**LALITHA MADHURI PUTCHALA**

**MY CENTRAL ID: 700657631**

**SOFTWARE ENGINEERING – ASSIGNMENT 2**

**FALL 2016**

**CRN # 13944 CS 4910**

**Part-1:**

**CASE Tools:**

Computer-Aided Software Engineering tools basically provide automated assistance for the software development. CASE tools helps software developers to step into complexities of the code while looking through the design of large projects. These computerized tools can be used throughout the Software Development Life Cycle(SDLC) phases. By these usage, we can ensure the reduced cost and time of the software development along with minimal defect rate and the high quality.

**Benefits of Case Tools:**

1. Assure Higher System Quality
2. Better Use of DP resources
3. Increase System Productivity
4. Increase System Reliability
5. Assure Greater User Satisfaction
6. Integrate Software Development and Data Administration
7. Increase System Efficiency
8. Lengthen System/Program Life
9. Improve System Portability
10. Better Project Management
11. Enable End User to Design System
12. Reduce Backlog
13. Better Documentation
14. Programming in the hands of non-programmers
15. Intangible Benefits

The benefits of CASE tools with respect to different phases of SDLC life cycle are:

Analysis and Design Phases:

1. Improve Analysis and Design Productivity
2. Improve Design Accuracy
3. Increase Communication between Developer and User
4. Increase User Involvement

Implementation Phase:

1. Increase Reusability of programs
2. Increase Programmer Productivity
3. Assure Faster System Development
4. Decrease System Testing Time

Maintenance Phase:

1. Improve System Maintainability
2. Improve System Documentation
3. Improve Maintenance Productivity

**CASE Tools:**

The different categories of the CASE tools are below:

* Diagram Tools:

Used to represent graphically the data and control flow, system components and interactions among software system components and structure

* Process Modelling Tools:

Create a model for the software process that can be used for software development. Also, these tools help the managers to select a best fit model and modify it as per the requirement for the software. EPF composer is an example for this type of tool.

* Project Management Tools:

These are used for project cost and effort estimation, project planning, scheduling and resource planning. The team has to strictly execute the steps mentioned in the software project management. These tools also aid in storing and sharing of the project information throughout the organization during real-time. Well noted examples are Trac project, Creative Pro Office and Basecamp.

* Documentation Tools:

Usually, documentation will be carried right from the starting point of the project through all the phases of SDLC process till the completion of the project. These tools take care of document generation for the end users and technical users. The usual technical documents refer to user training manual, system manual, reference manual, installation manual. The end user documents demonstrate the how-to and functioning of the system like user manual. The noted examples are DrExplain, Doxygen, Adobe RoboHelp.

* Analysis Tools:

Gathering requirements, checking inconsistency automatically, diagrams inaccuracy, erroneous omissions or data redundancies are taken care by these tools. Accept 360, CaseComplete for requirement analysis, Accompa, Visible Analyst for total analysis are the examples for the Analysis tools.

* Design Tools:

These tools help in designing the block structure of the software for the software designers which can be further bi-fabricated using refinement techniques into smaller modules. Ex: Animated Software Design

* Configuration Management Tools:

These kind of tools deal with a) change control management b) version and revision management c) baseline configuration management.

Examples are Git, Fossil and Accu REV

* Change Control Tools:

These are also said to be a part of configuration management tools. Changes made to the software after the baseline fixes and software releases are dealt by these change control tools. They automate file management, change tracking, code management etc., Change policy of the organization is also enforced by these tools.

* Programming Tools:

Provide comprehensive aid in software product built and also consist of features of testing and simulation. These consists of IDEs, simulation tools, in-built modules and other programming environments.

* Prototyping Tools:

A prototype of the software is a simulated model of a software product. It provides a first look of the product and simulates the selected aspects of the software product. These tools along with graphical libraries create design and interfaces that are hardware independent. They provide rapid prototypes on existing information and provide simulation of software prototype. Mockup Builder, Serena prototype composer are noted examples.

* Web Development Tools:

Provide live preview of the development and gives an idea about its future look after the development. They help in designing web pages with all allied elements like script, text, forms, graphic and so on. Adobe Edge Inspect, Fontello, Foundation 3, Brackets.

* Quality Assurance Tools:

Testing tools in a software company monitor the methodologies and engineering processes to develop the quality of the software product as per the organization standards. Quality Assurance Tools consist of change control tools, configuration and software testing tools. Examples are Soap UI, JMeter, AppsWatch.

* Maintenance Tools:

These tools include maintaining the modifications of the software product for every release. Error reporting and automatic logging techniques, root cause analysis and automatic error ticket generation are some of the CASE tools that help the software organization in maintenance. HP Quality Center, BugZilla are few examples.

**Part -2:**

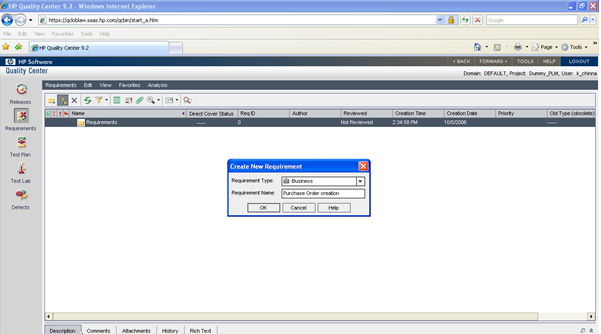
Two case tools that I have selected are 1) HP Quality Center – A Project Maintenance Tool and 2) Soap UI – A Web Service Testing Tool

**1) HP Quality Center – A Project Maintenance tool**

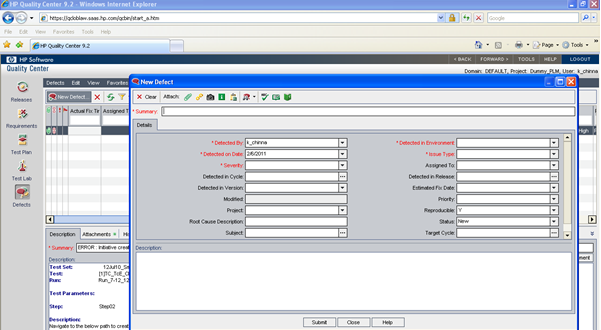
HP Quality Center (QC), a test management tool, is now popularly known as Application Life Cycle Management (ALM) tool, as it is no longer just a test management tool but it supports various phases of the software development life cycle.

HP-ALM helps us to manage project milestones, deliverables, and resources. It also aids in keeping track of project health, standards that allow Product owners to gauge the current status of the product

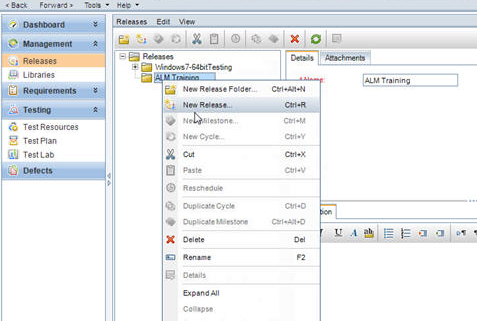
***Creating a New Requirement in HP QC:***



***Defect Module in QC:***



***Creating a Release:***



***Advantages of Quality Center:***

1. HPQC has feature for RTM (Requirement Traceability Matrix) i.e., it allows the requirement mapped to a relevant test case(s) and defects. This will help the project team to take informed decisions on the test coverage before Go-live. For e.g.: Double click on a defect (which is already linked to a requirement and test case), navigating to “Linked Entities” shows the relevant requirement and test case connected to the defect as below.

2. By default, the requirements, test plan and test lab module are displayed in the tree (hierarchy) structure. The defects module will be shown “Grid View”. However, User can always go to “View” Menu -> “Grid View” to change the view from tree view to Grid view and can customize the display columns.

3. E-mail notifications: The QC Admin can customize the email notifications each a time change in status happens to a Defect in Defect module.

4. All the roles in the QC will fall under mainly under following groups, i.e., TDAdmin, Project Manager, QA Tester, Developer, and Viewer.

5. Analysis Menu allows the user to generate different types of reports or graphs (Bar chart or pie chart etc).

6. Click on Help > Add-ins page menu, will display the useful plug ins that are available to connect to Quality Centre i.e., integration with other HP Testing tools such as Quick Test Professional and Word and Excel Add-ins to import tests into HPQC.

7. Click on Tools > Document Generator allows the user to create all or selected tests (requirements) into a complete Word document which will be handy in case of unavailability of QC during maintenance window timelines.

8. Favorites allow the user to create public or private “Customized settings”. The public favorite allows the setting to be visible for the entire project team who logs into particular project in HPQC, whereas private favorite allows the setting to be visible only for local user. For e.g.: public favorites will be useful for daily Defect Triage meetings during test execution. Private favorites can be used by testers for him to see how many test cases he has executed or high priority defects raised etc.

***Reference:***

<http://www.learnqualitycenter.com/8-features-quality-center/>

***Disadvantages of Quality Center:***

1. Customization is too rigid.

2. Workflow script code allows you to customize a lot of what happens in the system, but not enough events are exposed to provide full control.

3. User-defined fields can be created, but you are limited to a pre-defined set of field types: user list, lookup list, string, date, number. This covers most needs, but does have some limit. For instance, there is not a CheckBox style option (which actually used to exist in older versions of TestDirector).

4. Not all modules provide customization or workflow handling.

5. The Quality Center client tends to crash too frequently with Access Violation errors. Maybe this is the browser they host it in, maybe it's the ActiveX technology they use, or maybe it's just bad code.

6. Back-end overhead. The system may be too big and to demanding for most organizations. You will need to install, manage and maintain a Database Server, an Application Server, and sometimes even a Web Server. Not to mention back-ups and other procedures that need to be done regularly. This is something people seldom take into account and for some organizations this is simply too much for what they gain.

7. Customizations are not only rigid, but they demand knowledge of VBS and in many instances of the Quality Center API. Many times this knowledge is not there, and the organizations are left with an expensive tool that can potentially help them but it's not right for them...

8. Slow over long distances. This is especially true for companies that are distributed around the World.Even if the system is Web-Based and you can connect using your browser, as you start working from remote locations (e.g. US vs Europe or Asia) the response time starts getting very annoying, unless you have some pretty advanced back-end communication channel.

9. Last but not least, and hopefully it will only be temporarily... Support from HP has been pretty slow and not too helpful in the last months/years. They used to be pretty good but lately most of our tickets wait for weeks/months or are closed without a popper answer. Again, hopefully someone from HP is reading this and will do something about it.

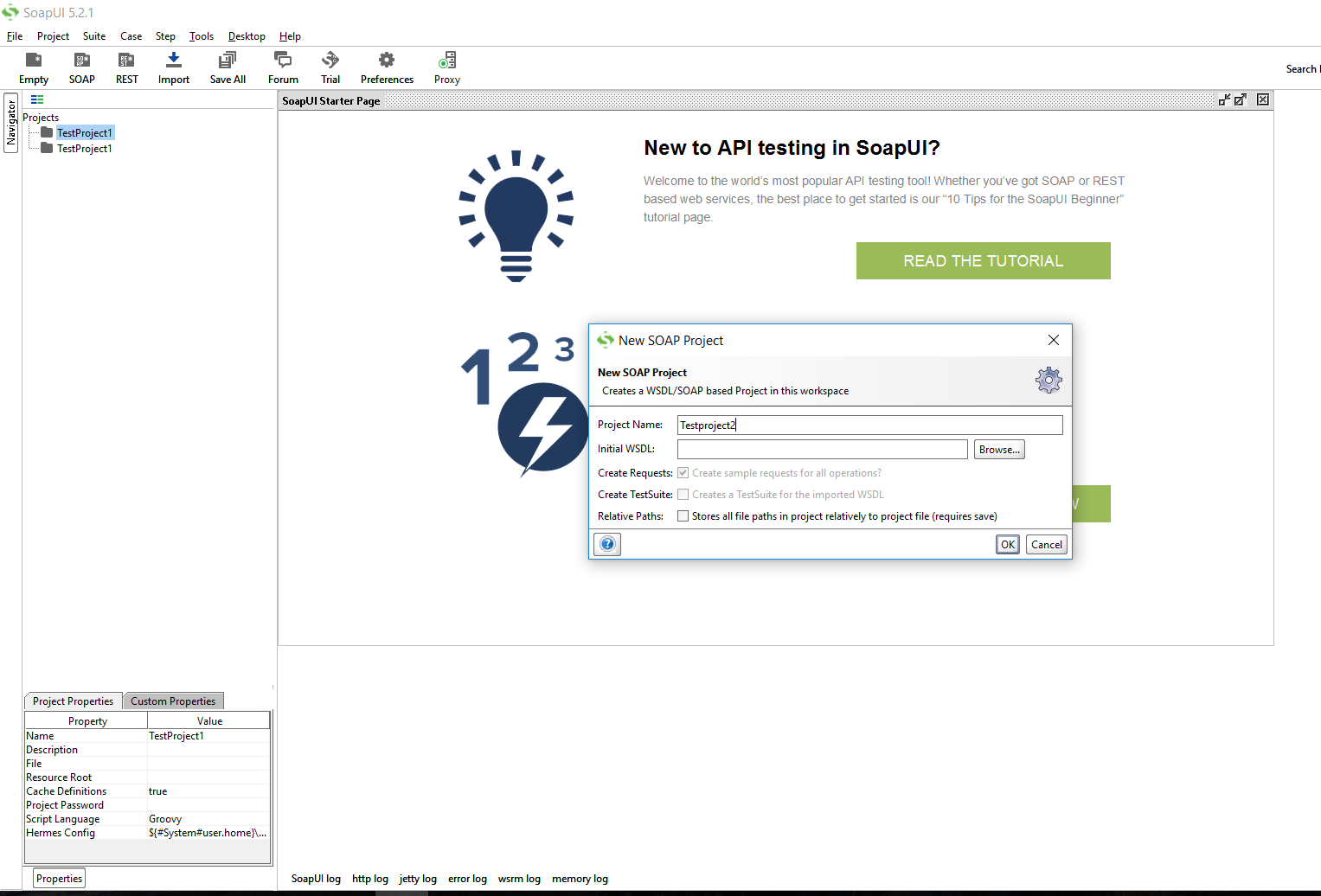
***Reference:***

<http://hpsqc.blogspot.com/2010/05/what-are-disadvantages-of-quality.html>

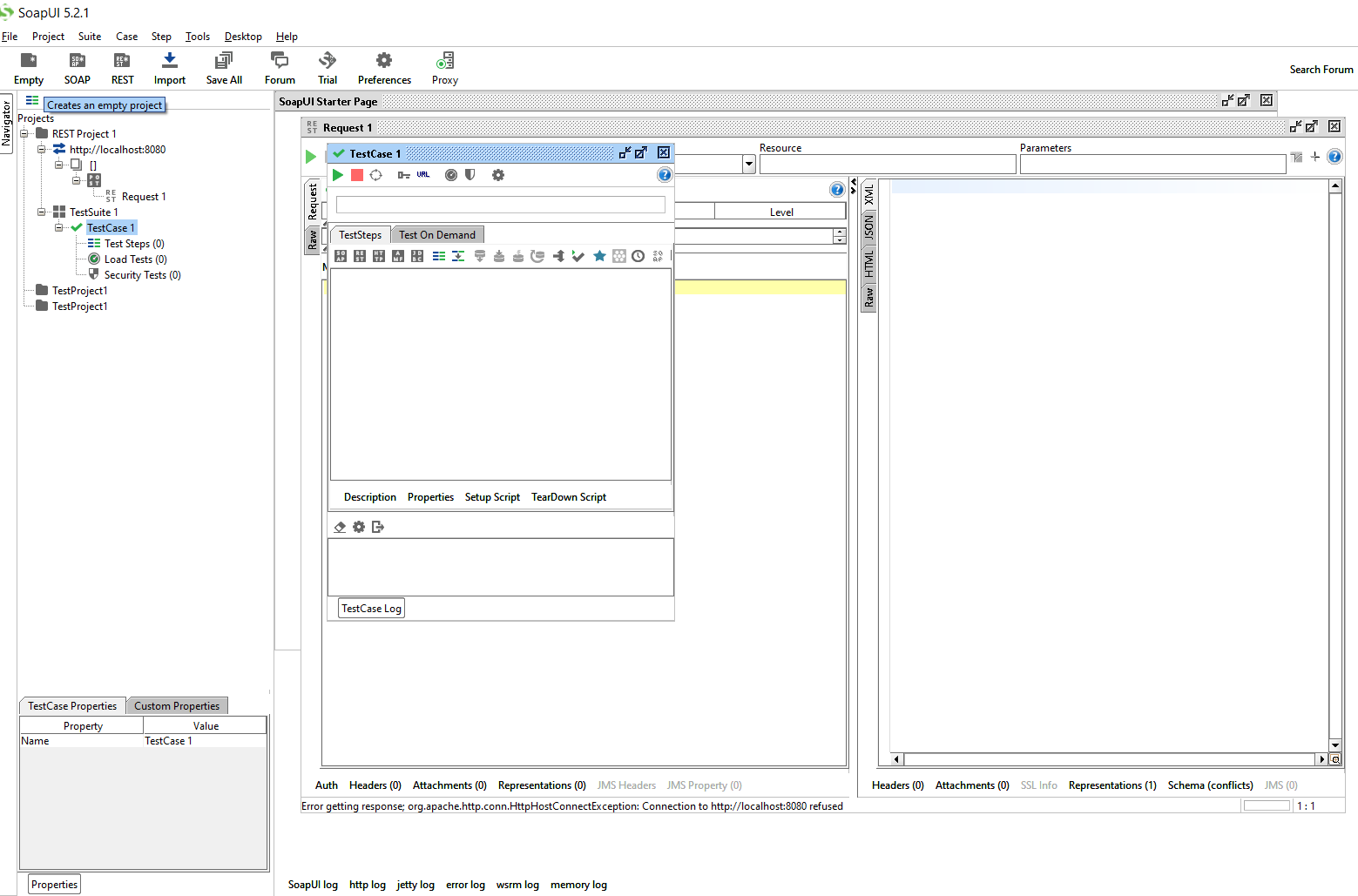
**2) Soap UI: - An Open Source Web Testing Application for Service-Oriented Architecture**

Soap-UI is the world's most widely-used open source API testing tool for SOAP and REST APIs. Soap-UI offers SOA Web Services functional testing, REST API functional testing, WSDL coverage, message assertion testing and test refactoring. With over 10 years of experience backed by a vast open source community, Soap-UI is the de facto method for ensuring quality when developing APIs and Web Services.

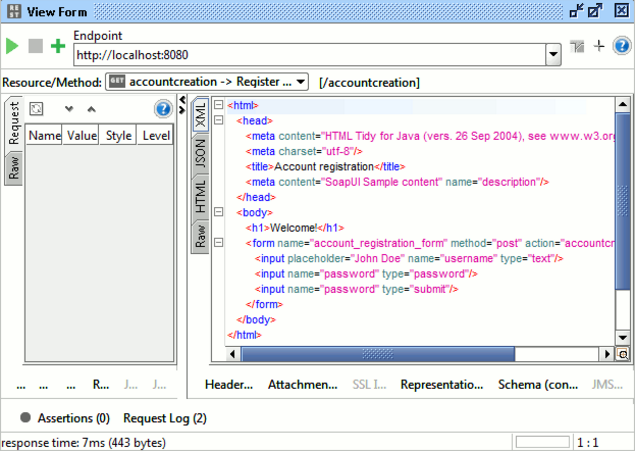
***Creating a new soap project in Soap UI:***



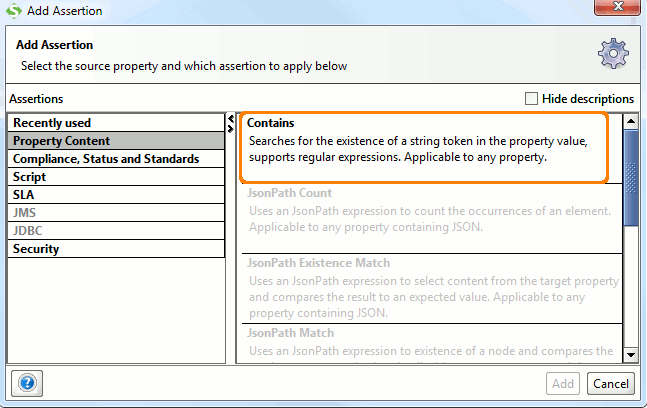
***Creating a new test suite and test case:***



***Screenshot of Response after triggering the request:***



***Screenshot while trying to add an assertion for a test case in Soap UI:***



***Soap UI Advantages:***

SOAP – Simple Object Access Protocol – is probably the better known of the two models.

SOAP relies heavily on XML, and together with schemas, defines a very strongly typed messaging framework. Every operation the service provides is explicitly defined, along with the XML structure of the request and response for that operation. Each input parameter is similarly defined and bound to a type: for example, an integer, a string, or some other complex object.

All of this is codified in the WSDL – Web Service Description (or Definition, in later versions) Language. The WSDL is often explained as a contract between the provider and the consumer of the service. In programming terms the WSDL can be thought of as a method signature for the web service.

1. Tool for functional testing of Web Services

2. Easy to use Graphical Interface

3. Easy Test Case creation and execution

4. Provides complete test coverage

5. Supports all standard protocols

6. Load testing via Load-UI

7. Automate Websites

8. Licensed

***Soap UI Disadvantages:***

* While the WSDL may seem like a great thing at first – it is self-documenting and contains almost the complete picture of everything that is required to integrate with a service – it can also become a burden. Remember, the WSDL is a contract between you (the provider of the service) and every single one of your customers (consumers of the service).
* WSDL changes also means client changes. If you want to make a change to your API, even something as small as adding an optional parameter, the WSDL must change. And WSDL changes also means client changes - all your consumers must recompile their client application against this new WSDL. This small change greatly increases the burden on the development teams (on both sides of the communication) as well as the test teams. For this reason, the WSDL is viewed as a version lock-in, and most providers are very resistant to updating their API.
* Furthermore, while SOAP offers some interesting flexibility, such as the ability to be transmitted over any transport protocol, nobody has really taken advantage of most of these. Thanks to how the Internet evolved, everything that matters runs over HTTP. There are new advances, but most of these are being hampered by infrastructure routers refusing to route non-standard HTTP traffic. Just consider: how long has the world been trying to switch over to IPv6?
* There is definitely a **need for a more lightweight and flexible model [than SOAP]**. Any situation where the size of the transmitted message does not matter, or where you control everything end-to-end, SOAP is almost always the better answer. This applies primarily to direct server to server communication, generally used for internal communication only within the confines of one company. However, there is a need for a world where almost every person on the planet has several low-memory, low-processing-power devices connected to multiple services at all times, there is definitely a need for a more lightweight and flexible model.

***Reference:***

<https://www.soapui.org/>