2.1

Read the document with the notes on Other Testing Methods (other\_tools\_methods\_notes).

FSM: used in numerous disciplines and elements consists of states, transitions, inputs and outputs. The system is always in ‘current’ state.

Random access that undergoes transitions from one state to another and must possess a property that is characterized as memoryless, where memoryless assures the probability distribution of the next state depends only on the current state and not on the sequence of events that preceded it.

In the context of the tool you are using in the laboratory, choose a functionality for which you think you need FSM testing and another one for which Markov chains could be used. Post your two proposals in the course forum. Explain in two sentences why the two techniques are required.

1. Importing a furniture model is suitable for Markov chain testing since it involves sequential loading of pre-defined model file loaders. Markov chain can test with importing list of furniture model files for probability of getting loaders error while traversing loader list, or chance of getting a not supported file. It focus on testing the compatibility for mainstream model file format.

2.2

Review the content of the Other Testing Methods (other\_tools\_methods\_notes) notes with special attention to:

How to enhance testing strategies.

Understand the concept of single state model.

Only one state waiting to execute a transition at a time, it can consider inputs as trigger events entering into the state, outputs are the results of that state and inputs, while would be transit to next state. This model holds this type of iteration to simulate sequential tested behavior.

Justify the use of FSMs.

Mapping of state, transition, input and outputs. Transition can be function of a states and a given input, that also imply to a next state. At same time, output can be function of state and an input or is just determined by that state. FSM system only considers one state at a time.

Description and use of FSMs.

FSM Representation.

FSM creation.

From FSM to Markov chains.

Test management tools.

2.3

PerfectSoftware has a product that is a web application hosted in local servers. There are thousands of users using the platform. The server running this product writes a line to a file every time a page is accessed by a user. The line has the following format (represented with one field per line to simplify visualization):

97.185.220.155

user

-

[09/Oct/2012:17:24:11 +0900]

"GET /platform/folders/user/document.html HTTP/1.1"

401

6012

"https://perfectsoftware.com/platform/folders/user"

"Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.1 (KHTML, like Gecko) Chrome/14.0.835.202 Safari/535.1"

The two relevant fields are number 5 and 8. Field 5 contains the page accessed by the user. Field 8 contains the web page containing the link that was selected to access the page in field 5.

You have a system that is too costly to test in all possible combinations. You would like to spread your testing according to the frequency of use of each functionality. With the rest of your team mates sketch the procedure to process this information (the file has a large number of lines like the one in the example) and produce probabilities to transform a (supposedly already existing) FSM into a Markov chain.

Transvers file attempt to file a mapping of request and its possible response urls as well as total number of this request, saving into a temporal storage. Then, transvers list of those requests, count the total number for each mapping of request and a response url, divided by total number of this request, it should give us the probability of getting a url with a specific request.

2.4

1. Review the section on Testing Management Tools on the notes on Other Testing Methods.

2. You are the Quality Manager at PerfectSoftware Pty Ltd. The company has no tool supporting the quality assurance process. Search for a commercial tool that offers support for Testing Management and has documentation about its functionality available to review. Consider reviewing a product that is not one of the big players.

Post a two sentence summary of your impressions in the course forum. Include the price if you found it.

*http://www.andagon.com/en/loesungen/aqua-it-project-management-software.html*

1. The management tool I choose is Aqua. It support automated unit tests, automated GUI tests, manual tests, defect management and having Multiple testing environments .For example, bi-directional synchronization with Atlassian Jira and we can also use aqua without leaving your development environment with the aqua plug-in for Eclipse

2. To use the aqua to the fullest, we could organize a team and make a test management for our product. First, creating and managing test cases, such as user login, searching a furniture in the furniture catalog pane, adding a furniture model. Every task could be assigned to a particular team member by opening a pull down menu. With aqua Project Planner, team member could monitor the progress made in processing requirements, tests, or errors for the quickest reactions and permanent project transparency on costs, budget, and timetable. When it comes to a defect, integrated defect management forms the interface to have precise descriptions of defects, including screenshots of defects, and adding more valuable information as attachments.

3. Review the functionality of the selected product. Write a maximum one page review of the product. You may take into account the following points for your review (in no particular order):

Supports description of software requirements. Connection between software requirements and tests. Grouping of tests in suites or similar units. Support for manual test creation. Support for test creation. Multiple testing environments. Reporting capabilities. Support for collaboration among team members. Connection with project management (milestones, deadlines, deliverables, easy creation of new tasks, etc.) Defect tracking functionality or connection with auxiliary platform for this purpose. Price? (Not always available)

Aqua consists of 4 modules for quality assurance processes:

Requirements management: captures requirements in text form and structures them in hierarchies. Integrate Office files. Use the integrated UML Designer and get a better picture by means of the aqua dependency maps.

Project management: model aqua according to the workflow for your project. Use the project-planner and Gantt charts. Prepare reports and keep track of things with dashboards and cutting-edge charts that show you the status quo of your projects at any time.

*Test management*: organize your quality assurance to its full extent and integrate requesters and development better for perfect results. Save time by automatically generating test-cases and carrying out manual and automated tests with ease. Reporting and Dashboards make team’s performance visible. Automatically generating test cases from requirements and corresponding dependency diagrams. Grouping test cases in test scenarios.

*Defect management*: track your defects and automatically generate defects from test-cases. Integrate other information, e.g. screenshots and attachments.

2.5 Present the features of a test management tool

Recommendation Testing Management tool for a company:

Xstudio can create a project by starting a new product in the system, and it could add the requirements of product into requirements tree and link them to the product.

It could also link the test cases with requirements and test converge of requirements would be indicated with percentage.

There are campaign session to gather the similar tests wish to run and each campaign has progress bar to indicate the status of those tests.

Creation of test cases would ask to define execution method, which is either manual or automatic.

This system provides various type of reports, such as Requirements report, Tests Campaigns Report and Defects Report.

The dashboard function of this system provides the information about

The tests you must develop (if you’re a test developer),

The campaigns sessions you must execute (if you’re a tester),

The bugs you have to fix (if you’re a developer) or the bug-fixes you need to verify (if you’re a tester).

However, it does not provide many functions connect with project management, despite of a project overview report.

The defect tracking function has all necessary states of defects or bugs need to know during the testing, it includes the total number of defects, the number of defects in each category, the number of defects in each folder, the status of each defect, the severity of each defect, the priority of each defect.

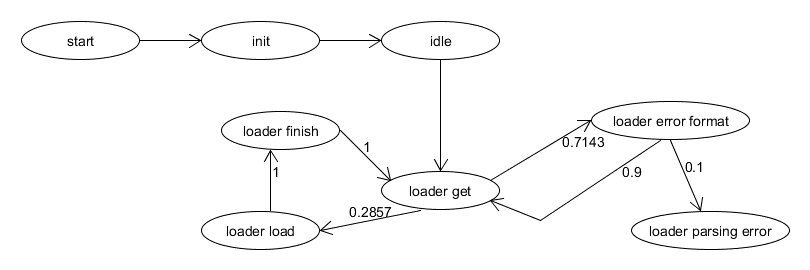
2.6 Create a FSM for testing your product

An initial description of a sequence of steps in your product that is suitable to be used with FSM testing.

FSM and its derived Markov chain of importing a furniture model function

State identification: this functionality start from accept the model file path then try with the 4 loader in sequential order (loader get), then if there is a format check error occurring before going to loading process (loader load), it goes to loader error format and then back to loader get for trying with another type of loader, otherwise it go straight to loader finish and then back to loader get if the other file is waiting for loading. In the case that it exist not any appropriate loader, it goes to loader parsing error.

FSM first version: the initial finite state machine as follow:

**

Refinement: there is no duplicate state existing during state identification, therefore no need to refine the FSM.

Markov chain: From the log of loading 4 types of model file, the results of each state transition are:

Total transition start from loader get is 14.0, the number of transition between loader get and loader error format is 10, and number of loader get to loader load is 4.

Total transition start from loader error format is 10.0, the number of transition between loader error formats to loader get is 9, and number of loader error format to loader parsing error is 1.

Total transition start from loader load is 4.0, then number of transition between loader load and loader finish is 4.

Total transition start from loader finish is 4.0, then number of transition between loader finish and loader get is 4.

For probability of each state transition is divided individual number of transition to the total number of transition starting from the state.