Company X Documentation

I have been asked to investigate three things:

1. User documentation
2. System design & implementation documentation
3. Intellectual property (opportunistically)

So far I have spent 2/3 of my time on the user documentation issues and one third of my time on the technical documentation. All that I have done with respect to IP issues is to quickly thumb through the patent applications. I did find application xxxxxxx on http://uspto.gov and some of the material provides a nice description of Company X technology. This report will not make any further reference to this topic.

My initial impression is very positive. Company X appears to be well-managed and well-staffed. The usual startup issues exist. I find nothing unusual for a company at this stage. More formal processes are needed, and in particular it seems appropriate to introduce the role of product manager at this time.

User documentation

The existing user documentation most closely resembles reference manuals, in the form of a RoboHelp document and PowerPoint slides. While a comprehensive reference manual is important, one does not learn a language by studying a dictionary. Instead, people try to read and understand simple documents, often by comparing the English version of the text to the same text written in a language that they can understand. As they become better readers, they attempt to read progressively more difficult documents. Eventually they try their hand at writing documents.

The RoboHelp material seems rather similar to the material in the Training Guide. I think the training guide and the PowerPoint presentations would best be combined and saved in DocBook format, then published to HTML and possibly PDF format. The RoboHelp format is awkward to work with, both as a user and an author. HTML format is better suited for most documentation requirements. The biggest benefit that HTML format provides is the ability to respond to links to specific pages, and to cross-reference between documents. Again, presenting DocBook allows you to publish documentation in multiple formats as appropriate.

The following documents should be created and maintained:

a. **Reference Manual** (the existing documentation is mostly reference material.) The biggest change in the existing documentation necessary to turn it into reference documentation would be to inform the reader where in the Company X site they would need to be in order to see the screens shown, and to explain how to get in the right state in order that the actions described would occur.

b. **User Guides** (task oriented, with a section for each user role.) The User Guide would orient the reader according to their role. The goal would be to explain the most common tasks that most users would need to accomplish; 80% of the tasks that 80% of the users need to accomplish. Roles identified by Mr. X include project management, accounting,

content, sales, marketing, technical and operations. Each section would present user role tasks to the reader using terminology and scenarios that they would encounter in the course of their day-to-day work. While it is true that different customers would have variations on the responsibilities of each role, they should be able to understand the material easily enough. The material in the user guide would detail a number of user scenarios, which ideally should be derived from the usage scenarios in the product specification; this information would also be helpful for describing testing scenarios. The difference between the existing documentation and a user guide is that the Company X User Guide would step a new user/administrator through the most common tasks that they would need to perform and explain why those tasks are necessary.

c. **Quick Start** for each user role. I suggest that this document be written after the User Guide is written, since it is essentially a brief summary of the User Guide.

d. **Tutorials** which work through common scenarios with realistic data. A good example of online training that works this way is Sun's Java Tutorial. Sun has laid out 'trails' through their documentation. Each document corresponds to a short tutorial. A trail is a sequence of related topics. Each tutorial and each trail is introduced to set context for the reader. Each page is rather short, and readers can look at a table of contents for the tutorial that they are reading and jump around. Tutorials sometimes have "Common problems and their solutions" in appropriate locations. I have read The Java Tutorial online and in printed form and after many revisions I no longer want the paper version - I only want to read the current version online.

   The tutorials should be accompanied with a dummy client site that each user would learn on and make mistakes with. A trainer could examine the history log to see what the user did, and point out what they did wrong, and offer suggestions as to what they might have done instead. Also, or perhaps alternatively, a series of movies showing an expert working through the scenarios might be helpful.

Cross-referencing between the User Guide, the Reference Manual and the tutorials would allow readers to learn more easily.

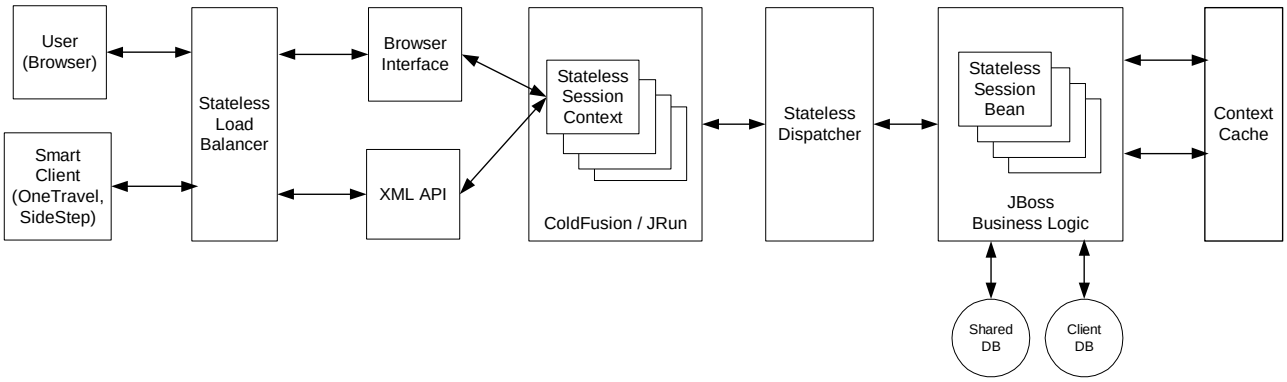
Mr. X sent out a list of approximately 200 common tasks that users need to know how to accomplish, grouped by user role. We picked a few tasks and tried to roughly estimate how much documentation would need to be written. This initial guess is best thought of as a possible indication of the order of magnitude of the work. More analysis would be necessary before a proper estimate could be generated. Assuming that the Reference Guide was to be completed before the User Guide was published; the User Guide could link to the Reference Guide as appropriate. I guessed that an average of 1.5 to 5 pages of documentation, including an average of three images would be necessary for each topic, for a total User Guide weighing in at 300 to 1000 pages. Perhaps 4 hours might be an appropriate time budget for a writer to create each page, which results in a grand total of 1200 to 4000 hours of writing, or 30 to 100 man-weeks. Remember, this is just a quick initial guess; I expect we could optimize the effort required.

A further suggestion: context sensitive help would be desirable on every page of the Company X web site. A menu of context-sensitive help should appear when a small help icon is clicked on each Company X page, like this:

- Read about this page in the Company X User Guide
- Work through a tutorial on how this page is used
- View a short video showing how this page is used
- Learn about this page in the Company X Reference Guide
- Search online help
- View online help table of contents
- View online help index

System design & implementation documentation

The existing system design & implementation documentation is barely existent and badly out of date. Effectively there is no technical documentation, which causes me grave concern. Happily, the engineering staff is very competent and has been very co-operative. With their help I have drawn up an overall system diagram (below) and have notes for drawing up a system-wide sequence diagram.



In order to estimate the work required to document the system, I measured the following statistics from the current production code base:

1086 Java files with 281,669 lines of code
172 abstract classes
1129 subclasses
202 interfaces
241 interface implementations
142 packages
291 test classes
11 HTML files
1540 Cold Fusion files with 177,149 lines of code
239 synchronized sections of Java code (measured by Mr. Y)

The Java documentation is barely existant. Although the Company X coding standards mandate the writing of package.html for each of the 142 packages, only one such file was ever written. Few class-level comments exist. Method comments do exist, but they do not provide the big picture. I did not find a single comment in the Cold Fusion code.

My approach to writing technical documentation would leverage the initial analysis done for the functional documentation. I would work with the developers to map the Java classes and Cold Fusion pages to the functional aspects of the Company X system. Scaffolding and utility classes would be separately categorized. I would interview the programmers to learn more about the high-level system architecture. Two weeks should be sufficient for me to do this, by which time I would have diagrams and accompanying prose. This information would be incorporated into Javadoc and a separate document for Cold Fusion, since it does not have a Javadoc equivalent. There would be an unavoidable impact on developer productivity while I talked to them, however I would only speak to one developer at a time, and would spend 1/3 of my time interviewing and 2/3 of my time writing. This works out to 15-30 hours of developer time for high level interviews.

The next step would be to descend a level of detail. I think the biggest need for documentation is the Cold Fusion code, since it has the least amount of documentation (none, actually), and because there is less technical backup for that material. I can't give an estimate for this now, but I suspect it would take a few months. A better estimate would be forthcoming once the overall system documentation is complete.