Glamour packages - User study

Presentation
The goal of this study is to assess packages forming a application. You will assess packages through their classes and class references, analyzing what they use and how they are used, both internally and externally to the application.

This study is organized in three sections which require more and more in-depth look at packages and their classes. Each section implies that you play a different role when assessing the application, first as newcomer and potential client who wants to use the application, second as an architect who needs to assess the organization, third as a developer who performs maintenance. There are 11 questions in this study, each question relating to a task.

You will perform the study on Glamour packages. Glamour is an engine for scripting browsers for any kind of models. If you are unfamiliar with Glamour, do not hesitate to test it before the study to get a basic understanding of Glamour capabilities. Information on usage and samples are available on: http://www.moosetechnology.org/tools/glamour

Tool used: Package blueprint browser

Instructions
- Use only the tool indicated for the study. Do not use another browser/tool.
- Browse the documentation before performing the study:
    - **PackageBlueprintsPrinciples.pdf** explains the basics of package blueprints
    - **packageblueprints.mov** shows interactions with the package blueprint browser
    - a draft version of the journal paper describing package blueprints is provided
- Process questions in the given order (do not read questions in advance!)
- Please provide only accurate answers like the name of a class or a package, the association between two classes, the method with references.
- **Time yourself** for each question.
- Do not spend more than **20 minutes** on a question. If you reach this limit, write it down, stop the task, and proceed to the next question.
- You also have a time limit of **1h30** to answer the 11 questions so take care of your time.

A. Application assessment
As a potential client, you are assessing the package dependencies of the application. You want an idea about the size of the application and the kind of dependencies needed, especially if it involves new dependencies to be loaded with the application.

1) How big is the application?
Time taken: 4 min
(a) In number of packages
10 visible, 11 from the window title (-announcements does not appear in the outgoing tab, and -scripting in the incoming tab; inheritance tab has all 11)
(b) In number of classes (one of the following ranges):
[ ] <100; [X] 100-200; [ ] 200-300; [ ] > 300

2) What are the most important packages?
Time taken: 5 min

(a) In terms of outgoing dependencies

-Tests and -Morphic, -Examples has a lot of external deps too

(b) In terms of incoming dependencies

-Core, -Browsers

(c) Overall, considering both outgoing and incoming dependencies

-Core has lots of both, -Morphic, -Examples, and -Browsers because they are into many external dependancies

3) Focus on package Glamour-Morphic:
Time taken: 15 min

list all package dependencies which are external to Glamour.

Morphic, Polymorph-Widgets, Mondrian, Morphic-MorphTreeWidget, Balloon, Shout, Magritte-Morph, Graphics, Collections-Strings, Collections-Sequenceable, DeprecatedPreferences, Collections-Arrayed, FreeType, Polymorph-Tools-Diff, Collections-Unordered

(d) in this list, please signal any external package which is not part of Pharo base (i.e., package must be loaded with Glamour).

Mondrian, Magritte-Morph, Shout for sure, probably Polymorph-Tools-Diff and maybe Morphic-MorphTreeWidget as well

(e) are there other unexpected/unwanted package dependencies?

DeprecatedPreferences, but looking at the GLMMorphicRenderer->Preferences dependency related to treeMorphFor:and:, I guess it's because the tree widget is pluggable

B. Application architecture assessment

As an architect, you now want to check the organization of your packages. You want your packages to have a good rationale for existence in the application. You want some parts of the application to be modular.

4) Please characterize **each** Glamour package as either:
Time taken: 7 min

- a provider package for external clients (package with which external clients interact)

-Browsers, -Presentations, -Tools

- an internal package (package which should not be accessed by external clients)

-Core, -Examples, -Announcements, -Helpers, -Morphic, -Tests, -Test-Morphic, -Scripting

5) Are some Glamour packages optional/modular (package can be unloaded without impacting application core)?
Time taken: 10 min

-Test-Morphic, -Tests (provided the previous one is removed with it), -Examples, -Tools
removing the tests also makes -Morphic removable

6) What are the important classes (consider incoming, outgoing, inheritance dependencies) in Glamour-Core? If possible, explain their roles.
Time taken: 15 min

GLMRenderer (entry point for the layouting)
GLMBrowser / GLMPresentation / GLMPane (parts of the abstract model)

GLMLoggedObject (superclass of a large-ish hierarchy)

7) Are there direct cyclic dependencies from Glamour-Core to another package?
Time taken: 30 min, the image ate all memory and was killed
not really, it only depends on -Announcements and -Helpers, which don't depend back (from the outgoing view)
But the incoming view doesn't agree (more packages are depended on by -Core (-Browsers, -Presentations). Probably due to extensions?

C. Detailed assessment
As a developer, you want a detailed comprehension and assessment of dependencies between classes and packages and optionally to refactor such dependencies, assessing impact of change.

First give a precise answer then provide your explanation.

What are the most cohesive packages of the application?
Time taken: 1 min
-Core, -Morphic, -Browsers (busy heads)

8) There is a dependency to DeprecatedPreferences in Glamour-Morphic. Can you detect the faulty class? Explain the dependency: do you see an easy way to solve it?
Time taken: 1 min
see 1.e, the tree widget should be standard

9) Can you explain the organization of Glamour-Morphic and its relationship with other packages?
Time taken: 2 min
it's the frontend, so it depends on lots of non-Glamour stuff, but is not depended on within glamour (besides tests)

10) Multiple packages of Glamour have dependencies to external library Mondrian. List such packages. Could you extract this dependency and make it optional (you can propose a solution)?
Time taken: 5 min
see 3.d, Shout and Morphic widgets could be made standard
Mondrian and Magritte are real needed deps I suppose

D. Personal remarks
You can provide any additional remarks about the study itself, the tasks, the tool used.
I lacked time for part C.
I didn't use the code browser
More reactivity is still needed, but event then I have an intuition that this is a tool that really wants to be interactive.
Maybe a tag-cloud view with class or surface names…
The differences of surfaces between in/out is confusing, class extensions need to be taken into account (I guess all the blue columns in the incoming of -Core are extensions)

E. Personal evaluation of Package blueprints

1	2	3	4	5

Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree

Question	1	2	3	4	5
Does package blueprint help you to understand dependencies between packages?				X	
Would you use package blueprint when you need to understand packages?				X	
Did the outgoing view help you?				X	
Did the incoming view help you?				X	
Did the inheritance view help you?		X			
Was package blueprint useful to get an impression of the most used classes in a package?					X
Was package blueprint useful to get an impression of the most referencing classes in a package?					X
Was package blueprint useful to get an impression of package cohesion?				X	