Working with REST APIs in SENG Workshop 3

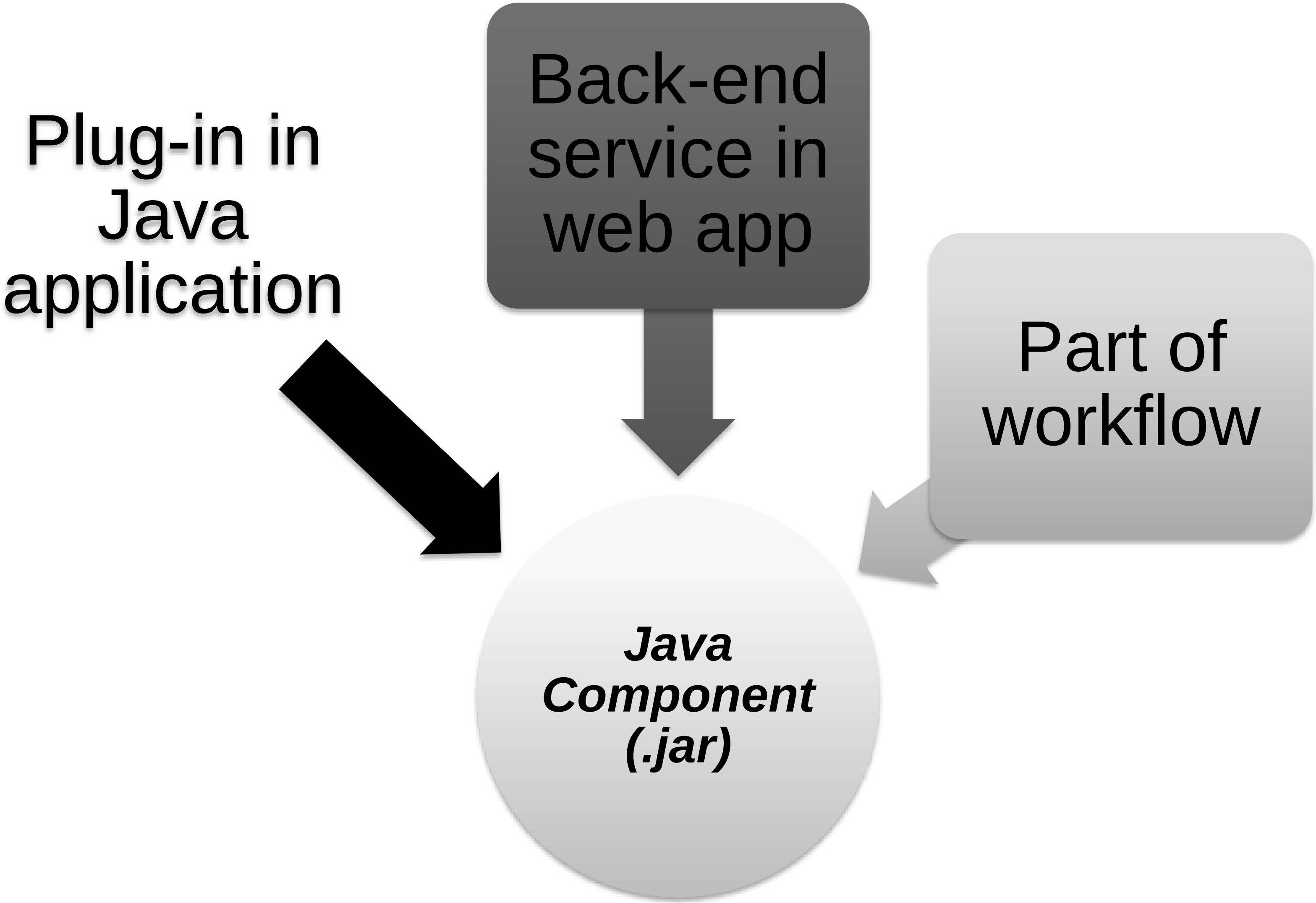
Fethi Rabhi

Software components

Designing Software Using Components

- High quality code
  - Modular
  - High cohesion
  - Low coupling
- Many technologies available for developing components
  - Library components (C# DLL file, JAR File etc.)
  - Web services (e.g. REST)

Multiple reuse of a component



Plug-in in Java application

Back-end service in web app

Part of workflow

Java Component (.jar)

How to Generate **.jar** File

- Export from IDE (e.g. Eclipse)
- Use command line:
    - jar cf jar-file input-file(s)
- Use popular build tools:
    - Maven
    - Ant
    - Buildr
    - … …

C# and .NET

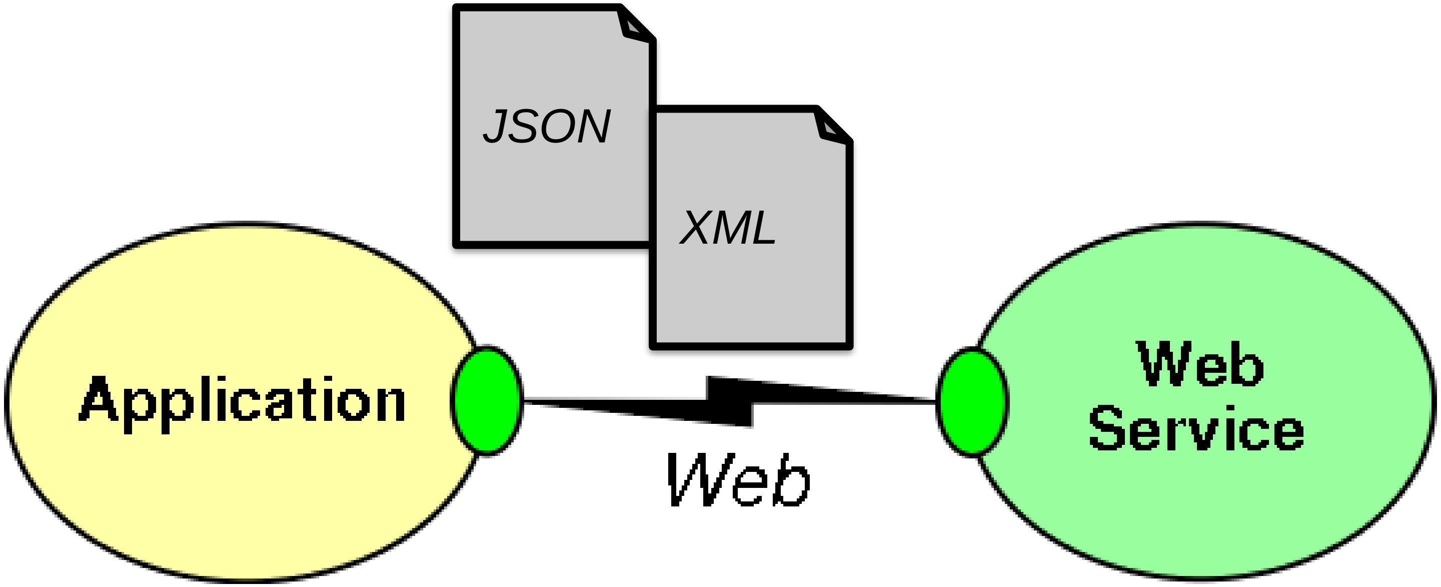
- Component technology for Windows systems
- DLL = basic component that can be executed by a Windows application
- Many utilities for creating and managing components

DLL File Example

- Create C# Classes
- Generate DLL file
- Generate EXE file
- Run the EXE file

REST APIs

Web service
WSDL/REST



- Representational State Transfer (REST) is an “architectural style” defined by Roy Fielding
  - The concepts of REST are independent of the Web, but the Web is well suited for the REST
- REST includes:
  - Resources(things) with
  - Unique ids (URLs) that can come in many
  - Representations(text, html, json, xml)
  - Verbs(GET, PUT, POST, DELETE)

Most common operators

GET

Retrieve a representational of resource (without changing it)

PUT

Create or replace a resource by supplying representational to it

DELETE

Ensure that a given resource is no longer exist

POST

Augment a resource with additional representational

Restlet is a Java framework for implementing REST architecture.

- Operators, Resources, Representations are all class entities in Restlet
- Highly pluggable implementation to support extensibility and interfaces to other web technologies

Atom,GWT, JSON,XML,SSL,Jetty, etc..

Calling several REST APIs

- Can be done programmatically
- Can use Business Process Management Framework
  - BPEL/BPMN
- Can use a workflow language
  - TAVERNA

Using SWAGGER

- To enable testing, all APIs and their documentation will be made available via SWAGGER
- More information on using SWAGGER
  – https://swagger.io/tools/open-source/getting-started/
  – https://idratherbewriting.com/learnapidoc/pubapis_swagger.html
  – https://www.baeldung.com/swagger-2-documentation-for-spring-rest-api

More on SWAGGER

- Swagger Editor
  - This is the "official" text editor that can be used immediately to create documentation by hand.
  - Demo version at https://editor.swagger.io
  - Has example already populated (data saved locally in the web browser, not the cloud).
  - Can be downloaded and installed locally from https://swagger.io/tools/swagger-editor/download/
- OpenAPI Specification
  - The official specification reference currently version 3.0.2: https://swagger.io/specification/
  - Students encouraged learn the latest 3.0 version because better than v2.0 but many tools still based on 2.0

HANDLING INPUT FILES AND OUTPUT FILES REST-FULLY

Solution #01
Enable File upload via REST commands

Input:

- Upload entire files to the web service.
  - E.G. upload pictures to Facebook, or files to Dropbox
  - Granted there are UIs to facilitate this, and for this first deliverable there is no user interface.
- Achieved through standard HTTP request verbs
  - E.G. **POST** , **PUT**
  - Make clear API(s) using HTTP for file uploads.

Solution #01
Enable File upload via REST commands

Output:

- Teams have more flexibility in module output.
- **OPTION #01**
  - Return output as JSON response.
  - This is a very common return format for API calls in the real world.
- **OPTION #02**
  - Return download links to output files.
  - Links would be returned as part of a JSON response (as opposed to all the information being contained in a JSON response as with the first option).
- **SUGGESTION** Examine the responses from API calls from available services like Twitter

Solution #02
Multipart / form-data

- Sending multipart / form-data message
- Very complicated!

Solution for SENG Workshops

- Both solutions are applicable
  - Creative, alternative and effective software designs are always impressive (to us).
- There will be points allocated to the adoption rate of your modules.
  - Practical indicator of design quality => How many people use it!
- Document Well
  - Your solution can't be used if no one knows how to use it *properly*!
- **ASK QUESTIONS!!!**
  - Filling in gaps in your knowledge and information provided : That's part of the **real process** out there
  - Asking effective questions early is paramount

Common Mistakes

- Component run accurately , but Log file incomplete or doesn’t exist
- No clear instructions on how to execute the component.
- The group said the version on their website is the wrong version, they will upload the correct version as soon as possible.
- Clear execution instructions, but lack of unit testing, errors generated when running the component
- Output doesn’t change when changing input parameters (i.e. hardcoded the parameters)
- Who is doing what in the group, clarify from the beginning don’t leave it to late.