**Web Services**

Web services allow different systems to talk to each other over the internet. Generally, they allow for interactions between the client and the server. Data is transferred between client and server with a common language like xml, JSON…

The two types of web services are:

1. SOAP => XML
2. RESTful => http

Advantages

**Reusability** – Used by multiple systems including 3rd parties

**Language transparency** – This means that the service language can be written in one language, like PHP, and the service client can be written in a totally different language, like python. These can communicate because they speak a common language like XML, JSON...

**Usability** – Can be used by a wide range of platforms

**Deployability** – Over the internet so globally available

Considerations

**Latency** – Amount of time from a request to receive a response. Latency increases when the client and server are running on different machines

**Partial failure** – When a server or network fails from being down or overloaded. The more web services used in a single application, the more chance it is to receive a partial failure

Partial failures

The application should be designed to handle partial failures. E.g. imagine buying a book on Amazon:

*Service 1 (main application) –* search book, find book, add book to cart

*Service 2 –* buy book

If *service 2* has a partial failure, Amazon leaves the book in the basket for the user to try again at a later time.

Security

When allowing access to data via web services, security is very important. We want to limit who can view and update data.

**Authentication** – Validates identity of the client, e.g. username & password

**Authorization** – Once authenticated, what is the user authorized to do with the data? E.g. update, delete, get…

*Basic auth* is the simplest protocol available for performing web service authentication over the http protocol. A 404 status is return if unauthorized access.

*API key auth* is another protocol used.

API principle

**URI** – Uniform Resource Identifier

**Operations** – GET, POST, PUT, DELETE

**Formats** – html, xml, plain text, json etc…

**Stateless** – server will not store any state (cookies & sessions not remembered/stored)

HATEOAS

**H**yper media **A**s **T**he **E**ngine **O**f **A**pplication **S**tate

This basically means a hypermedia-driven site provides information to navigate the sites REST interfaces dynamically by include hypermedia links with the responses.

{

“name”: “ali”,

“links”: [

“href”: “http://localhost/customer/1”

]

}