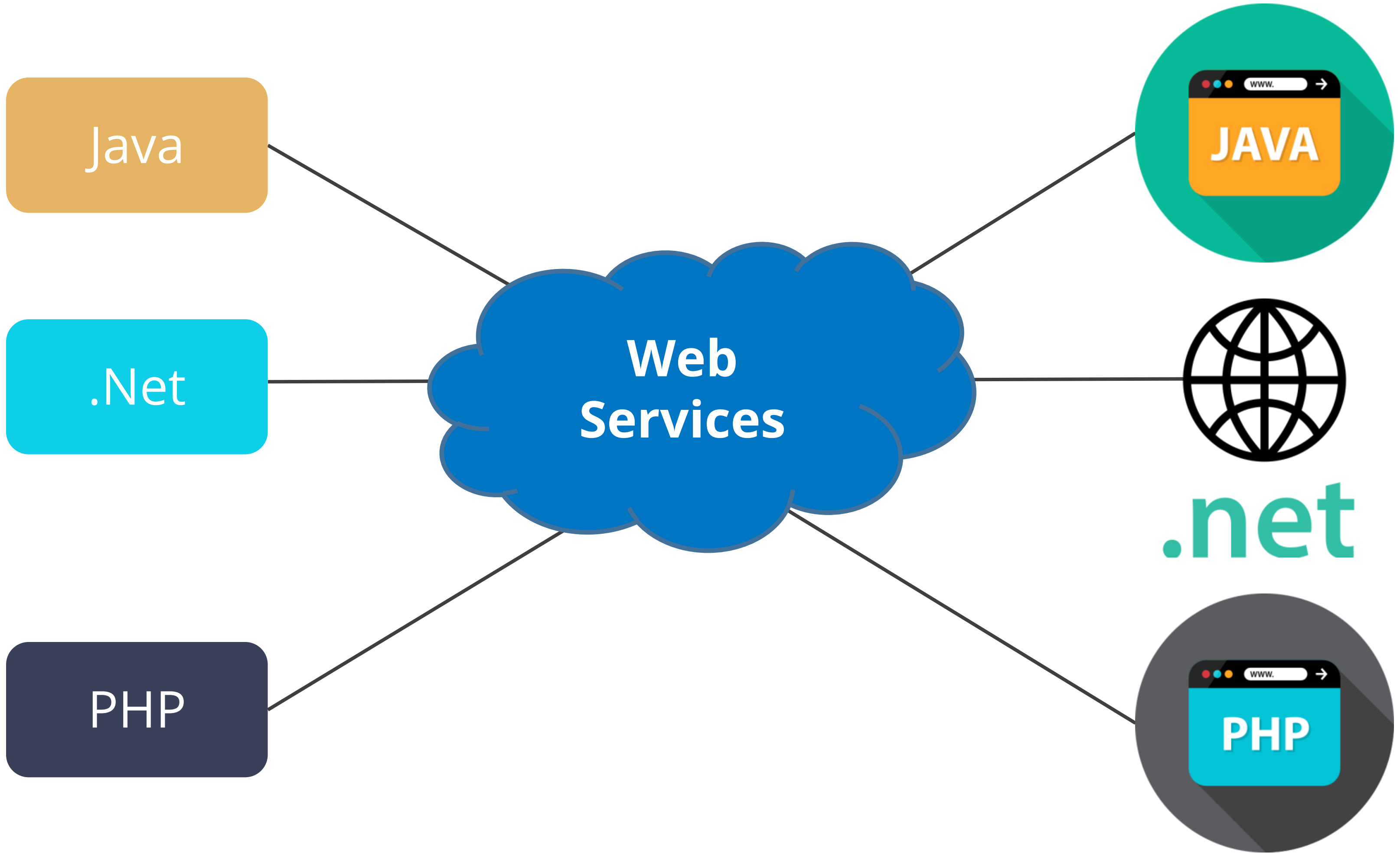
Working of a Web Service

The service responds with an XML response when a client calls a Web Service by submitting an XML request.



Web Service

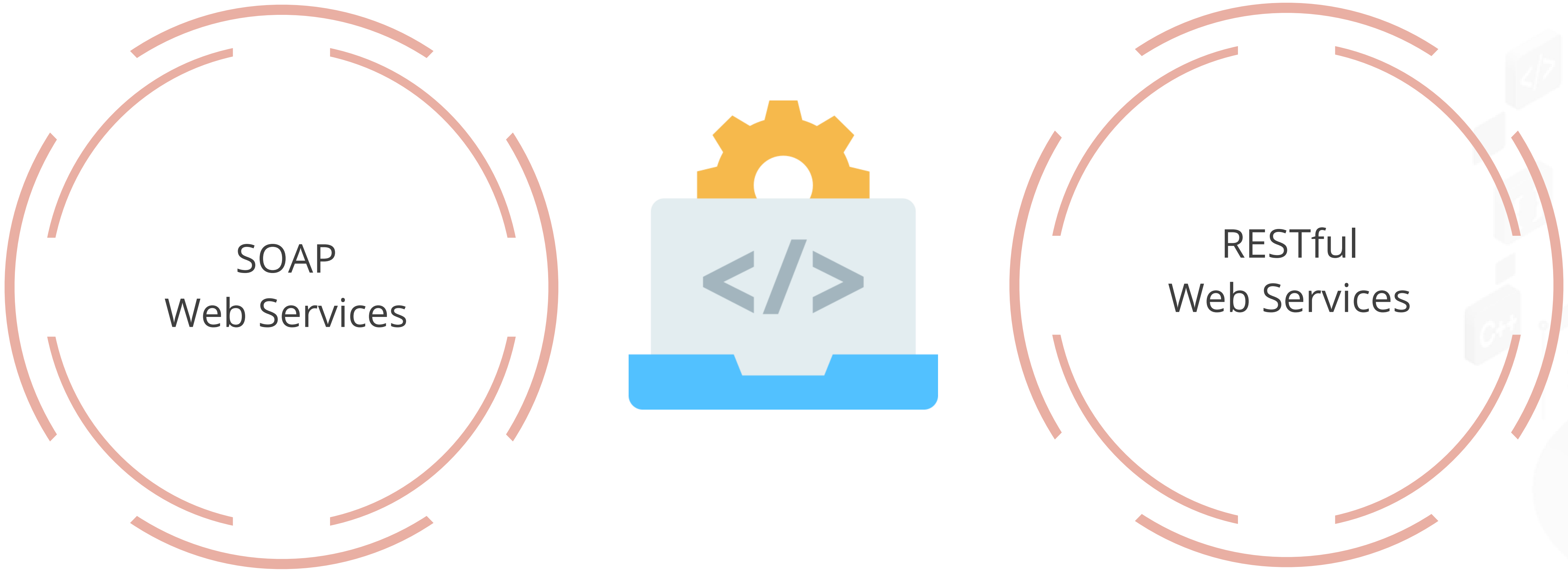
Shown here are Java, .NET, and PHP applications communicating with other applications with the help of Web Service.



Web Service is a language-independent form of communication.

Web Service

There are two major types of Web Services:



SOAP
Web Services

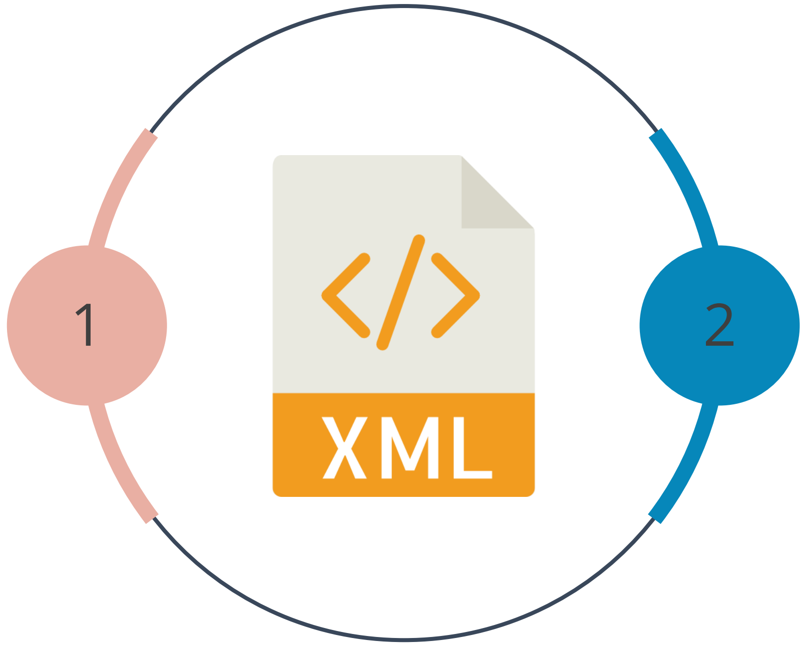
RESTful
Web Services

TECHNOLOGY

Features of Web Services

XML Based

Web Services uses XML for:



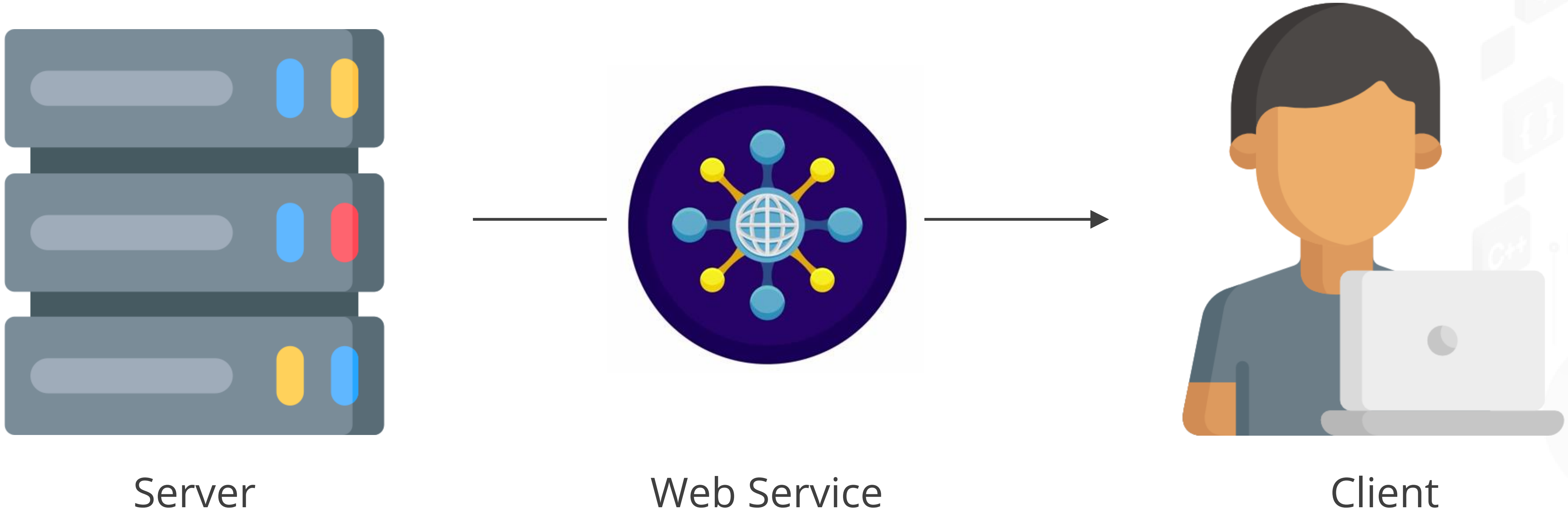
Data transportation

Data description layers

XML helps to exclude any operating system, networking, or platform building.

Loosely Coupled

The Web Service interface supports the communication of the client with the server.



Loosely Coupled

A loosely coupled architecture:

1 Makes the software systems more manageable

2 Allows more straightforward integration across systems

Coarse-Grained

Object-oriented technologies expose their services with the help of individual methods.

A Java program development needs various fine-grained methods created and collected into a coarse-grained service.

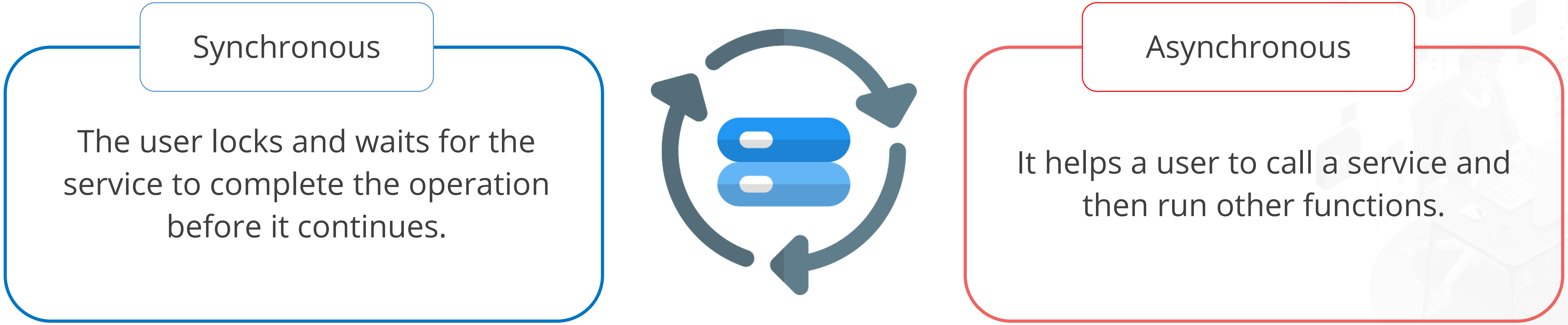
Coarse-Grained

It is necessary for the exposed interface and the businesses to be coarse-grained.

Web technology also provides a natural way to define coarse-grained services.

Coarse-Grained

Web Services can be Synchronous or Asynchronous.



Synchronous

The user locks and waits for the service to complete the operation before it continues.

Asynchronous

It helps a user to call a service and then run other functions.

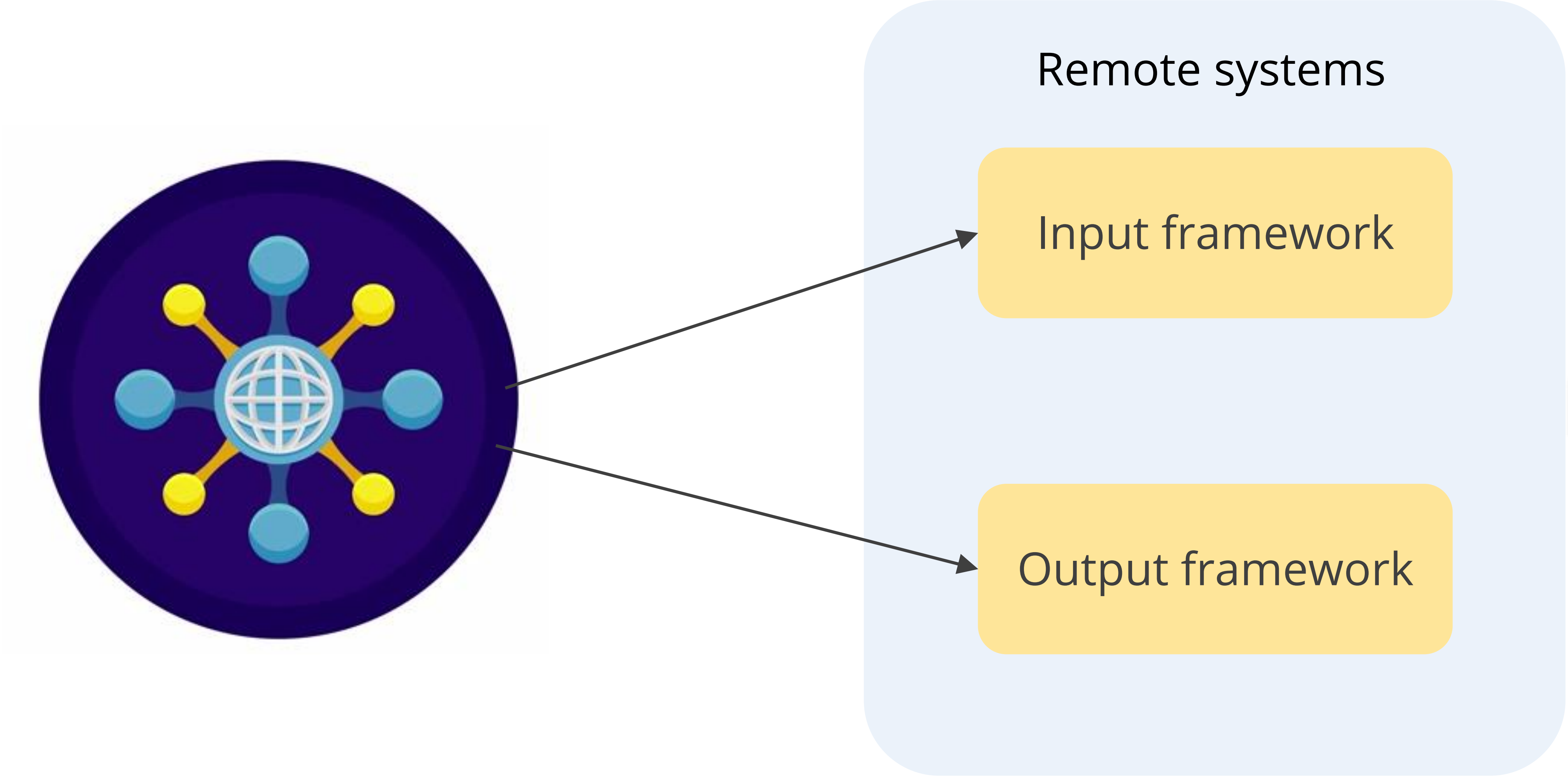
Supports Remote Procedure Calls

Using an XML-based protocol, Web Services help the user to call:

- Functions
- Procedures
- Methods on remote objects

Supports Remote Procedure Calls

Remote systems expose output and input frameworks that Web Services should support.



Remote systems

Input framework

Output framework

Supports Remote Procedure Calls

Web Services support the RPC mechanism by:

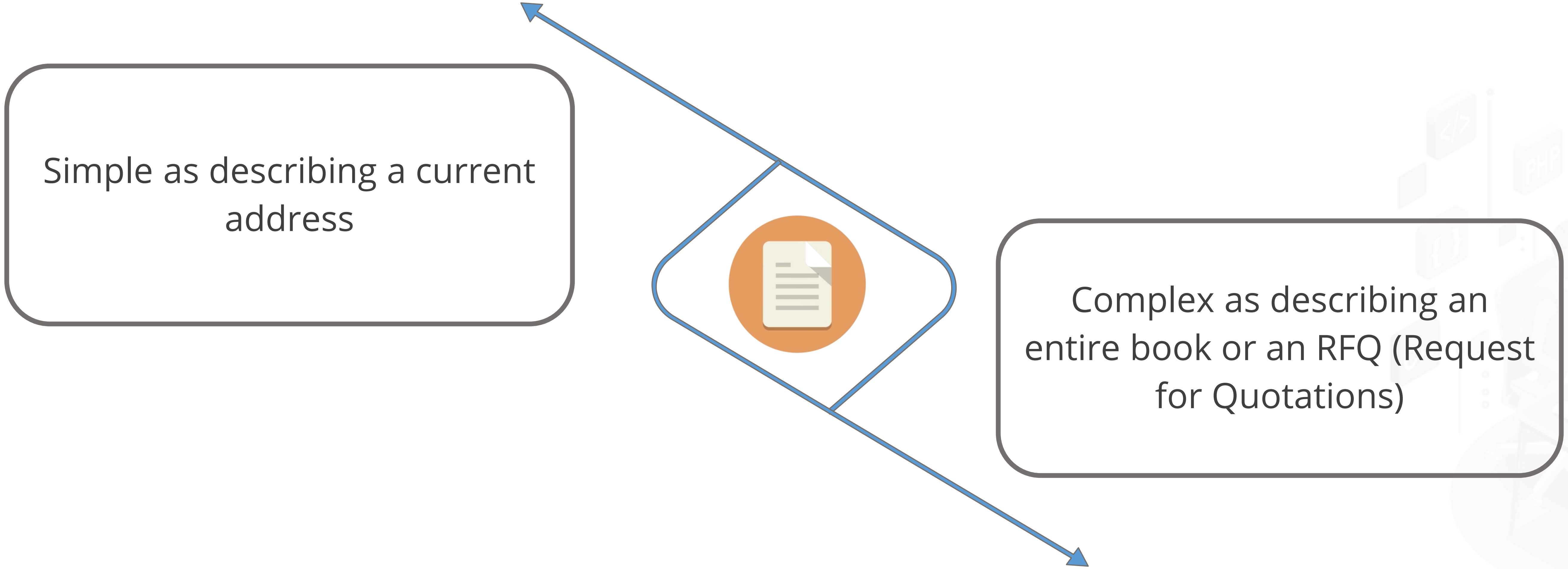
1. Providing services of their own, equivalent to those of a traditional role

OR

2. Changing the invocations that are received into a .NET component or EJB

Supports Document Exchange

Web Services use XML for describing data and complex documents.



Simple as describing a current address

Complex as describing an entire book or an RFQ (Request for Quotations)

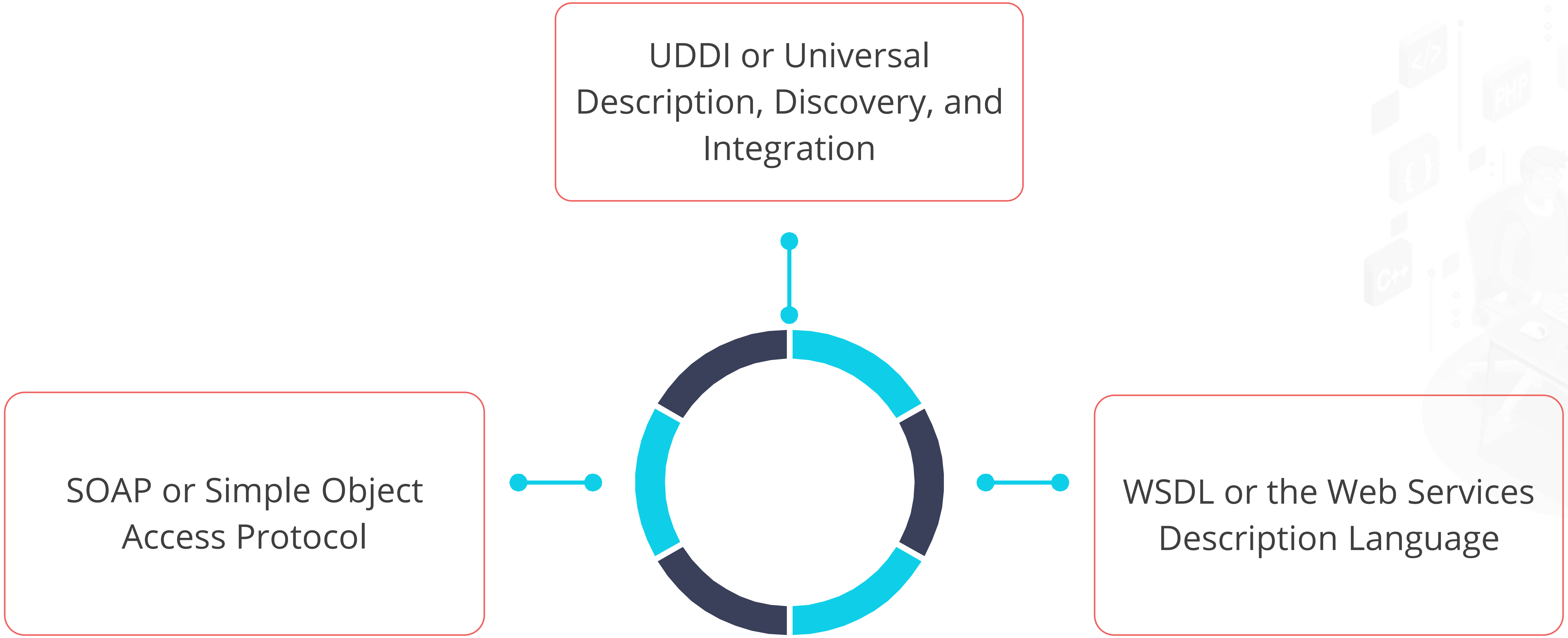
Web Service supports the clear exchange of documents for facilitating business integration.

Components of Web Services

Web Service Platforms

HTTP and XML are referred to as the most basic Web Service platforms.

Certain components are used by all the Web Services, like:



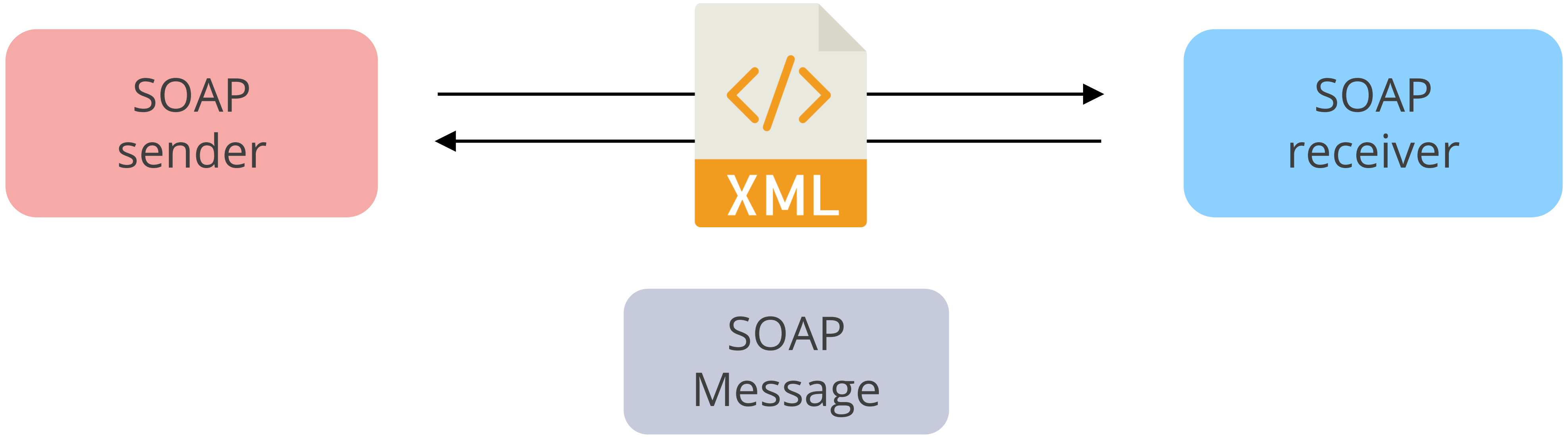
UDDI or Universal Description, Discovery, and Integration

SOAP or Simple Object Access Protocol

WSDL or the Web Services Description Language

SOAP

SOAP is a transport-independent messaging protocol that is created when sending XML data in the form of SOAP Messages.



The structure of these XML documents follows a pattern.

SOAP

In SOAP, everything is shared as HTTP, the standard web protocol.

It needs a root element known as the elements in every document.

SOAP

The XML envelope is divided into two parts:

Header

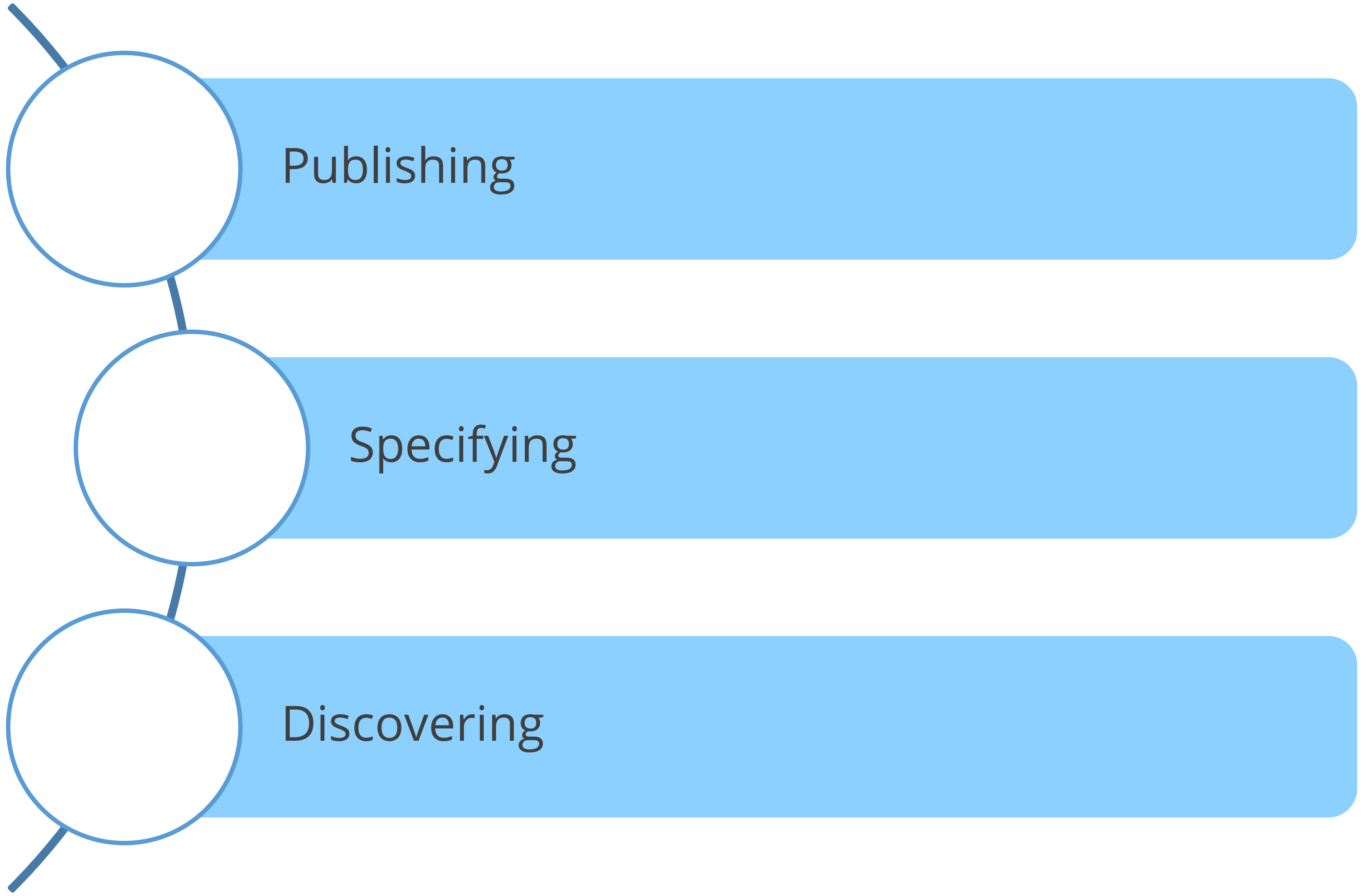
The information or the routing data that controls the XML document is contained in the header.

Body

The body consists of a real message.

UDDI

UDDI or Universal Description, Discovery, and Integration is a standard way for:



- Publishing
- Specifying
- Discovering

UDDI

The following are the features of UDDI:

- Provides a specification that helps in hosting data through Web Services
- Provides a repository where WSDL files can be hosted
- Contains all the required information for the online service to locate it

WSDL

If there is a problem in finding a Web Service, then the client:

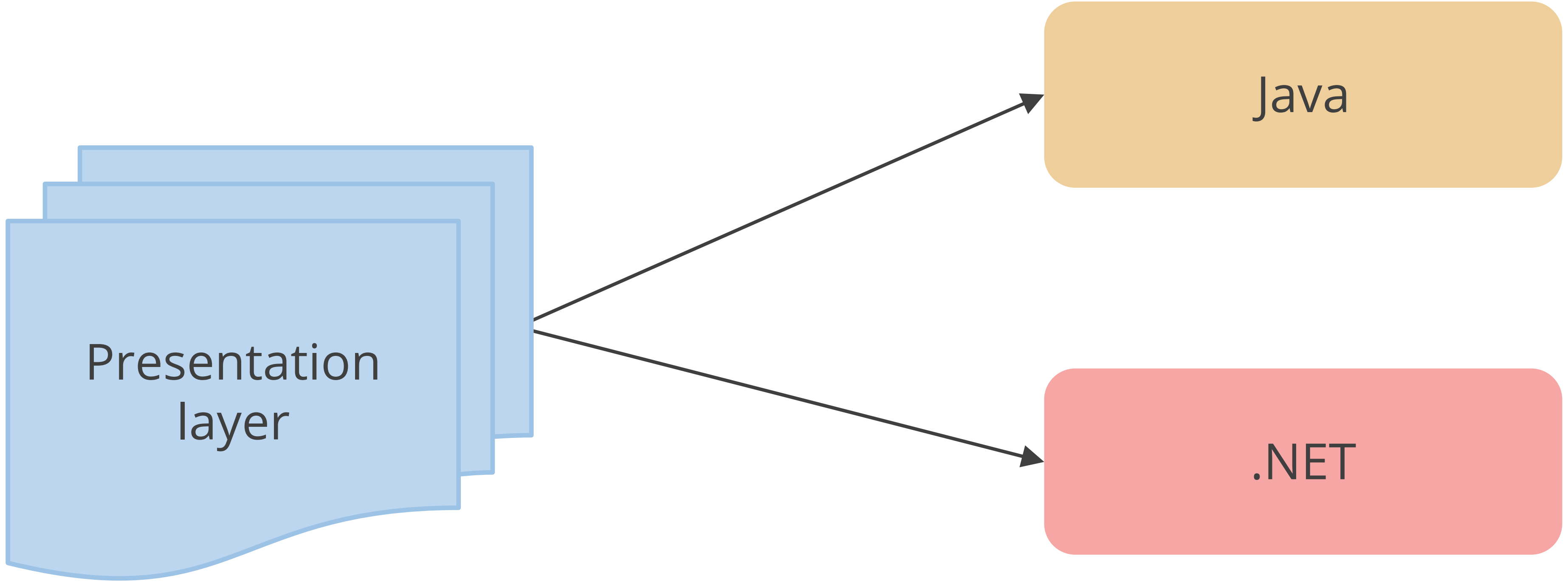
- ✓ Must be aware of the location of the Web Service
- ✓ Should understand what the Web Service performs

WSDL is used for calling the correct Web Service.

Working of Web Services

Working of Web Services: Example

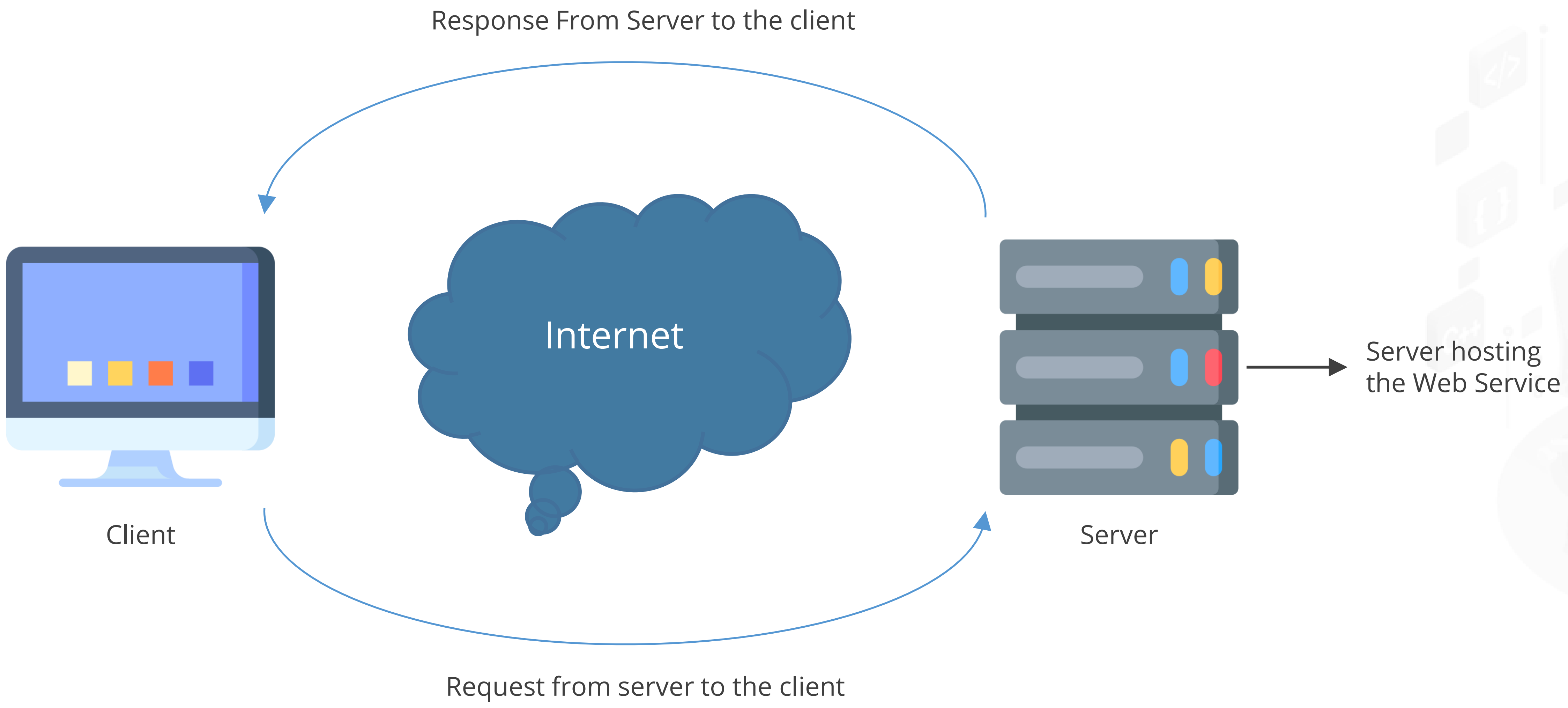
Amazon has a Web Service that lists the items available on amazon.com where the presentation layer is written in Java or .NET, and the Web Service can be communicated with the help of either programming language.



The exchanged data in XML between the server and client is crucial in Web Services.

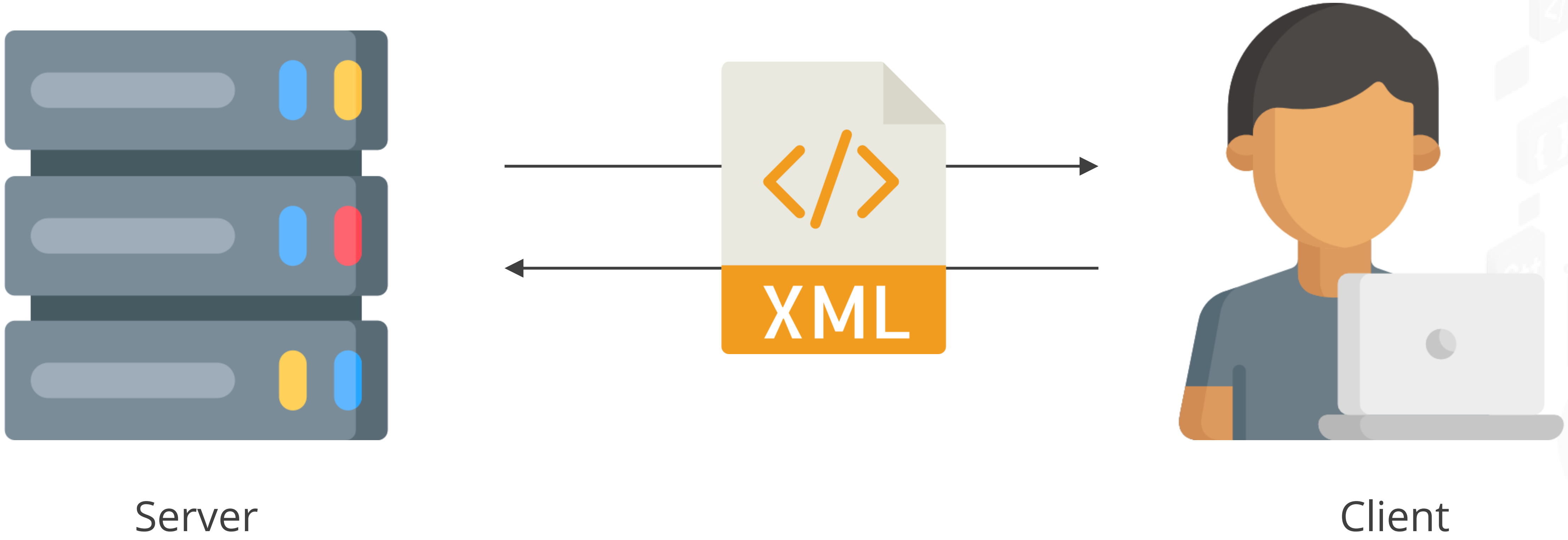
Working of Web Services: Example

The client uses a request to send a sequence of Web Services invoked to a server that helps to host the actual Web Service.


Response From Server to the client
Client
Internet
Server
Server hosting the Web Service
Request from server to the client

Working of Web Services: Example

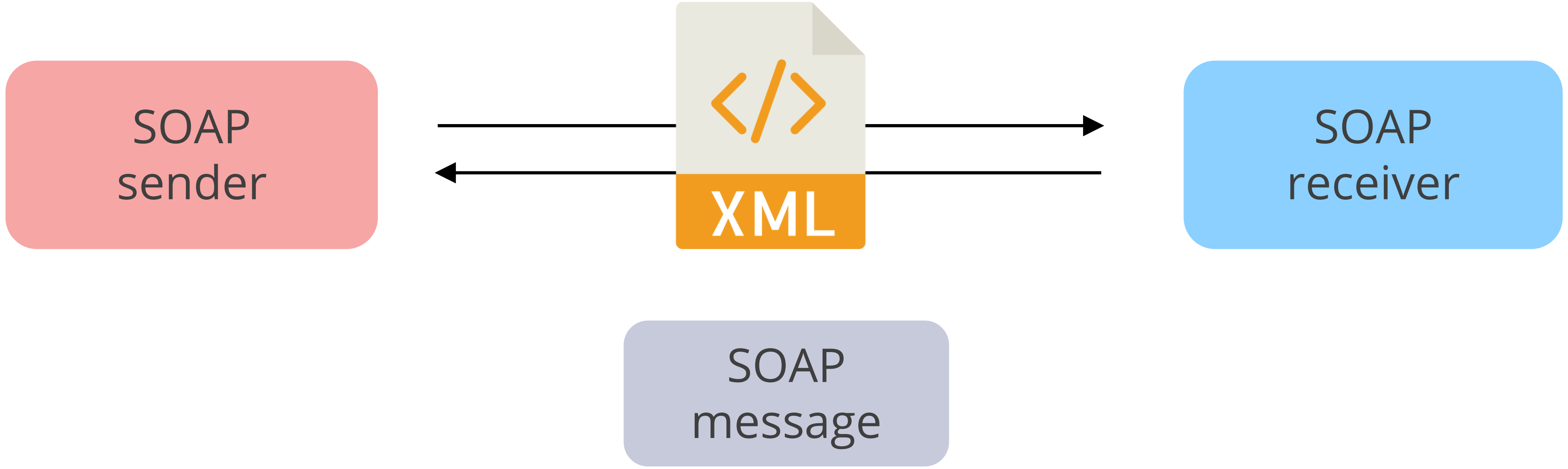
XML understands various programming languages. It is a counterpart of HTML.



SOAP is used to transmit XML data between applications.

Working of Web Services: Example

A SOAP Message is a piece of information sent from the Web Service to the application.



The client application that calls the Web Service is usually generated in any programming language because the content is in XML.

TECHNOLOGY

Security in Web Services

Security in Web Services

Security is critical to Web Services.

SOAP

XML-RPC

There are three specific security issues with Web Services.

Confidentiality

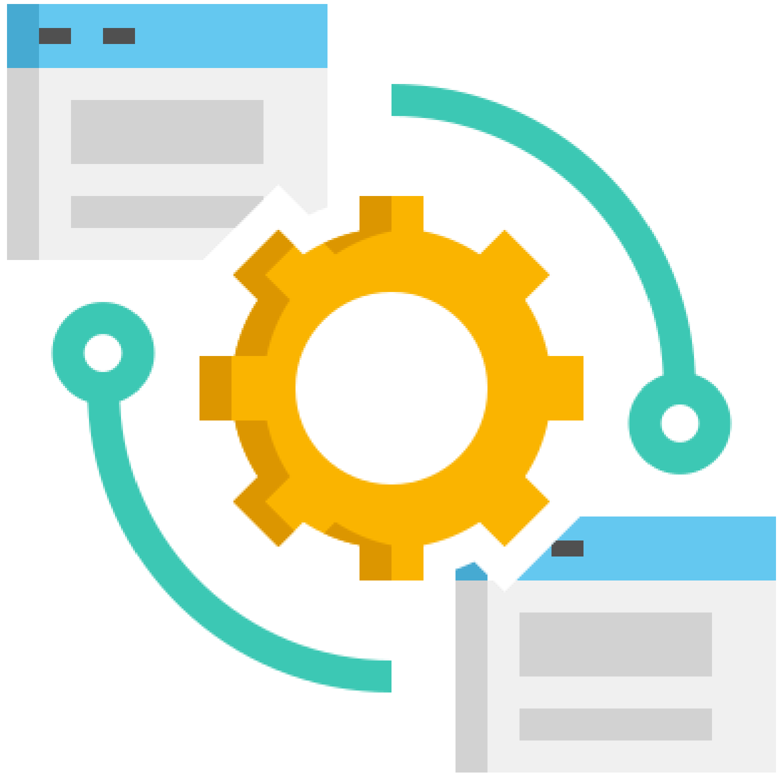
SOAP and XML-RPC run on top of HTTP.

Secure
Socket Layer

HTTP supports the Secure Socket Layer that helps in encrypting the communication.

Confidentiality

A single Web Service contains many applications.



For example, one large service may be combined with the services of three different applications.

Confidentiality

Here, the messages should be encrypted at every node along the service path, as the Secure Socket Layer is not adequate.

The W3C XML Encryption Standard is employed to decrypt and encrypt the required XML document.

Authentication

HTTP and SOAP are considered under authentication.

HTTP offers built-in support that helps with digest and basic authentication.

These services can be protected in the same way HTML documents are protected.

Authentication

SOAP’s digital signature uses public-key cryptography to sign SOAP Messages digitally.

It enables the client or server to validate their identity.

Network Security

There is no solution to this problem. For now, HTTP POST requests are employed for:

Filtering out all SOAP or XML-RPC messages

Setting their content type to text or XML

HTTP POST

Service-Oriented Architecture

Service-Oriented Architecture

It is a style of software design where services are provided to other components by application components.

Service-Oriented Architecture

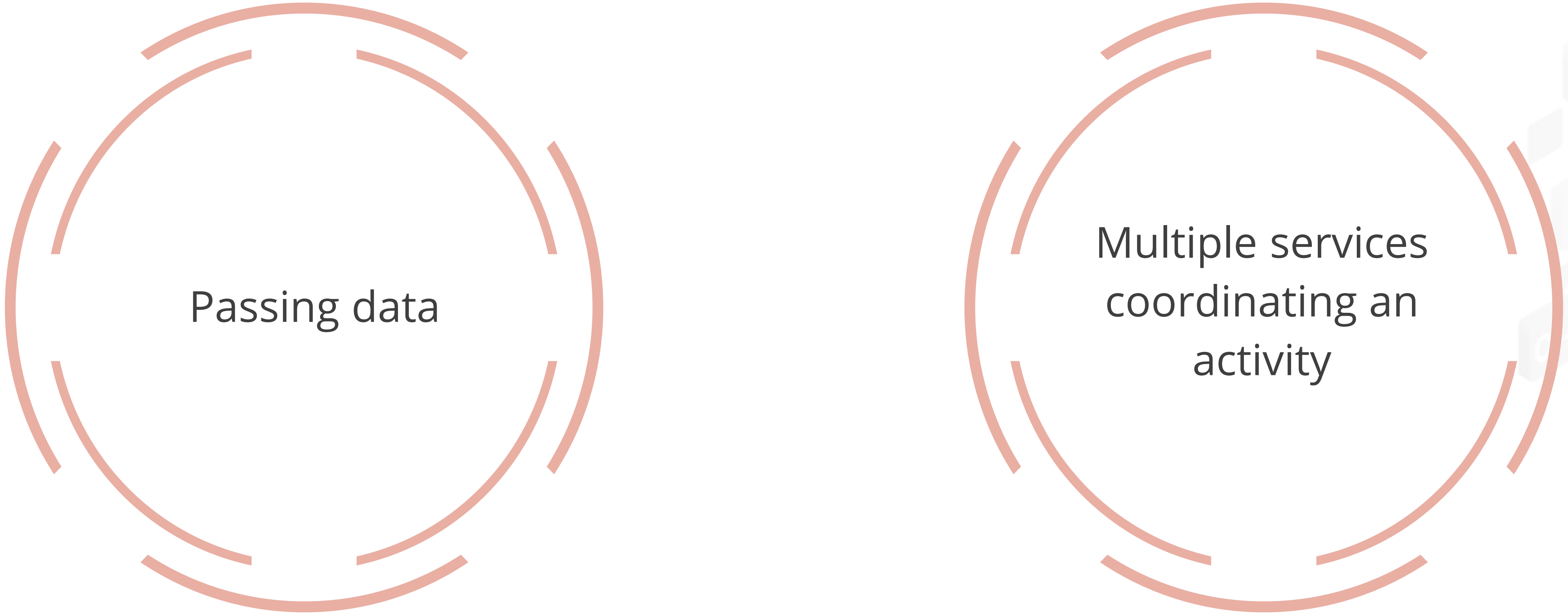
A few applications of SOA are:

1. Employed by armies and air forces for developing situational awareness systems
2. Used in games that use inbuilt functions for execution
3. Used to maintain museums with a virtualized storage pool
4. Used to improve healthcare delivery

Service-Oriented Architecture

Its principles are not dependent on vendors or other technologies.

In SOA, several services communicate with each other in two ways. They are through:



Passing data

Multiple services coordinating an activity

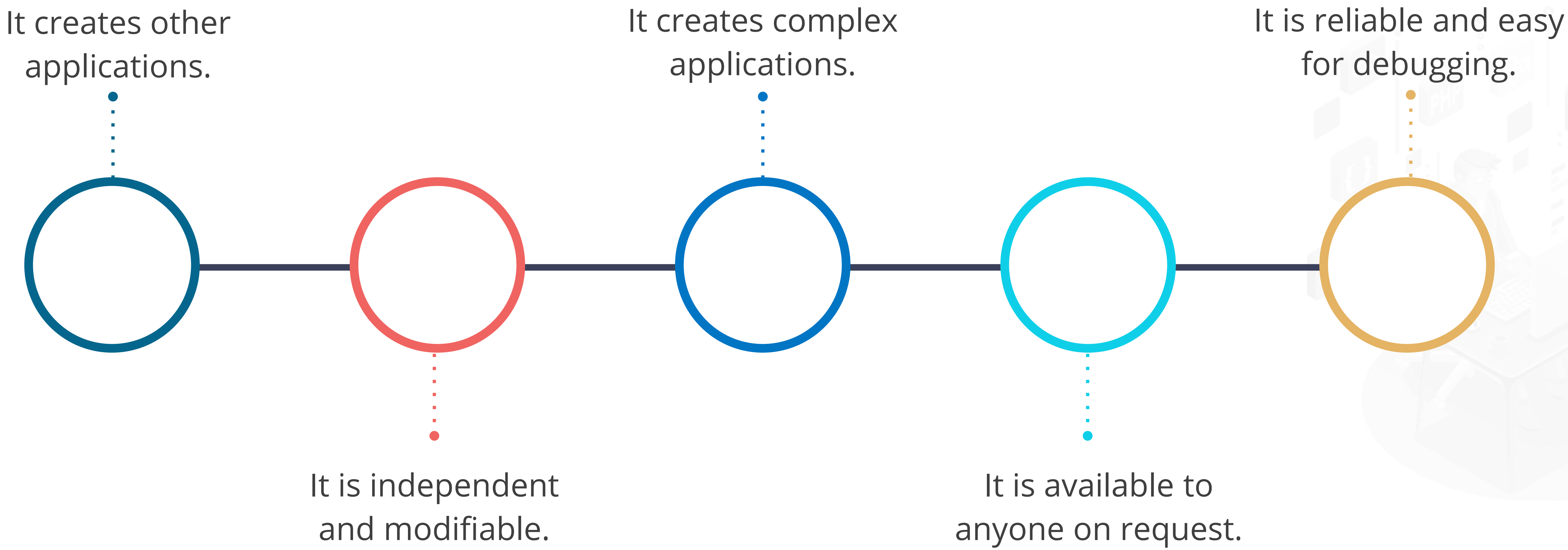
Service-Oriented Architecture

SOA is simply a stage in application development or integration, or both.

It makes software components reusable with the help of interfaces.

It is an architectural approach where the application uses the services available on the network.

Service-Oriented Architecture



It creates other applications.

It is independent and modifiable.

It creates complex applications.

It is available to anyone on request.

It is reliable and easy for debugging.

SOA: Roles

There are two major roles within a service-oriented architecture.

Service Provider

It maintains the organization and service that makes them available for users.

Service Consumer

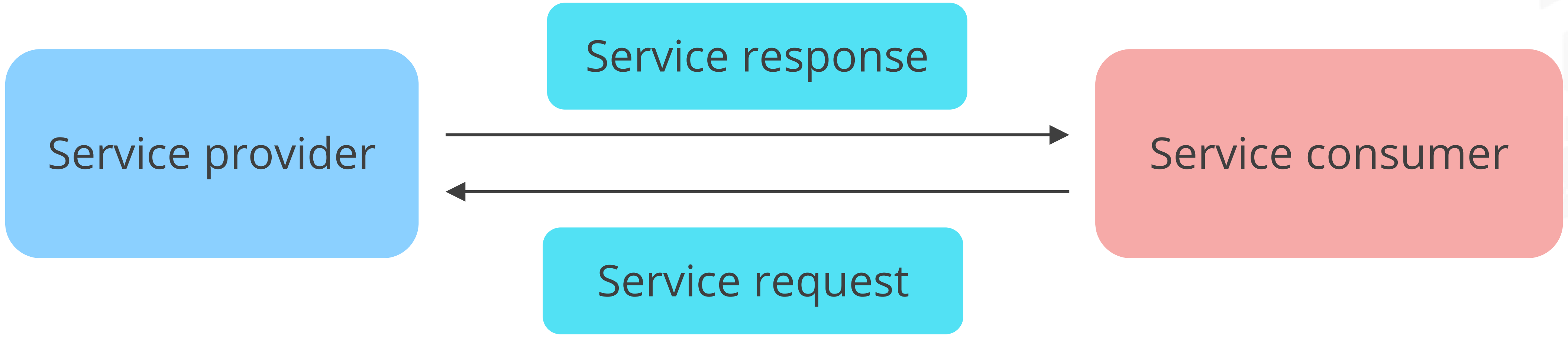
It can locate the service metadata in the registry and create the customer components needed to bind and use the service.

Service Provider

The service provider can publish advertising services in a registry with a service contract.

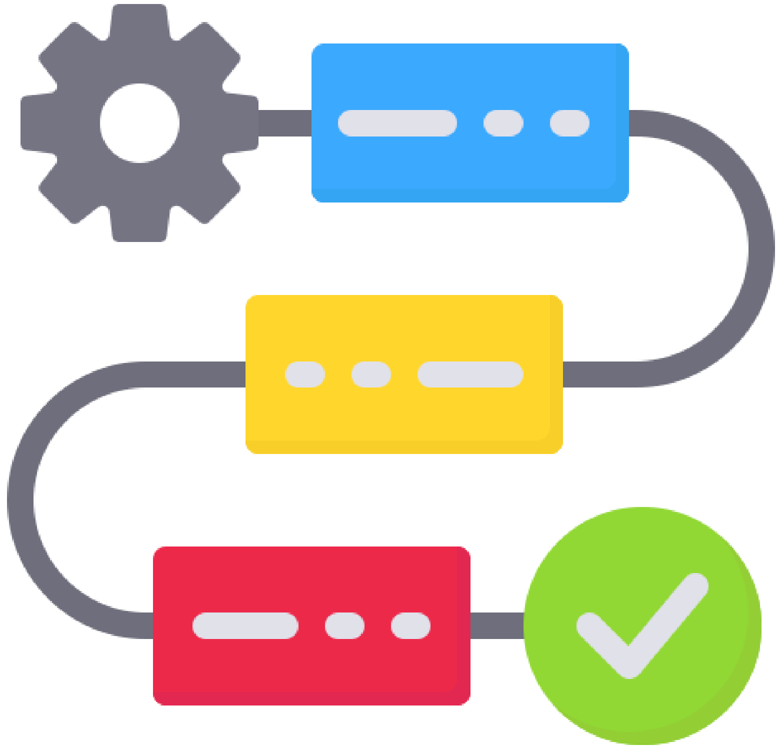
Services

Services may also collect information and data retrieved from other services.



Services

It is also employed to generate workflows of services to satisfy a user's request.



Orchestration

Service choreography helps to coordinate the interaction of services without any single point of control.

TECHNOLOGY

Components of SOA Architecture

SOA Architecture

SOA architecture comprises several layers that include:

Consumer Layer

Helps the user and user interface to interact

Business Process Layer

Stores business applications

Services

Runs in the services layer

SOA Architecture

SOA architecture comprises several layers that include:

Service Component Layer

Stores the technical and functional applications

Operational System Layer

Contains the data model in the system

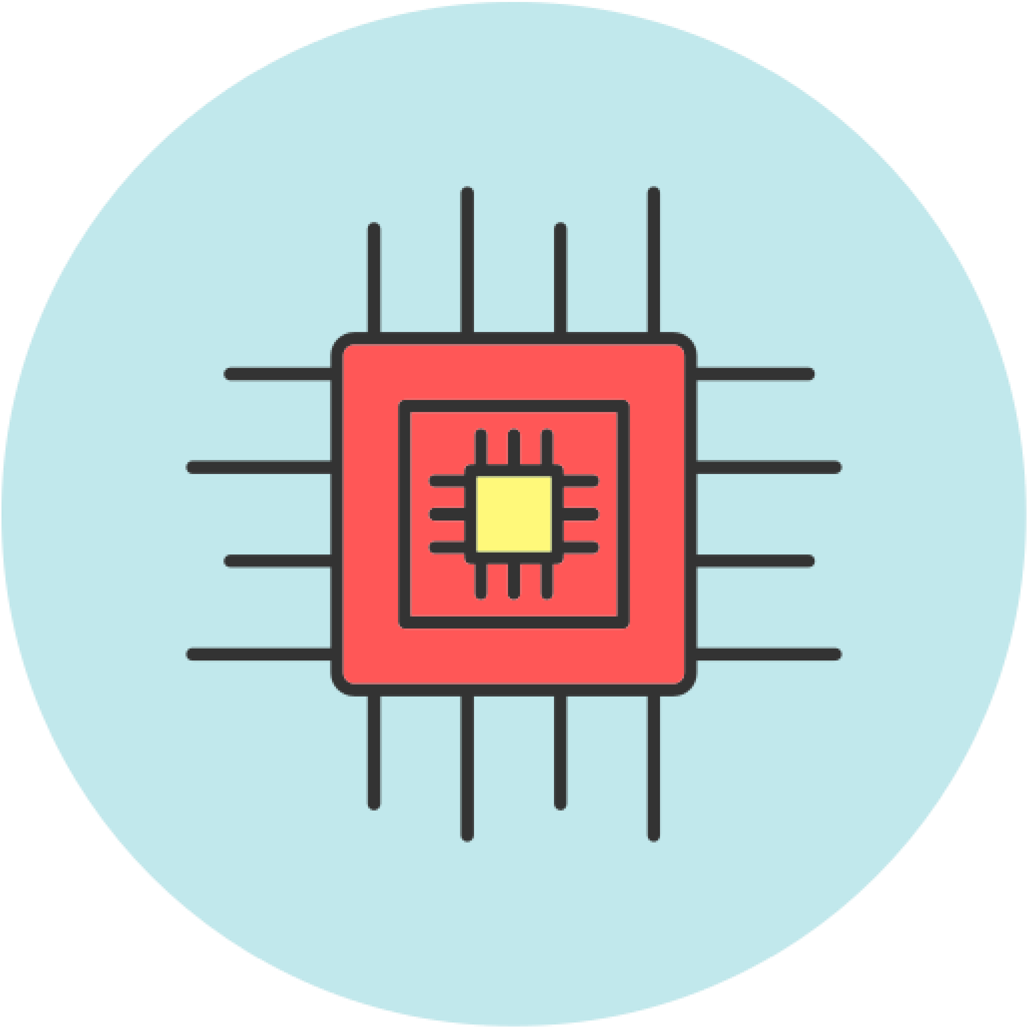
Features of SOA Architecture

SOA services allow the processing of raw materials by the user.

It helps pass on the information to the next layer for proper service implementation.

Features of SOA Architecture

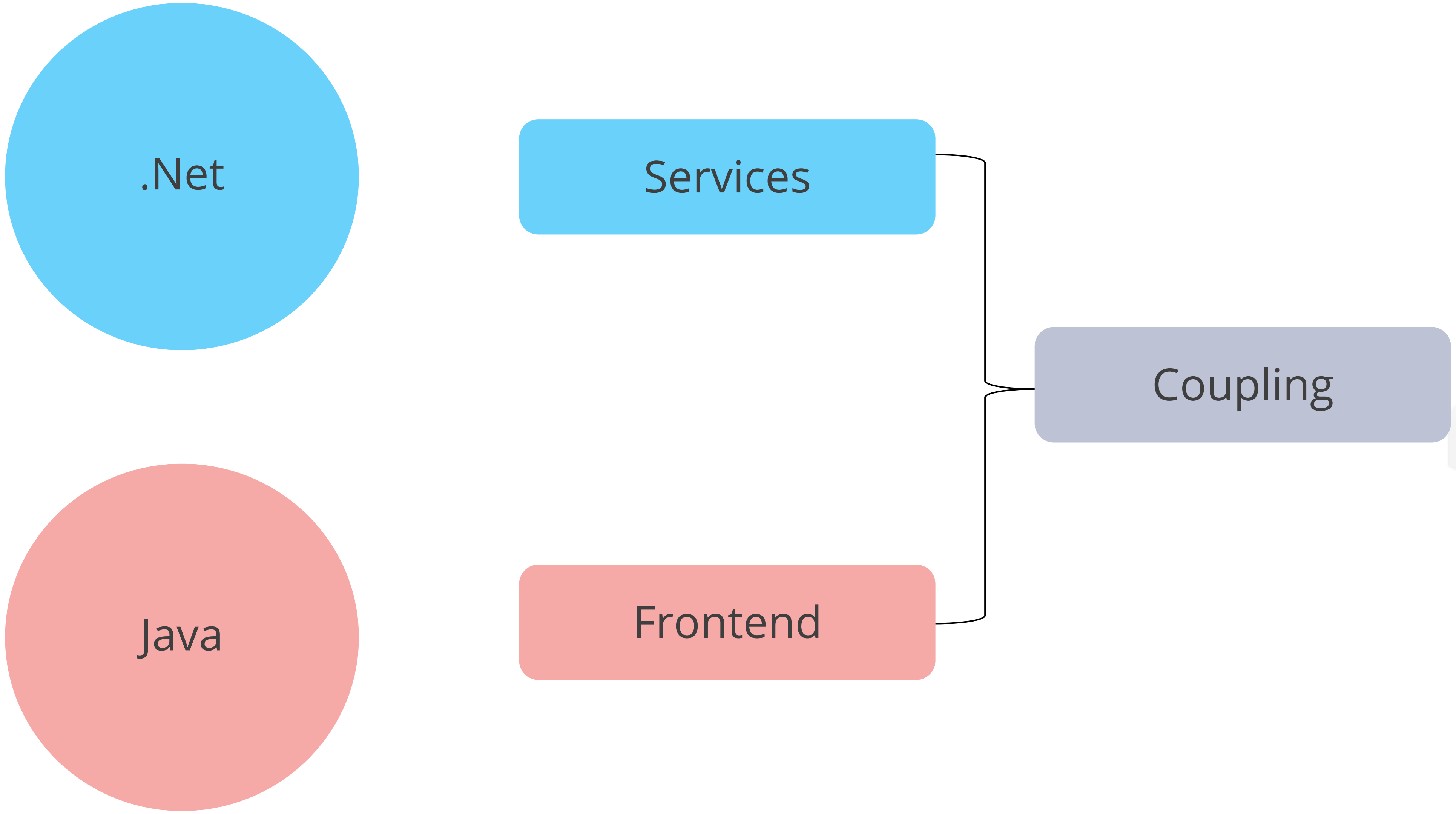
The process layer is responsible for managing the different parts of SOA.



New services and applications are handled using the services in other forms.

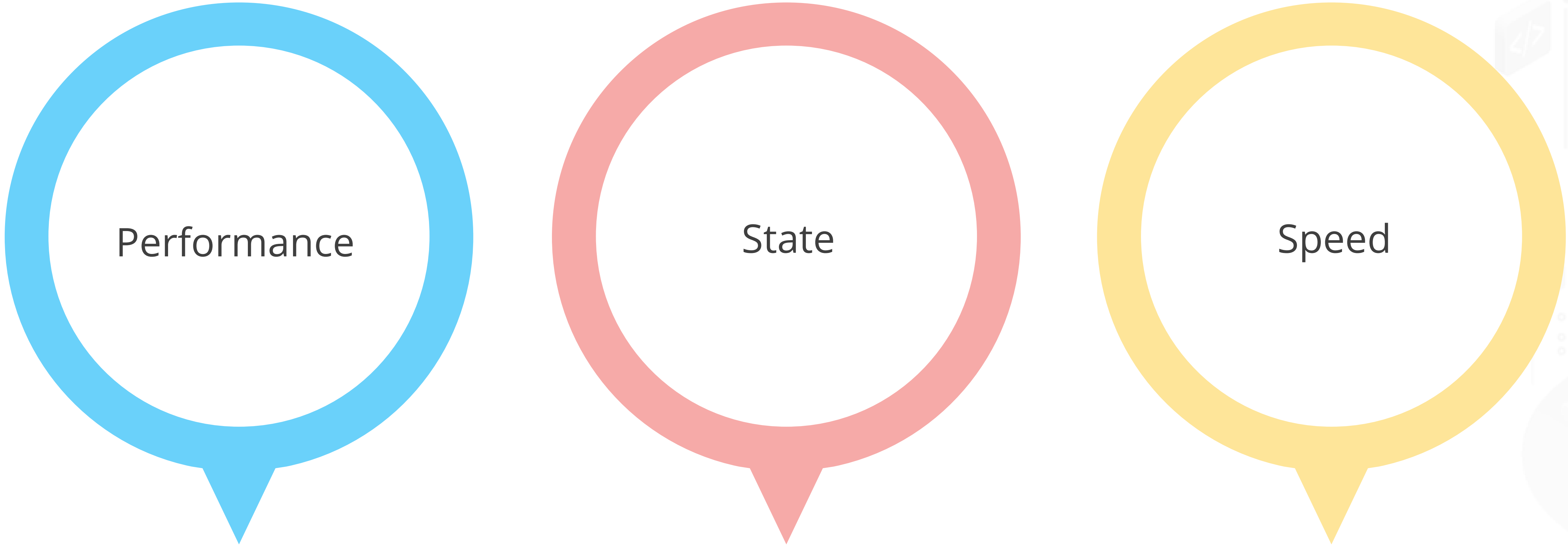
Features of SOA Architecture

A framework is used in the SOA, which binds the business logic and interface.



Features of SOA Architecture

The processes are monitored continuously for:



Performance

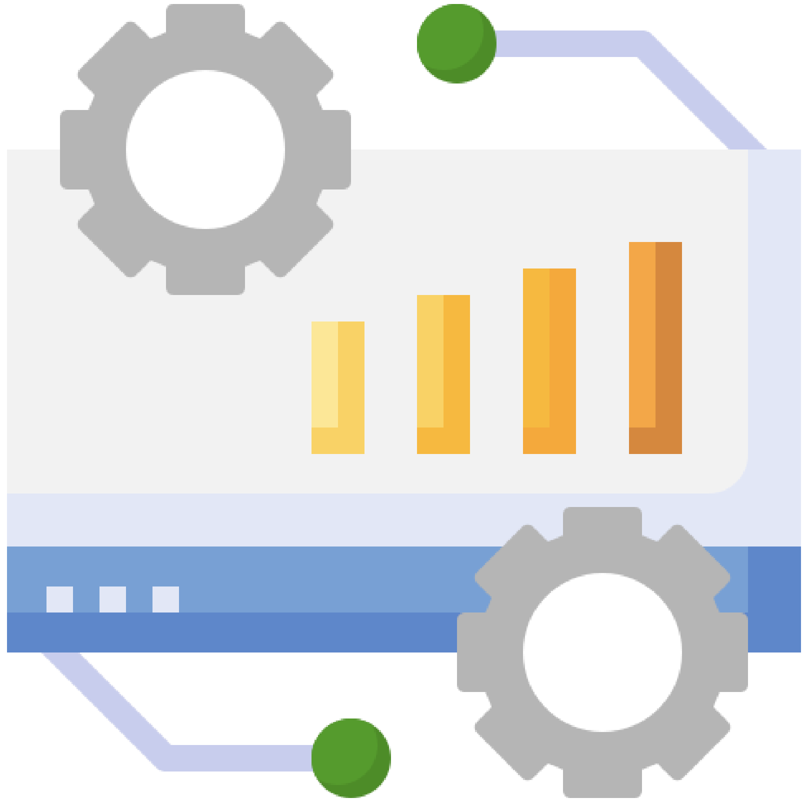
State

Speed

Features of SOA Architecture

Business activity monitoring is enabled in the system.

This helps to generate a report and a dashboard to know the process.



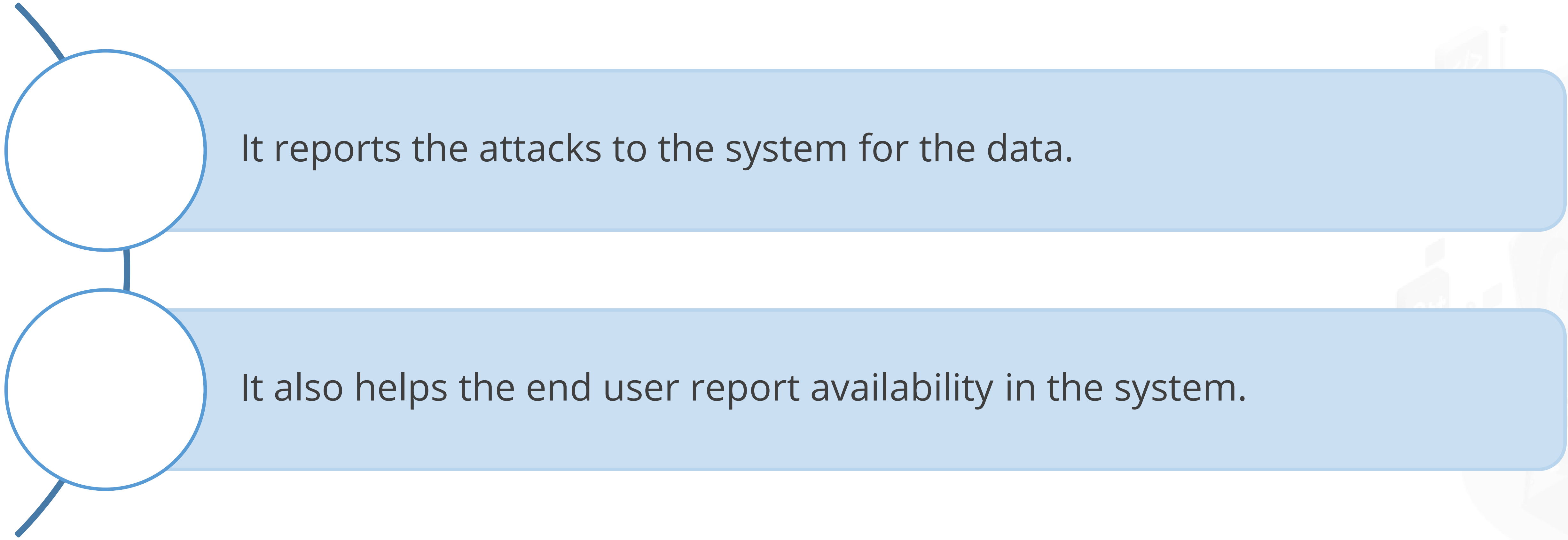
Features of SOA Architecture

SOA contains an operational data store to give all the data a synchronized view from the client's perspective.

This helps gather information about the business in the long run.

Features of SOA Architecture

Business intelligence is used in reporting the instances of data services in ODS.



It reports the attacks to the system for the data.

It also helps the end user report availability in the system.

Features of SOA Architecture

Security for the SOA is uncompromised due to the data it takes.

Security is provided in all the components of the architecture.

Features of SOA Architecture

All the components in the architecture are loosely arranged in the structure.

In the absence of management, operators in the architecture are used to monitor the system's performance and status.

TECHNOLOGY

SOAP-Based Web Services

SOAP

SOAP stands for Simple Object Access Protocol.

- ✓ It is based on the XML protocol to access web services over HTTP.
- ✓ It lays out a few specifications that are used across applications.

SOAP

SOAP defines how Web Services talk to client applications that invoke them.

It is built as an intermediate language, which helps to:

- ✓ Develop the applications in different programming languages
- ✓ Remove the extreme development effort

SOAP

XML is employed as the intermediate language.

SOAP is designed to work with XML over HTTP.

SOAP contains several specifications that are used across all applications.

Advantages of SOAP

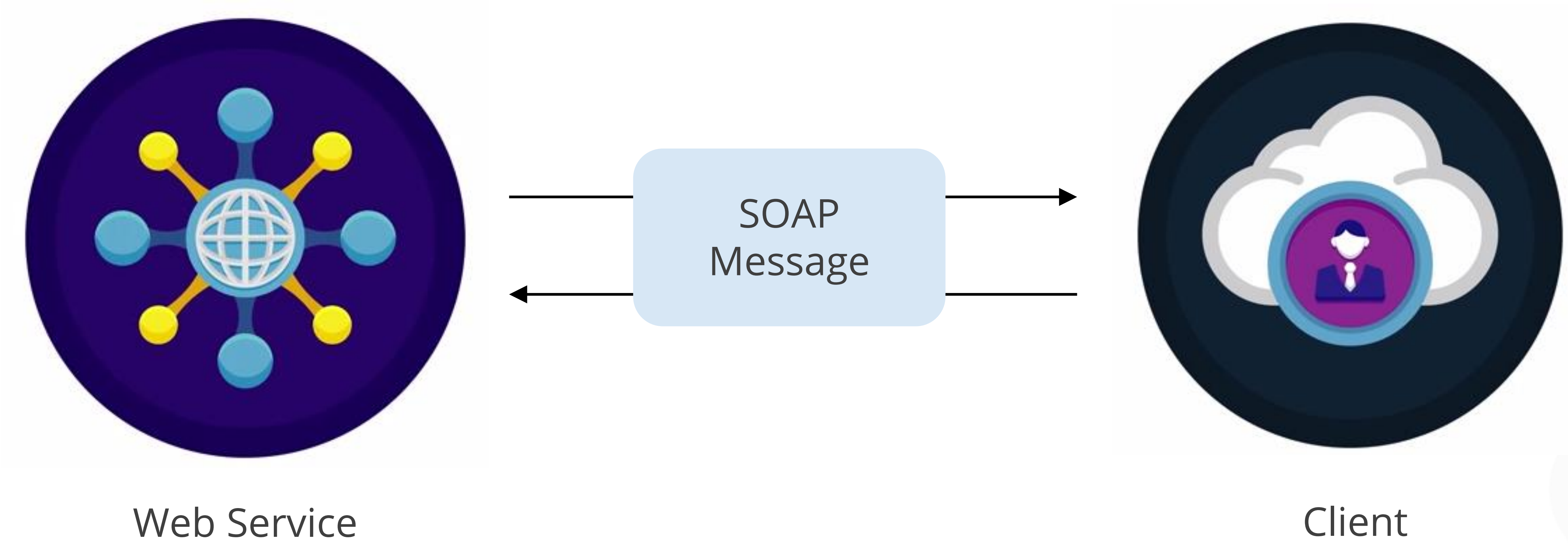
The following are the advantages of SOAP:

1. SOAP is recommended by the W3C consortium, which is the main body for web standards.
2. SOAP is a lightweight protocol that allows the exchange of data between applications.
3. SOAP protocol works with all programming language-based applications on Linux and Windows platforms.
4. SOAP also works on the HTTP protocol; HTTP is the default protocol used by all web applications.

SOAP Building Blocks

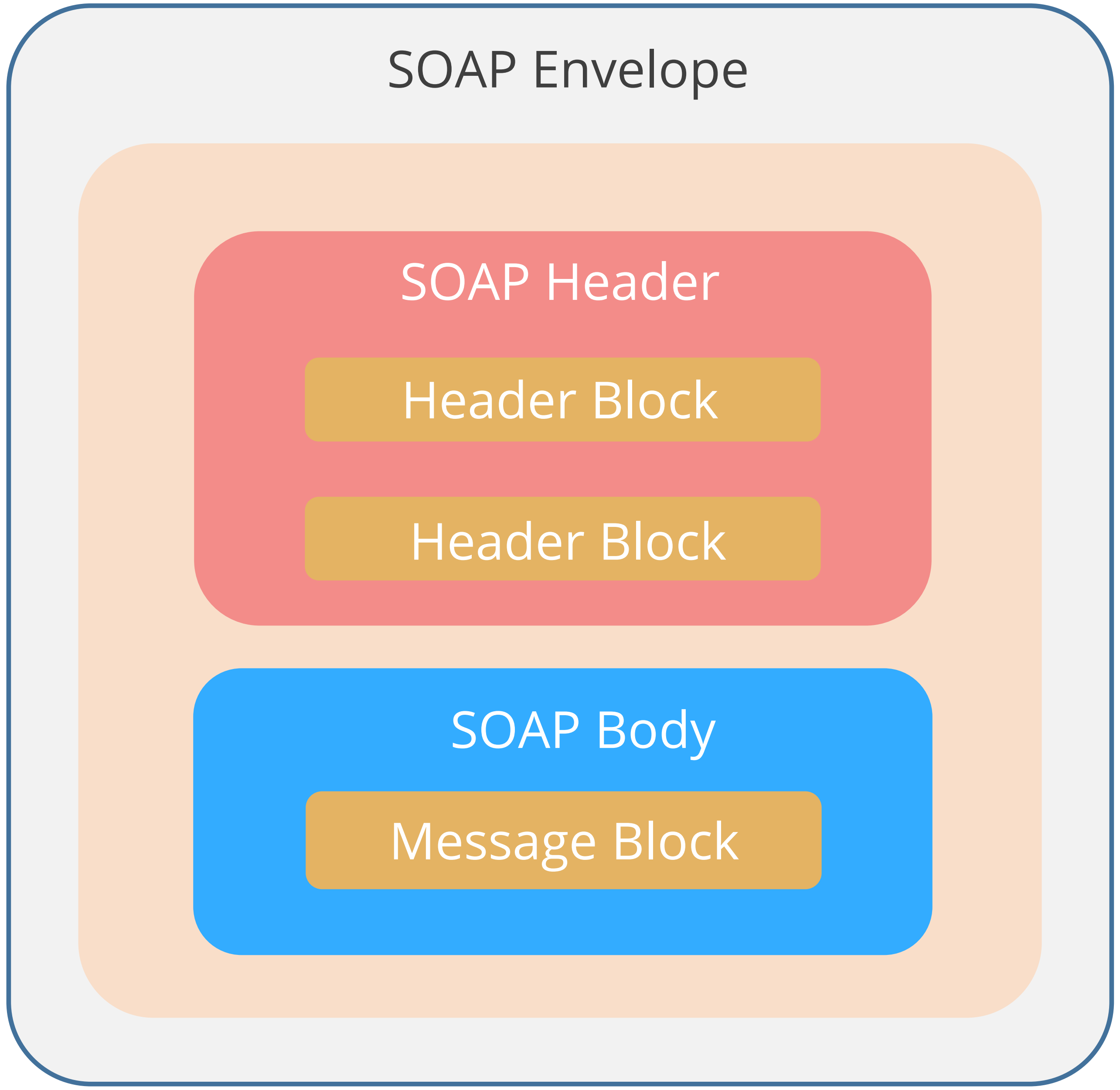
SOAP

SOAP defines a **SOAP Message** which is sent to the Web Service and the client application.



SOAP Building Blocks

The SOAP with different building blocks for SOAP Messages is shown below:



SOAP Building Blocks

The SOAP Message is a type of XML document that contains a few components.

Envelope

It helps to identify the XML document as a SOAP Message.

It is referred to as the containing part of the SOAP Message because it encapsulates all the details in the SOAP Message.

It is the root component of the SOAP Message.

SOAP Building Blocks

The SOAP Message is a type of XML document that contains a few components.

- It includes information that can be used by the calling application.

- It contains the definition of difficult types that can be helped by the SOAP Message.

SOAP Building Blocks

The SOAP Message is a type of XML document that contains a few components.

Body

It contains call and response information and the actual data that needs to be sent between the calling application and the Web Service.

SOAP Building Blocks: Example

Below is the syntax of SOAP building block:

```
<xsd:complexType>
 <xsd:sequence>
      <xsd:element name="value" type="value_type"/>
      <xsd:element name="value"  type="value_type"/>
  </xsd:sequence>
</xsd:complexType>
```

SOAP Building Blocks: Example

To send a structured data type containing a fusion of **Student Name** and **Student Address**, the complex type is:

```
<xsd:complexType>
 <xsd:sequence>
      <xsd:element name="StudentName" type="string"/>
      <xsd:element name="StudentAddress"
type="string"/>
  </xsd:sequence>
</xsd:complexType>
```

SOAP Building Blocks: Example

The SOAP body is:

```
<soap:Body>
   <GetStudentInfo>
            <StudentName>John</StudentName>
            <StudentAddress>2144 ABC
Road</StudentAddress>
   </GetStudentInfo>
</soap:Body>
```

SOAP Communication Model

HTTP Protocol

The HTTP protocol is employed for all communication in SOAP.

Web Services use the Remote Procedure Call Style for communication, which has a lot of limitations.

Web Service

Consider a case where the server is hosting a Web Service that gives two methods:

getStudent:

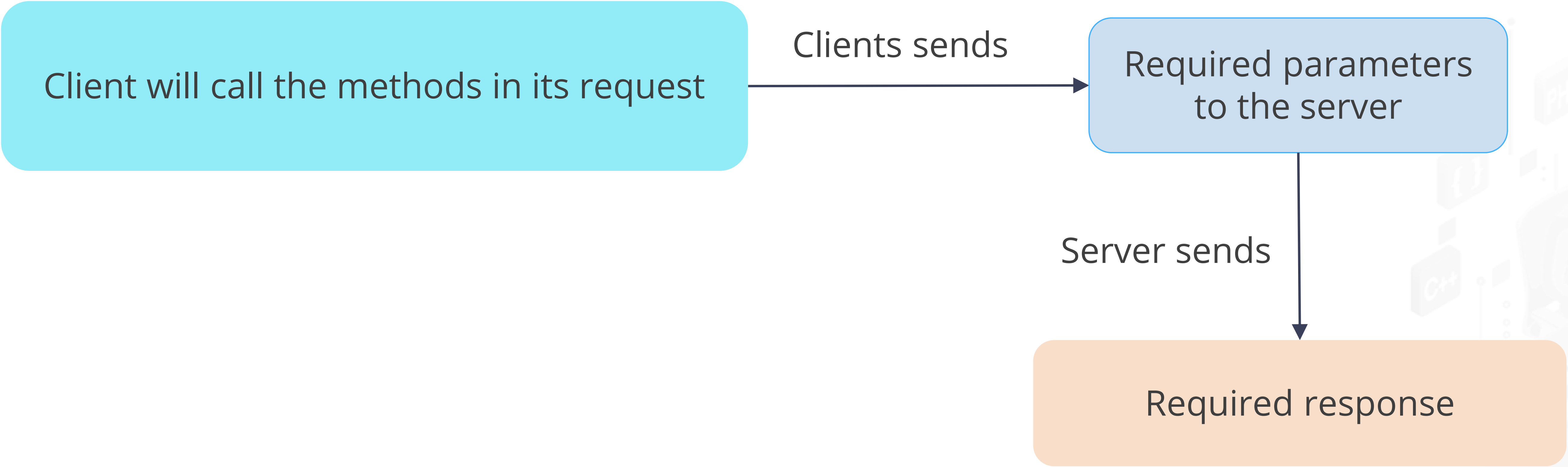
This will get details of the student.

setStudent:

This will set the value of the details of the student, like name, address, and so on.

Remote Procedure Call

RPC style of communication:



Client will call the methods in its request

Clients sends

Required parameters to the server

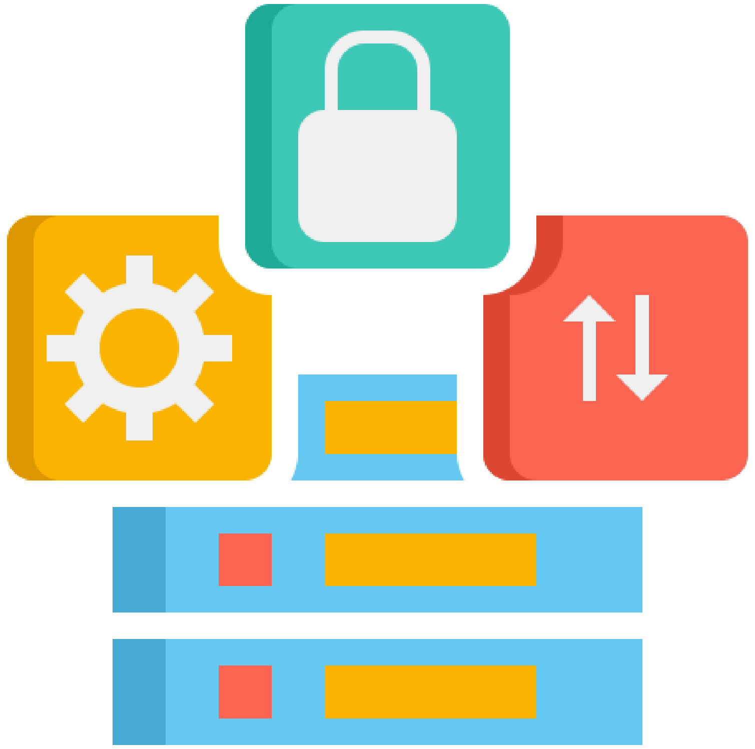
Server sends

Required response

Limitations of RPC

In RPC, the call is not taken out through standard protocol when a call is made to the remote procedure.

Methods hosted by the server and the calls should be in a specific programming language.



Separate ports are required for opening the server to allow communication with the server.

Limitations of RPC

Employ the SOAP model to overcome the limitations

The following are the two processes:

(Client) Marshalling

It is a process in which the client edits the information on the procedure call and arguments into a SOAP Message that is sent to the server as an HTTP request.

(Server) Demarshalling

In this process, the server opens the message from the client, evaluates the client's requests, and sends the required response to the client as a SOAP Message.

TECHNOLOGY

Restful Web Services

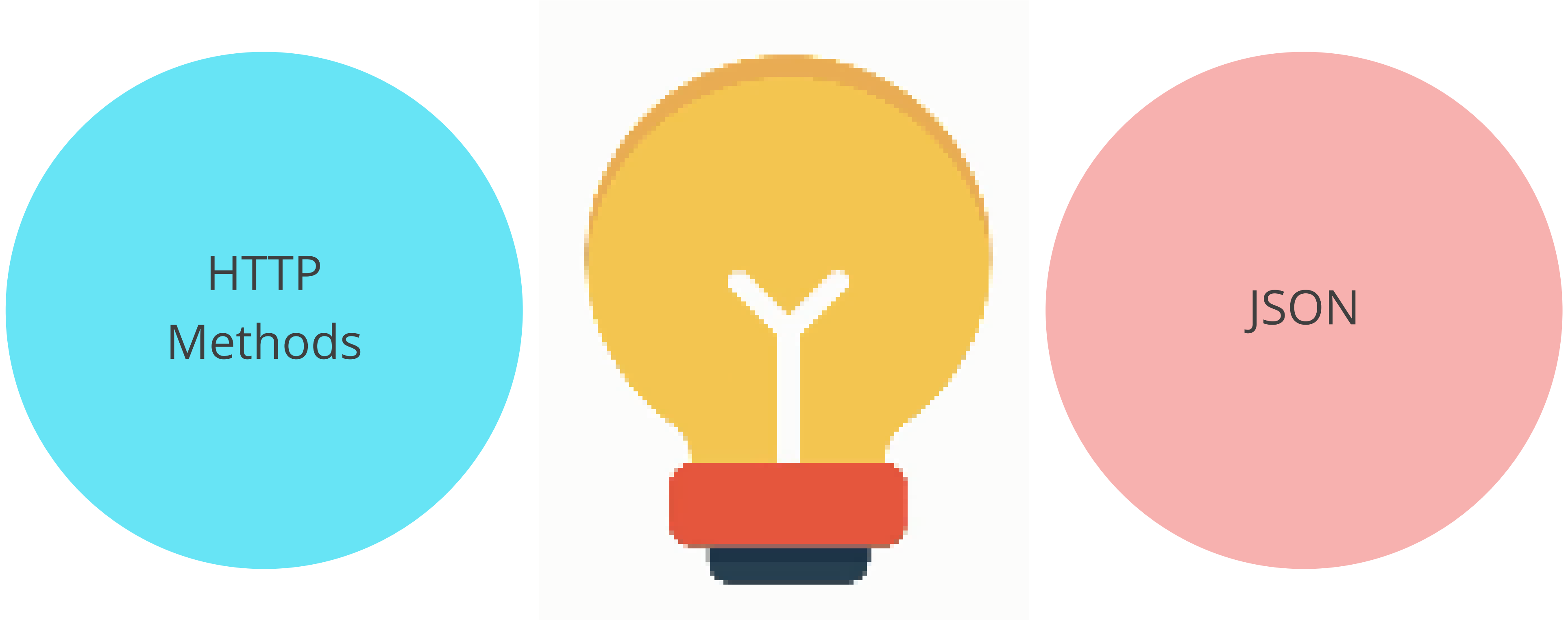
Web Services

Web Services based on the REST architecture are called RESTful Web Services.

Employ the HTTP methods for implementing the concept of REST architecture.

URL

It is employed to define a URL as a service, giving resource representations like a set of HTTP methods and JSON.



HTTP Methods

JSON

REST

REST stands for Representational State Transfer, which uses the HTTP protocol and is a web standards-based architecture.

- ✓ Each component is a resource accessed by a common interface with the help of HTTP standard methods.
- ✓ REST client accesses and updates the resources.

All resources are identified by global IDs or URIs.

REST

REST employs a lot of representations to represent a resource, like JSON, text, and XML.

The rest-based architecture uses four HTTP methods:

GET

It gives read-only access to a resource.

POST

It is used to create a new resource.

DELETE

It is used to delete a resource.

PUT

It is used for updating an existing resource or for creating a new one.

TECHNOLOGY

REST Resources

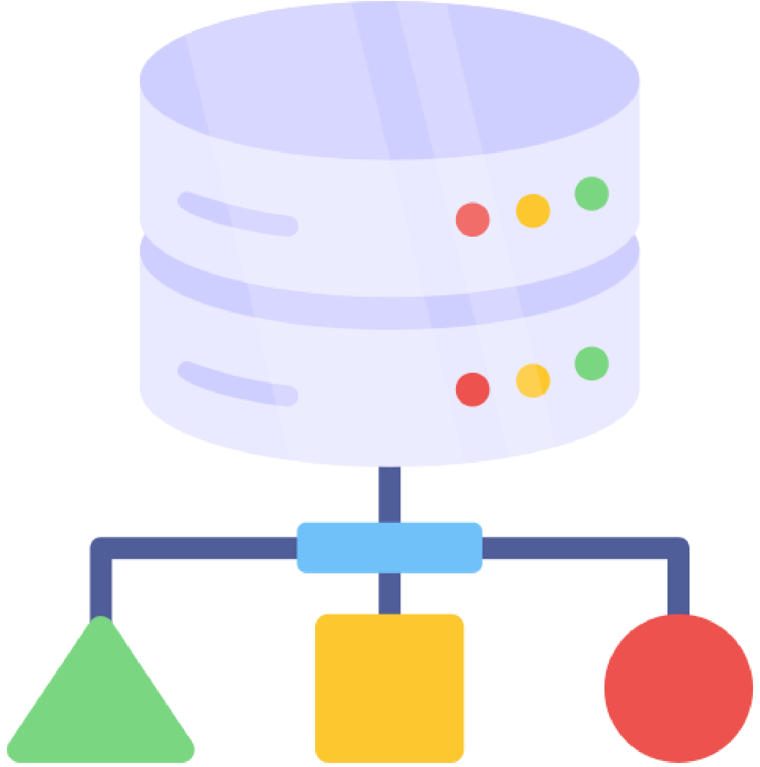
REST Architecture

In REST architecture, each content is treated as a resource.

These resources can include HTML pages, videos, text files, or dynamic business data.

REST Server

It provides resource access to the client that accesses and modifies the resources.



The resource is like an object in OOP or an entity in a database.

REST Server

Once a resource is identified, its representation can be determined using a standard format.

The server can send the resource in the same format the client understands.

REST Server

For example:

A student is a resource that is represented with the help of the XML format, as shown:

```
<student>
   <id>1</id>
   <name>John</name>
   <department>Physics</department>
</student>
```

In JSON format:

```
{
   "id":1,
   "name":"John",
   "department":"Physics"
}
```

REST Server: Features

Features of good resource representation are:

Completeness

The format should represent the resource completely. The format must be able to represent simple as well as complex structures.

Understandability

The client and server must understand and use the represented format of the resource.

Linkability

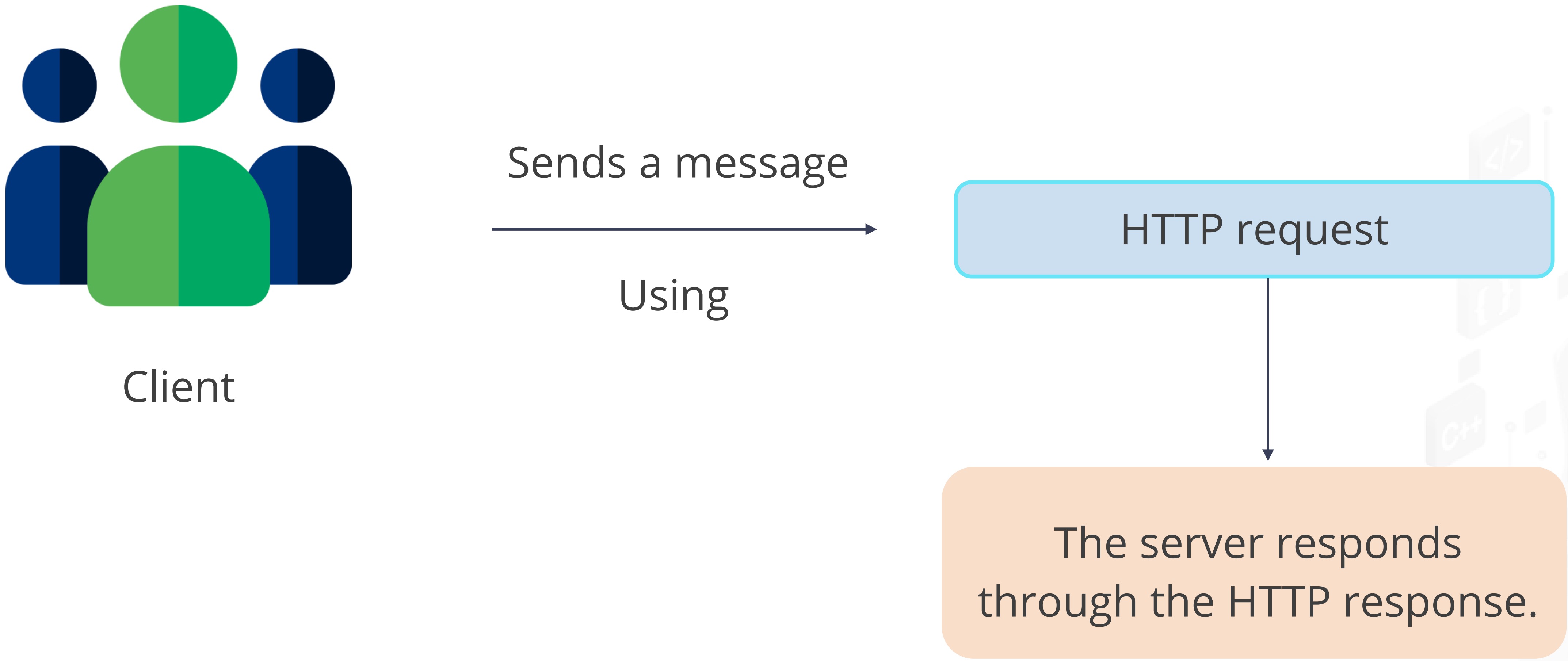
If a resource can have a link to another resource, then the format must be capable of handling these types of situations.

TECHNOLOGY

REST Message

HTTP Protocol

HTTP protocols are employed by RESTful Web Services for communication between the server and the client.



Messaging contains message data and metadata, that is, information regarding the message.

HTTP Request

There are five major parts to the HTTP Request, which include:

Parts	Description
Verb	This includes HTTP methods such as POST, GET, DELETE, PUT, and so on.
URI	This is used to identify the resource on the server.
HTTP Version	This indicates the HTTP version.
Request Header	This contains metadata for the HTTP request message in the form of key-value pairs.
Request Body	This part represents the resource representation or message content.

HTTP Response

There are four major parts to an HTTP response, which include:

Status

This part indicates the status of the server for the requested resource.

HTTP Version

This part indicates the HTTP version.

Response Header

This part contains the metadata for the HTTP Response.

Response Body

This message represents resource representation or message content.

TECHNOLOGY

REST Addressing

REST Addressing

Addressing refers to the process of locating single or multiple resources lying on the server.

REST Addressing

Resources in the REST architecture are identified through URIs.

The syntax:

```
 <protocol>://<service-
name>/<ResourceType>/<ResourceID>
```

REST Addressing

A URI is employed to locate a resource on the server hosting the Web Services.

Example:

http://localhost:8080/StudentManagement/rest/StudetService/students/1

An important term for the request is VERB, which identifies the operation to be done on the resource.

Construction of URL

Crucial points during the construction of URL are:

Avoid using spaces

Use a hyphen (-) or underscore(_) when using a long resource name

Use plural noun

Use plural nouns for defining resources

Use lowercase letters

Use the URI in lowercase, as it is case-insensitive

Construction of URL

Crucial points during the construction of URL are:

Use HTTP verb

Use HTTP verbs like PUT, GET, and DELETE for doing operations on the resource

Maintain backward compatibility

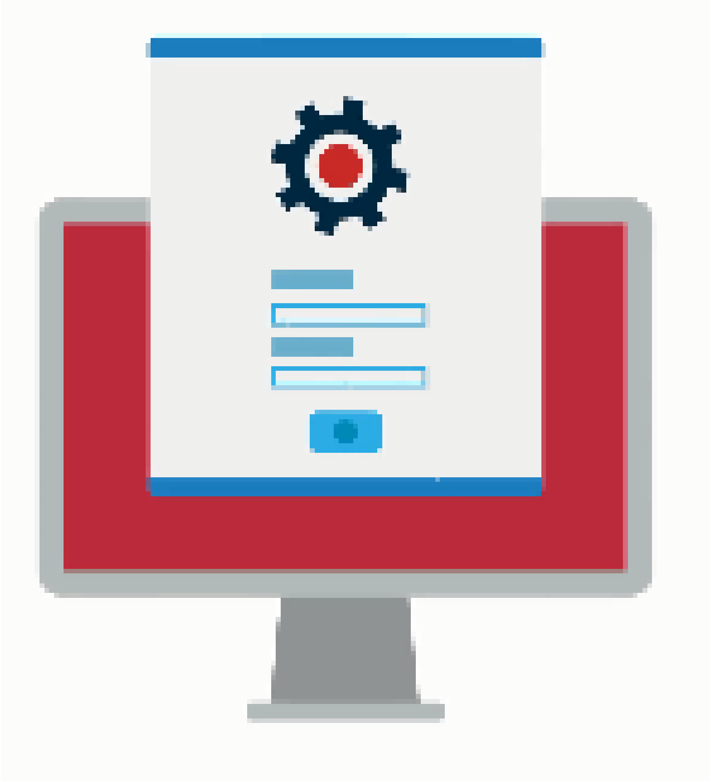
Use the URL as a public service, as it must always be available

TECHNOLOGY

REST Statelessness

REST Statelessness

RESTful Web Service must not keep a client safe on the server, as per the REST architecture, and it is called Statelessness.



The client must pass its context to their server, and then the server can store this context for processing the client’s request.

Web Services

Web Services stick to this restriction, and methods of these Web Services do not store any information from the client they are invoked from.

An example is shown below:

```
https://localhost:8080/StudentManagement/rest/StudentService/students/1
```

Web Services

The result will always be the Student XML, whose ID is 1.

Here, the server will not store any client information:

```
<student>
   <id>1</id>
   <name>John</name>
   <department>Physics</department>
</student>
```

Advantages of Statelessness

The advantages of Statelessness are as follows:

1 There is no need to maintain the client’s previous interactions.

2 The client gets a simplified application design.

3 The Web Services can treat every method request independently.

4 RESTful Web Services can work with HTTP protocols easily.

TECHNOLOGY

REST Caching

REST Caching

Caching refers to the process of storing the server response on the client.

The client does not have to place server requests for the same resource repeatedly.

Note

The server response must have information on how catching should happen, which helps the client cache the response for a particular period or never cache the server response.

Headers

Headers for which a server response includes a few catches are:

Date

Time and date of the resource of its creation

Last modified

Time and date of the resource of its last modification

Age

Duration in seconds from the fetching of resources from the server

Headers

Headers which a server response includes a few catches are:

Expires

Expiration of time and date of caching

Cache-control

The primary header for controlling string containing these directives

Public

Directive indicating that the resource is cacheable by any of the components

Headers

Private

Derivative indicating that the resource is cacheable only by the server and client

Max-age

Derivative indicating that caching is valid up to max-age in seconds. After the max-age, the client has to make another request

No-cache

Directive indicating that the resource is not cacheable

Must-revalidate

Derivative indicating the server for revalidating the resource if max-age has passed

TECHNOLOGY

REST Security

RESTful Web Services

RESTful Web Services work with HTTP URL paths. The security of RESTful Web Services is very important and cannot be ignored.

RESTful Web Services

Practices to be followed while designing a RESTful Web Service:

Session-based authentication

Uses session-based authentication for authenticating a user whenever a request is made to a method of Web Service

Validation

Validates all the inputs on the server

No sensitive data in the URL

Suggests avoiding using a username, password, or any other important information in the URL

RESTful Web Services

Practices to be followed while designing a RESTful Web Service:

Restriction on method execution

Supports restricted use of methods like POST and GET

Throw generic error messages

Suggests that Web Services must use HTTP error messages with the HTTP status codes

HTTP Status Code

Frequently used HTTP status codes are:

200 - OK:
It means success.

400 - BAD REQUEST:
It indicates that there is invalid input.

304 - NOT MODIFIED:
It is used to reduce network bandwidth usage in the case of conditional GET requests.

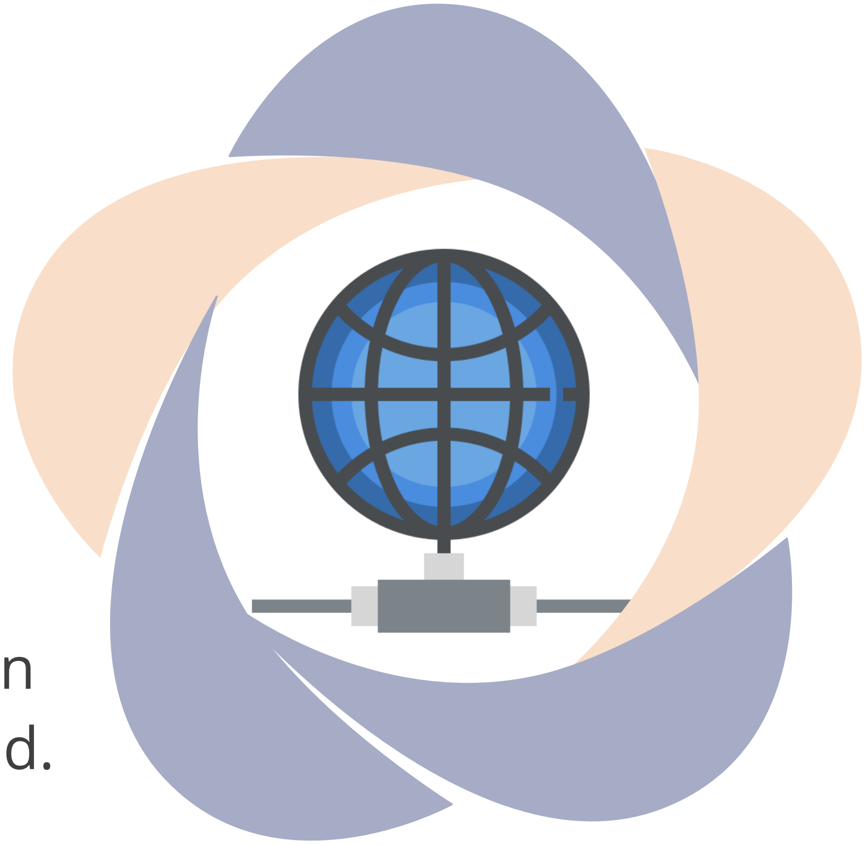
201 - CREATED:
The resource is successfully created with the help of a PUT or POST request.

204 - NO CONTENT:
It appears when the response body is empty.

HTTP Status Code

Frequently used HTTP status codes are:

401 - UNAUTHORIZED:
It indicates that the user is using wrong authentication token.



500 - INTERNAL SERVER ERROR:
It indicates that the server has thrown some exceptions.

403 - FORBIDDEN:
It indicates that the user is not allowed to use this method.

409 - CONFLICT:
It indicates the conflicting situation during the execution of the method.

404 - NOT FOUND:
It states that the method is not available.

Develop a SOAP Web Service

Problem Statement:

You have been asked to develop a SOAP web service for user details through Java objects.

ASSISTED PRACTICE

Assisted Practice: Guidelines

Steps to be followed are:

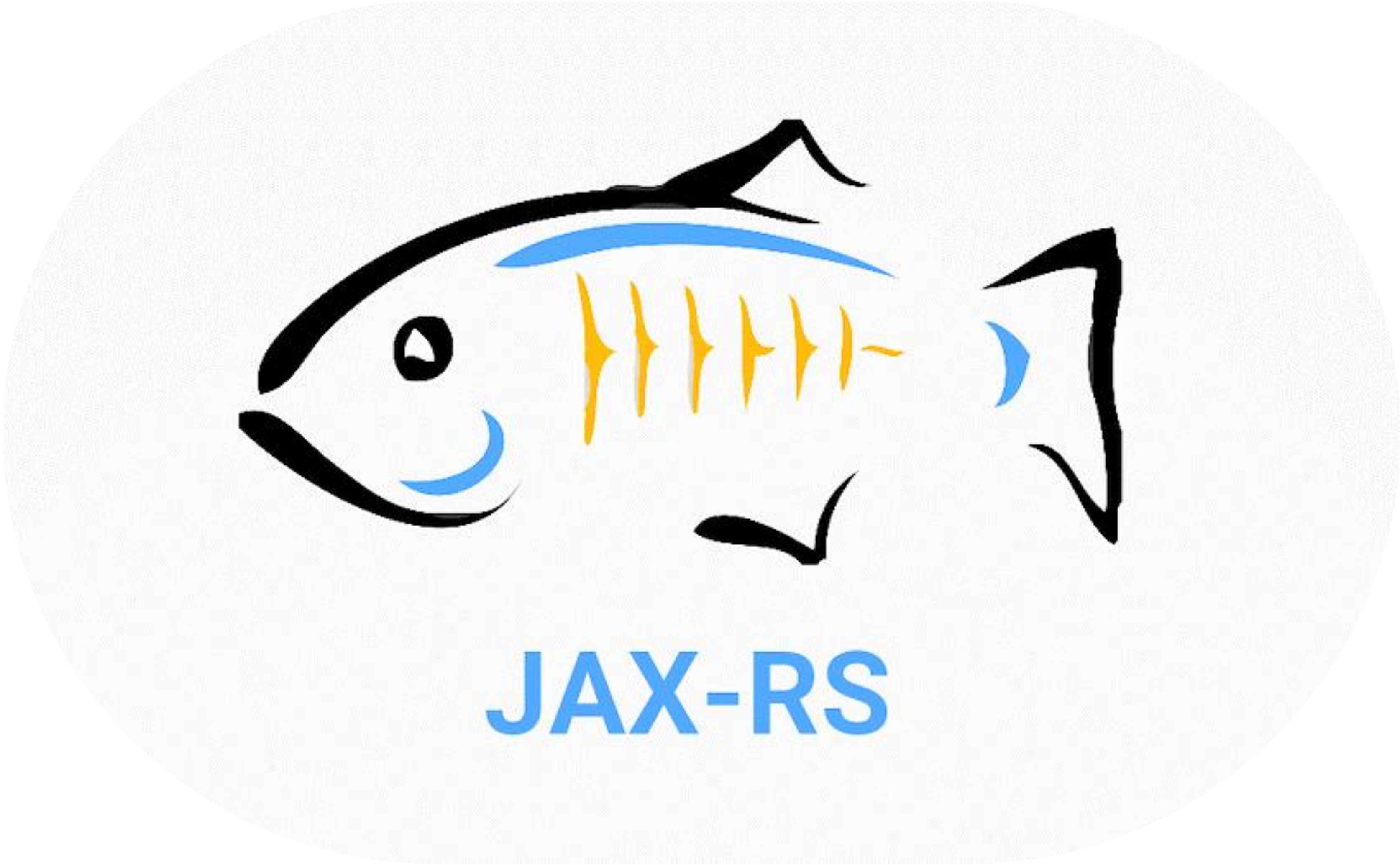
1. Creating model components
2. Creating user service components

TECHNOLOGY

Example of Creating a Web Service

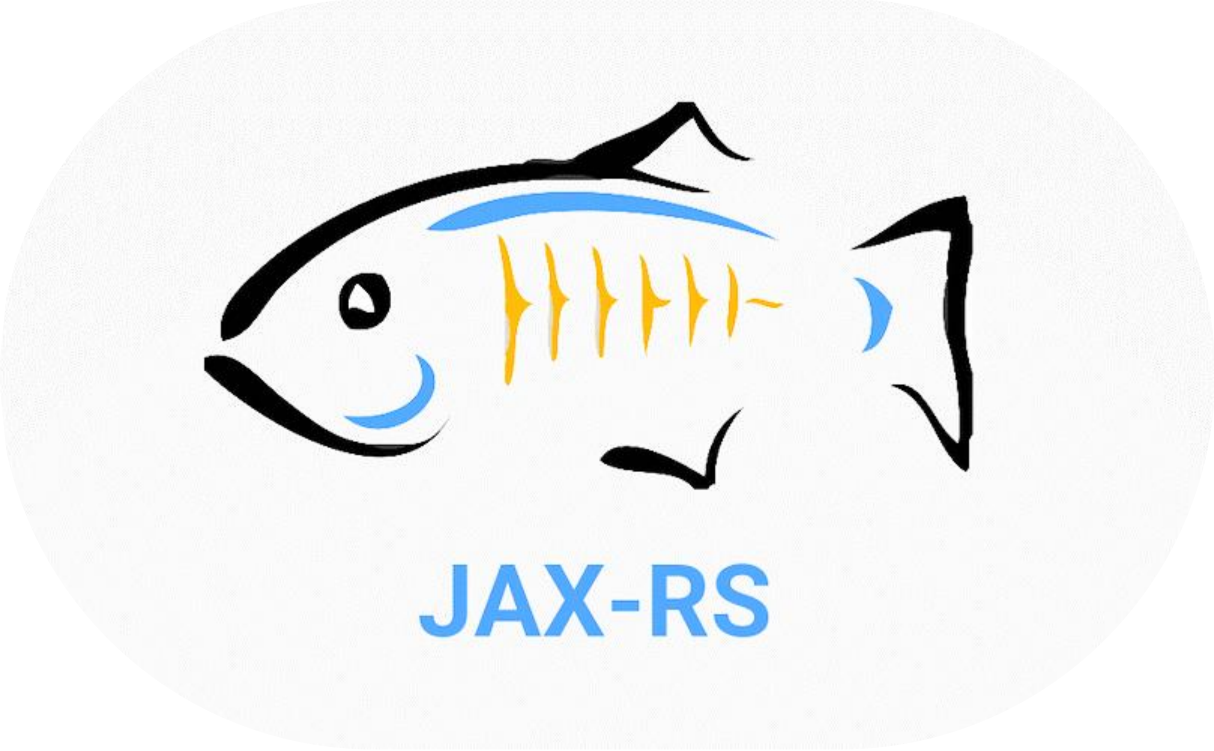
Web Services

To create a Web Service, understand JAX-RS API and its Annotations



RESTful Web Services

JAX-RS API is used to develop RESTful Web Services in Java and is officially approved by Oracle.



JAX-RS API contains several annotations, making it easier to develop a REST architecture.

Annotations

Annotations defined by JAX-RS:

Annotation	Description
@Path	The @Path annotation is a relative URI path that indicates the location of the Java class, for example, /testworld. Here, one can append the variables to the URIs to make a URI path template. For example, /testworld/{userId}. Here, userId is a variable added to the URI Path and provides the user's ID when passing the URI.
@GET	The @GET annotation is a request method designator, and the Java method annotated with this request method designator will process HTTP GET requests.

Annotations

Annotation	Description
@POST	The @POST annotation is a request method designator, and the Java method annotated with this request method designator will process HTTP POST requests.
@PUT	The @PUT annotation is a request method designator, and the Java method annotated with this request method designator will process HTTP PUT requests.
@DELETE	The @DELETE annotation is a request method designator, and the Java method annotated with this request method designator will process HTTP DELETE requests.
@HEAD	The @HEAD annotation is a request method designator, and the Java method annotated with this request method designator will process HTTP HEAD requests.

Annotations

Annotation	Description
@PathParam	The @PathParam annotation is a type of URI parameter that is extracted from the request URI, and the parameter names correspond to the URI path template variable names specified in the @Path class-level annotation.
@QueryParam	The @QueryParam annotation is a type of parameter that you can extract for use in your resource class. Query parameters are extracted from the request URI query parameters.
@Consumes	The @Consumes annotation is used to specify the MIME media types of representations a resource can consume that were sent by the client.
@Produces	The @Produces annotation is used to specify the MIME media types of representations a resource can produce and send back to the client, for example, text/plain.

Annotations

Annotation	Description
@Provider	The @Provider annotation is used for anything that is of interest to the JAX-RS runtime, such as MessageBodyReader and MessageBodyWriter. For HTTP requests, the MessageBodyReader will map the HTTP request body, and the HTTP response will be mapped by the MessageBodyWriter.

RESTful Web Services: Example

An example of developing a RESTful Web Service using Jersey:

In the dynamic web project in Eclipse, create a Web Service called **Welcome**.

```
package com.example;
import javax.ws.rs.GET;
import javax.ws.rs.Produces;
import javax.ws.rs.Path;

// Welcome Java will at  path "/helloworld"
@Path("/welcome")
public class Welcome {

    // HTTP GET requests will be handled by the web method
    @GET
```

RESTful Web Services: Example

An example of developing a RESTful Web Service using Jersey:

```
// The Java method will produce content identified by the
MIME Media and it is text/plain in our case

    @Produces("text/plain")
    public String getWelcomeMessage() {
        // Return some cliched textual content
        return "Welcome to RESTful Web Service";
    }
}
```

RESTful Web Services: Example

Configuration of web.xml deployment descriptor with the JAX-RS API runtime:

```
<servlet>
     <servlet-name>Jersey Servlet</servlet-name>
     <servlet-class>
         com.sun.jersey.spi.container.servlet.ServletContainer
     </servlet-class>
                  <init-param>
                              <param-name>

      com.sun.jersey.config.property.packages
                              </param-name>
                              <param-value>
                              com.example
```

RESTful Web Services: Example

Configuration of web.xml deployment descriptor with the JAX-RS API runtime:

```
                  </param-value>
                  </init-param>
                  <load-on-startup>1</load-on-startup>
      </servlet>
<servlet-mapping>
     <servlet-name>Jersey Web Application</servlet-name>
           <url-pattern>/*</url-pattern>
</servlet-mapping>
```

RESTful Web Services: Example

Access the RESTful Web Service using:

http://localhost:8080/estore/welcome

Output:

```
Welcome to RESTful Web Service
```

Create a RESTful Web Service

Problem Statement:

You have been asked to create and consume a RESTful web service with Jersey.

ASSISTED PRACTICE

Assisted Practice: Guidelines

Steps to be followed are:

1. Creating a dynamic project
2. Setting up pom.xml and creating the service class
3. Creating an HTML file and mapping a Servlet in web.xml
4. Creating a JSP file and mapping it to a Java client file

Key Takeaways

- Web Service is a method to send messages between the server and client applications.
- SOAP is a transport-independent messaging protocol.
- SOA is a style of software design where services are provided to other components by application components.
- HTTP protocols are employed by RESTful Web Services for communication between the server and the client.

TECHNOLOGY

Thank You