Model-Driven Approach
Frequently Asked Questions

Mario G.C.A. Cimino

Department of Information Engineering

06) 08/12/2018 How to structure the Testing application

The purpose is to write a minimal testing code. The Orchestrator class should be a simple Java class with a unique method called *executeProtocol*. Two string variables, *url* and *message*, for each web service are statically assigned, and then the orchestration logic is executed. The result of the method is a StringBuilder variable *log* that is returned to the Testing application. During the execution of the protocol, a different char is added to the *log* for each type of event that is important for testing. For example, in a tab-separated values, *log.append("u\t")* in case of unavailable web service. The Testing application is a simple Java class with a unique method *main*. In the first part, a for-loop invokes the protocol for a predefined number of times and fills a corresponding array of logs (*String[] logs*). In the second part, the array of logs is stored in a text file *logs.xsl*[1]. The logs.xsl file is a deliverable, which can be opened and processed with MS Excel, by adding some aggregated information[2].

05) 06/12/2018 How to emulate complex parameters with mnemonic strings

The purpose is to write a minimal code. The input/output complex parameters of web services are simplified by using simple types. The implementation of each service is emulated via coin flip (see the emulation of DistilledWater). In each

---

[1] The Java statement for this task is Files.write(Paths.get("logs.xls"), (String.join("\r\n",logs)).getBytes());).
It requires jdk8, www.iet.unipi.it/m.cimino/sse/res/jdk8.zip

[2] For example, COUNTIF(A2:F1000;"u") to count the events “unavailable web service”.

service the input is not processed: simply, the output variants needed by the orchestrator are statistically generated.
For this reason, a service does not provide a detailed content (as the XML instance does) but a simple mnemonic description identifying the output variant.
For example, a web service returning two variants of the user profile “Jack, male, 35” and “Jane, female, 16”, will return only “male” and “female”, respectively, if the orchestration condition checks whether the user is male or female. This corresponds to a Java condition *response.contains("male")*.
If the orchestration condition checks whether the user age is under 18, the web service can return two string variants, e.g. “35” and “16”. The orchestrator, will first read the *age = Integer.parseInt(response.substring(x,y))*, where x and y are positions statically determined according to the SOAP message.

04) 26/05/2018 How to request corrections to the JoA

Correction requests should not be specified with narrative style in natural language.
To avoid ambiguity, and for best efficiency, please request corrections with the following format:

copy & paste of the current record
REMOVE

or

copy & paste of the current record
copy & paste of the current record modified as you like.

Corrections should be requested as soon as possible, and not after many days.

03) 20/04/2018 Structure of early requirements

Consider the following structure for the textual document, avoid images and extra narrative text:

NAME OF APPLICATION #1
short name of operations with input and output parameters (e.g. "calendar", "map")
NAME OF APPLICATION #2
short name of operations with input and output parameters
NAME OF APPLICATION #3
short name of operations with input and output parameters
NAME OF APPLICATION #4
short name of operations with input and output parameters

DESCRIPTION OF A PROTOCOL
Narrative text of a normal case with some variant (at least three "IF") and some "FOR EACH" (at least two ones), using all operations of the four above applications.

Note: IF and FOR EACH should be performed by the human coordinator, involving more applications (coordination logic) and not performed by the single application logic, which is out of the scope.

02) 02/06/2016 Which service availability rates should I use?

You can take into account the following examples of service availability rates offered by large companies:

https://cloudharmony.com/status-1year-of-storage-and-compute-group-by-regions-and-provider

01) 26/04/2016 Is this diagram right?

Actually what is also significant is the sequence of steps that brought you to the diagram. For example, when creating a use case diagram, first summarize the initial textual analysis and the rules in the book to find actors and use cases, in order to show with evidence that your solution comes from knowledge. Avoid agnostic deliverables, i.e. diagrams without written knowledge.