Functional Specification Documents:

Process and Application

by Allen Smith

1.0 When and Why do You Write a Functional Spec

1.1 What is a Functional Spec and what is it used for?

A functional spec is a document detailing the client's requirements for an application. Typically, the client has a high level view of what they want the application to do and it is the job of the Functional Spec to take that high level view and provide the details of what it does, how it is used, and what it looks like. By creating a blueprint of the application first, time and productivity are saved during the development stage. It's much easier to change functionality around in a Word document, or rework a couple mockups in PhotoShop, than it is to reprogram an entire application because the client suddenly thought of a critical piece that's very important to them. A Functional Spec can also serve as a deliverables document, thereby managing the expectations of both parties. If the spec has been written and approved by the client, then the client must renegotiate to have anything modified or change once it is in production. This saves the developer from having to "wing it" while under a tight deadline, but also means you will be responsible for delivering exactly what you agreed to deliver.

1.2 Why write a Functional Spec?

When the Functional Spec is completed, anyone who reads it should have a very clear idea about what the application is going to do and what it is going to look like. It should read very much like an instruction manual or cook book, with step-by-step descriptions of what happens and how the individual elements of the application work together.

Another key benefit of writing up a Functional Spec is in streamlining the development process. The developer working from the spec has, ideally, all of their questions answered about the application and can start building it. And since this is a spec that was approved by the client, they are building nothing less than what the client is expecting. There should be nothing left to guess or interpret when the spec is completed...and this, in a nut, explains my love affair with the Functional Spec.

Joel on Software has a very good, and in-depth, analysis of why Functional Specs should be written: http://joel.editthispage.com/stories/storyReader$212

1.3 Who writes the Functional Spec?

Typically, you're going to want somebody who is very familiar with user-experience issues and web design. Programmers should not be writing specs, and neither should marketing/business folks or graphic designers...ideally, they are busy focusing on what they do best and lending their expertise to the other areas of the project. The writer needs to be a person whose discipline is web design and user-interface, and also happens to be a very skilled and detailed writer. Writing a spec is a lot like creating an imaginary world. While writing a spec, you will spend much of your time imagining how a user might use a certain feature, and how they may navigate their way through the information. Not only do you need to map this world out visually, but you also have to write out in great detail what this world does; all the while balancing everything with the current technological limitations and business demands.

2.0 Define the Application

2.1 What is the application supposed to be?

Pretty important. Make sure you have a strong grasp on what this application is before you start anything. I often like to meet with a cross-functional team of people (marketing, web designers, programmers) if necessary, to hammer out exactly what it is we're building. This is the stage when you should be asking a lot of questions. If you're to be spec'ing out a thingamajig but you don't quite understand what a thingamajig does, sit down with the person who does and make sure you walk away knowing exactly what it is.

2.2 What is the application supposed to do?

Now that you know what it is supposed to be (hopefully more than just its name), you should now start dissecting what, exactly, it does. It's important to define what it does so you know what to focus on. You don't want to be spread too thin, or develop an application that is supposed to do everything. As in everything related to Functional Spec writing, specifics and details are important. After this phase, you should be able to define in a high level sentence the specific function the application performs.

2.3 Who is going to be using the application?

Who are your customers? Who's the audience? Any niche segment will play a part in defining the scope and functionality of the application. Know your audience, why they're going to be using this application, and how they're going to be using it.

2.4 Are there metrics?

Is it important that this application drive revenue from sales? Page views? New members? Most applications have some sort of metric attached to them somewhere. Find out what it is and keep it in the back of your mind while developing the application. In the end, if the product fails to meet its metrics it will have failed as a product. Make sure the product you're developing matches up with the metrics the customer is looking to reach.

2.5 Is there a precedent for this application?

If so, what is currently good and bad about it? What things should be kept and what should be redone? Do some research and find out if there is a similar product. You may benefit from looking at an implementation and improving on it. Don't try to reinvent the wheel (unless it simply doesn't work) and don't try to change user behavior. Instead, find examples you can look at and ideas you can borrow.

2.6 Is usability testing required?

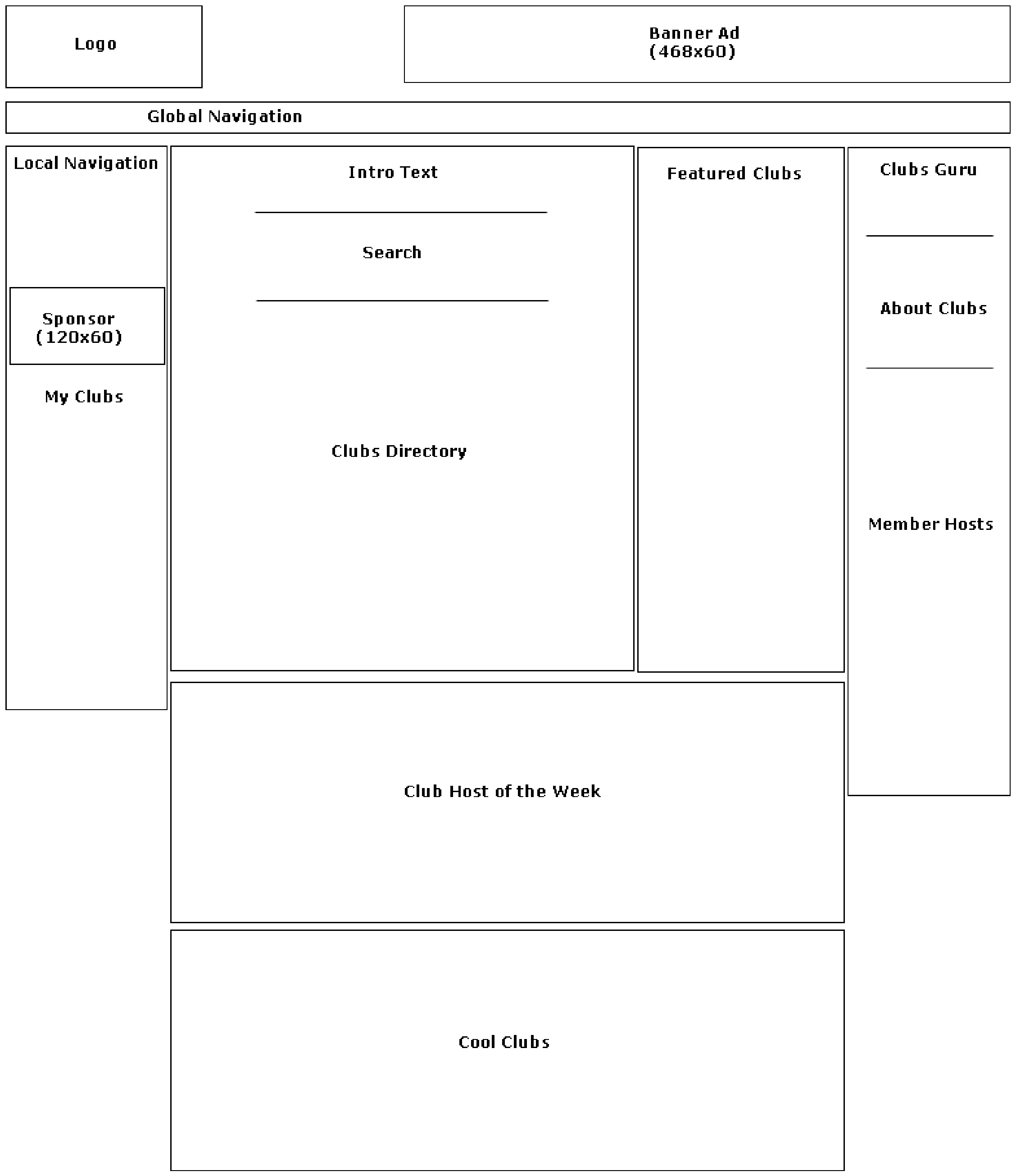
If so, do the testing at the beginning and not just at the end. When developing these applications, we want to keep the things that work and improve the ones that don't. We don't want to institute a whole new group of things that don't work while systematically getting rid of the ones that did work. Testing should, in fact, be done throughout the lifecycle of the project when possible.

3.0 Create Initial Mockups

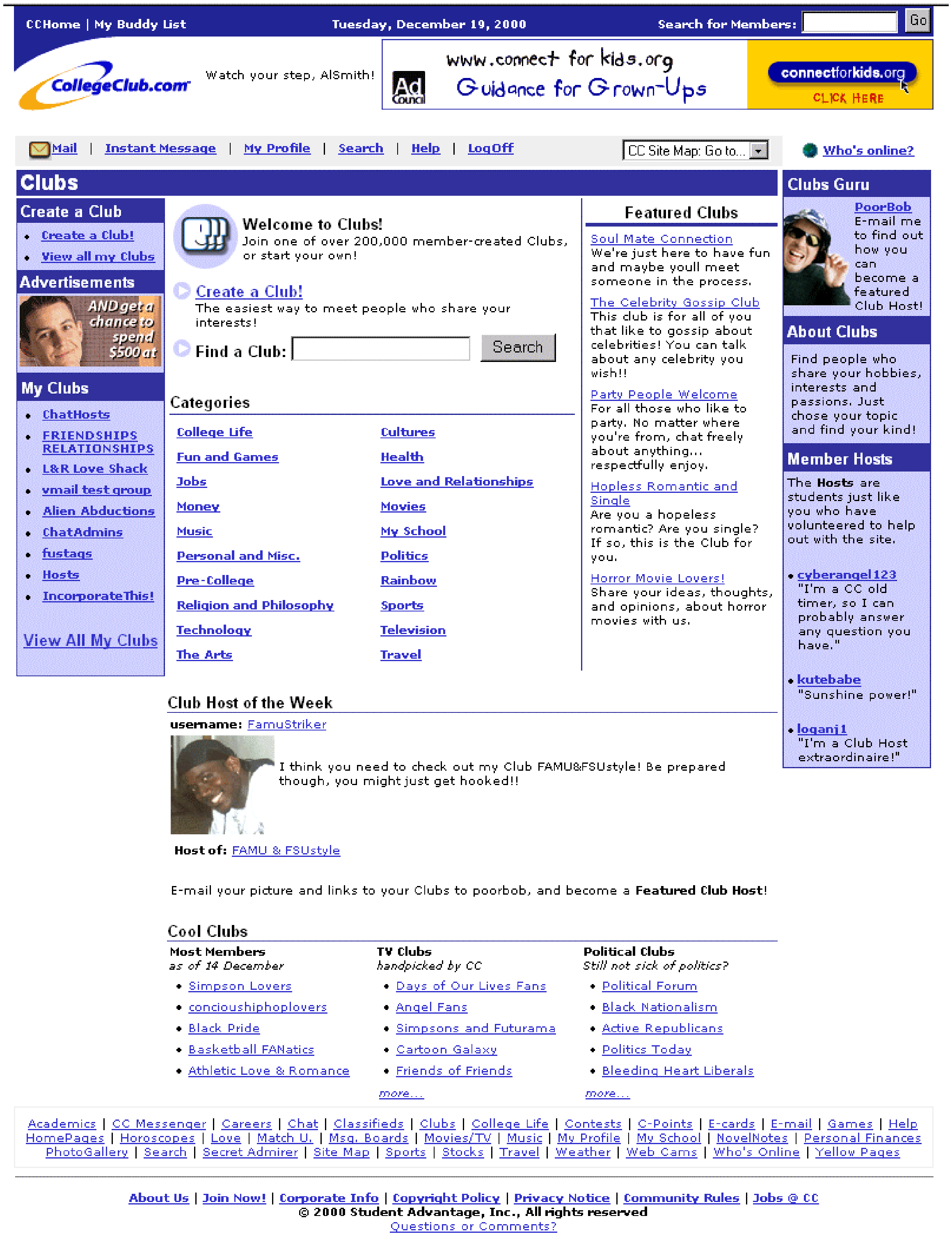
3.1 Mockup the key pages of the application.

At this stage, you don't want to start hammering out every single detail but you do need a strong frame work with which to work from. The info and its placement should be the focus of these early mockups, and not so much how it looks (colors, etc.). I consider "key pages" to be any page a user must interface with when using the application. Often, I'll start out with a written outline of the page names and then use that to start mocking up the pages. In this early stage, it's important to remain flexible and realize that you're going to be doing a lot of changing around and renumbering early on before you start building the foundation. The initial mockups do not need to be aesthetically pleasing or stylistically correct. Creating wireframes to show the placement of elements is probably more useful and time saving at this point than trying to decide on a final look and feel.

Wireframe:


Logo
Banner Ad (468x60)
Global Navigation
Local Navigation
Sponsor (120x60)
My Clubs
Intro Text
Search
Clubs Directory
Featured Clubs
Clubs Guru
About Clubs
Member Hosts
Club Host of the Week
Cool Clubs

Mockup:



CCHome | My Buddy List   Tuesday, December 19, 2000   Search for Members: [ ] Go

CollegeClub.com   Watch your step, AlSmith!

Mail | Instant Message | My Profile | Search | Help | LogOff   CC Site Map: Go to...   Who's online?

Clubs

Create a Club

- Create a Club!
- View all my Clubs

Advertisements

My Clubs

- ChatHosts
- FRIENDSHIPS RELATIONSHIPS
- L&R Love Shack
- vmail test group
- Alien Abductions
- ChatAdmins
- fustags
- Hosts
- IncorporateThis!

View All My Clubs

Welcome to Clubs!
Join one of over 200,000 member-created Clubs, or start your own!

Create a Club!
The easiest way to meet people who share your interests!

Find a Club: [ ] Search

Categories

College Life	Cultures
Fun and Games	Health
Jobs	Love and Relationships
Money	Movies
Music	My School
Personal and Misc.	Politics
Pre-College	Rainbow
Religion and Philosophy	Sports
Technology	Television
The Arts	Travel

Featured Clubs

Soul Mate Connection
We're just here to have fun and maybe youll meet someone in the process.

The Celebrity Gossip Club
This club is for all of you that like to gossip about celebrities! You can talk about any celebrity you wish!!

Party People Welcome
For all those who like to party. No matter where you're from, chat freely about anything... respectfully enjoy.

Hopless Romantic and Single
Are you a hopeless romantic? Are you single? If so, this is the Club for you.

Horror Movie Lovers!
Share your ideas, thoughts, and opinions, about horror movies with us.

Clubs Guru

PoorBob
E-mail me to find out how you can become a featured Club Host!

About Clubs

Find people who share your hobbies, interests and passions. Just chose your topic and find your kind!

Member Hosts

The **Hosts** are students just like you who have volunteered to help out with the site.

- cyberangel123
  "I'm a CC old timer, so I can probably answer any question you have."
- kutebabe
  "Sunshine power!"
- loganj1
  "I'm a Club Host extraordinaire!"

Club Host of the Week

username: FamuStriker

I think you need to check out my Club FAMU&FSUstyle! Be prepared though, you might just get hooked!!

Host of: FAMU & FSUstyle

E-mail your picture and links to your Clubs to poorbob, and become a **Featured Club Host**!

Cool Clubs

Most Members
as of 14 December

- Simpson Lovers
- concioushiphoplovers
- Black Pride
- Basketball FANatics
- Athletic Love & Romance

TV Clubs
handpicked by CC

- Days of Our Lives Fans
- Angel Fans
- Simpsons and Futurama
- Cartoon Galaxy
- Friends of Friends

more...

Political Clubs
Still not sick of politics?

- Political Forum
- Black Nationalism
- Active Republicans
- Politics Today
- Bleeding Heart Liberals

more...

Academics | CC Messenger | Careers | Chat | Classifieds | Clubs | College Life | Contests | C-Points | E-cards | E-mail | Games | Help | HomePages | Horoscopes | Love | Match U. | Msg. Boards | Movies/TV | Music | My Profile | My School | NovelNotes | Personal Finances | PhotoGallery | Search | Secret Admirer | Site Map | Sports | Stocks | Travel | Weather | Web Cams | Who's Online | Yellow Pages

About Us | Join Now! | Corporate Info | Copyright Policy | Privacy Notice | Community Rules | Jobs @ CC
© 2000 Student Advantage, Inc., All rights reserved
Questions or Comments?

I should also point out that sections **3.0** and **4.0** are very interchangeable. After writing a spec, you'll have a better idea as to what works best for you and how you should approach each project.

3.2 Focus on meeting the functional requirements.
Don't be led astray by trivial details or tasks. At this stage, it's more useful to simply make decisions and stick with them than to spend hours deciding on one item versus another. Regardless of which you choose, it has a high chance of changing before you're done writing. What's more important is that you keep your key requirements at the forefront. Make sure you're hitting these, and the rest of it will eventually fall into place. The initial mockups you're creating serve to get you started and thinking through the details. Attack the big issues first (the "must haves") and then worry about how the little things fit together later.

3.3 Review to ensure all objectives are being met (revenue, usability, etc.)
Where do these initial mockups fail to meet the key requirements? Where is there awkward navigation? Is there functionality that can be combined or streamlined? This will become a very familiar process, as you will be reviewing and editing your work many...many times.

3.4 Redesign for aesthetics and site guidelines.
Now that you have the kinks worked out of the key pages, start working on a final look and feel. Or, better yet, hand it off to a graphic designer who can work on it while you start plowing through the nitty gritty details of writing the spec.

4.0 Information Flow

4.1 Define the navigational elements.

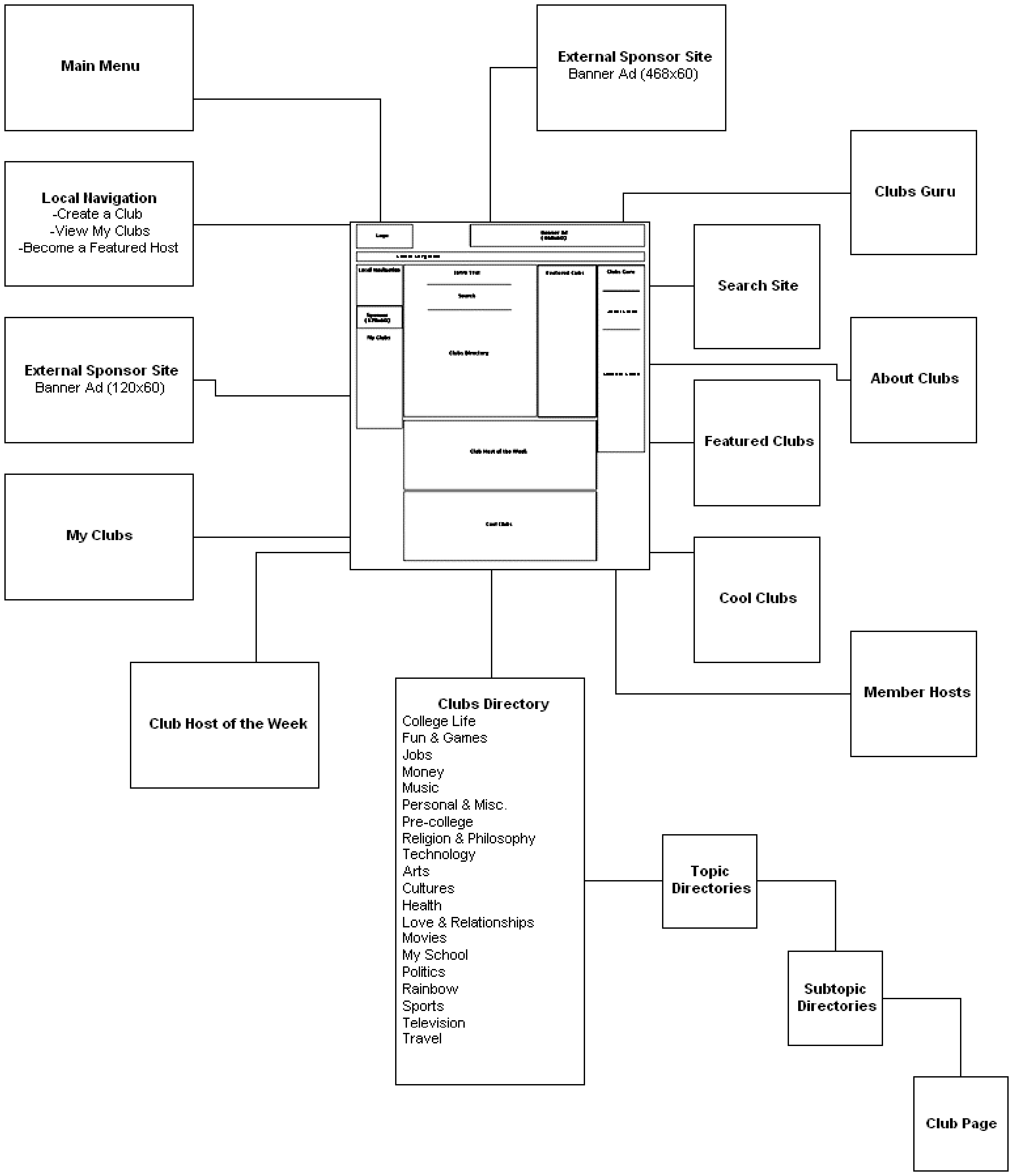
You've already started thinking about this in a small way while putting together your initial mockups, now it's time to get specific. What is the local navigation of your application? What are the actions a user can perform while using the application? Is there a global navigation (e.g., a web application that's contained within the context of a larger site, like a Message Boards application in a web portal)? How deep is your navigation going to be? How does a user find their way around?

Also, where is this navigation placed on the page? At the top? The side? Now that you know the key areas the user will be interacting with, it's time to figure out how they go about accessing these areas.

4.2 Diagram the organization of the information.

Creating flowcharts can be very handy when trying to work through a lot of information or a very deep navigational set. It's also useful to give to the programmers building the application, so they can easily see how everything is organized.

Flowchart:



During this phase, you'll want to define any classification system or directory you may need. This is especially important if the application involves a large amount of content.

4.3 Diagram the navigational path through the information.

Now that you've defined the navigational elements and created an organizational system for your information, you'll need to define how these elements tie together and how the user navigates their way through it all.

4.4 Outline all pages in the application.

The final step in this section is to identify and outline all the pages of the application. After you're done with this, you will have a complete picture of the application. The goal here is to list all of the pages needed for the application and use this as the starting point for writing the spec. Start creating a numbered index of page names that follow in a logical order (e.g., if your application involves a hierarchy, then list the "top" page first), and assign each page its own number.

5.0 Write the Spec

5.1 Each page is a separately numbered section.
This is a good guideline to follow. I use separate numbers for each page and then subnumbers for elements on the page (much like what you see here). Doing this, in conjunction with numbering the pages of the document, makes things easier to find.

5.2 Use lots of screen shots.
I insert a specific element screen shot after each element section or group of elements. At the end of the section, I insert a screen shot of the entire page.

5.3 Use correct grammar and vocabulary.
Short, descriptive sentences work best. Never use "I" or slang. Break out complex actions into bullet points. Make sure every action that can be performed on the page is documented, and always run spell check.

5.4 Text that appears on the page should be written into the spec with a different font face and size.
You want to differentiate between your spec and the text that's supposed to actually go into the application. To make it as least confusing as possible, I use a different font face and size. Additionally, I always explain that the following text is to be read by the user.

5.5 Use consistent terms.
If you use "pull down menu" in the first part of your document, don't start using "drop down menu" (or any other variation) in other places. Keep all terms consistent, and as in line with the standardized industry terms as possible.

5.6 Use a consistent style.
Just like when you read a book it uses the same font face, size, and margins throughout, your spec should also follow a format. That format can be decided on by you, your organization, or you can follow mine. I've set up a few style guidelines that are consistent throughout all of my specs.

Here is the formatting I like to follow while writing my spec:

- **Font face** **-**Times New Roman
- **Size** **-**10pt
- **Headings** **-**Underline, Bold, and use 12pt Times New Roman
- **Spacing** **-**I space everything based on half inches (following the MS Word guides at the top). So, section headings are right against the margin and subsection headings are indented a half inch. The body of each subsection is indented another half inch and any bullets under the body text are indented a half inch and so on.
- **Numbering** -Whole numbers indicate a Section, subnumbers indicate a Subsection. So **1.0** would indicate an entire Section (probably the main page of the application/site), while **1.1** would indicate an element of that section.
- **Application Text** -Text that's used in the application/site is 12pt Verdana. I use a different font face to make it easier to distinguish between spec text and the text that is actually going to be read by the user.

6.0 Rewriting and Editing

6.1 Editing...and editing...and editing

Rewriting and editing are half the battle. The first half is laying the groundwork and building the infrastructure, while the last half is polishing and checking.

When setting the timeline for your completion of the spec, always try to add three to four more days than you anticipate needing. In my experience, something always comes up during the editing process: I've forgotten something, someone now wants to add a new element, details need to be filled in, etc. Since so many people from so many different areas of expertise will be reviewing this document, it is important that it is perfect: no spelling errors, no grammatical errors, no logical errors, and no mock up errors either. Run your spell check and have someone proof read it for you (if it's possible).

Try editing your document using a printed version and a pen. While editing, try to go through the entire spec and mark up the corrections with a pen. After you're done, go back to the computer and start making the electronic corrections. You're far more likely to miss items if you're simply correcting them on the fly. Taking the time and focusing on editing will pay off with a much stronger document in the end.

6.2 Check your index.

Simple thing, but necessary. You'll usually end up renumbering and editing your index many times before you're done. To avoid confusion, update your index after each section you edit and check to make sure your index follows a nice, logical order.

6.3 Edit from beginning to end at least three times.

I don't know why, but it seems like three times is all it takes. The first pass usually involves a lot of rewriting and renumbering. The second pass is typically much lighter and may catch some of the obvious things you missed, and then the third time is merely to polish and catch any small details that were missed. Between each round of editing, implement your changes in the electronic document and then print it out. By all means, feel free to edit more if you feel it's necessary.

6.4 Have someone proofread it for you.

This is pretty optional, since I've never had the benefit of having my specs proofread...but, it is the ideal. Whoever is proofreading should be a person who has a strong grasp of grammar and spelling, but is not involved with the project (since you'll have enough people involved with the project reading it).

6.5 Review.

Whether you're writing a spec for your company or for a client, you're going to have a series of people reviewing your spec to make sure they understand what's going to be built. Expect changes and be prepared to give reasons for why everything is written as it is. By the time you take your spec to any clients or colleagues, it should be airtight. Part of the review process is working *with* your client or colleague to come up with the best solutions. If they feel that the application will not meet their needs as spec'd out, find out why and offer solutions as to what can be done to meet their needs. After the review, take your notes back to the computer and make your changes while they're still fresh in your mind. Be sure to change your mockups and all related text, double check to make sure your changes don't affect other functionality (and if it does, remedy that), and then double check your index. Prepare your Functional Spec for another review.

Many times, the people reviewing your spec do not have the time (or inclination) to read through the entire document in detail. It is often helpful to print up screenshots of the application and pass those out and then refer to the spec as needed.

7.0 Resources

Although there are a scant few sites out there that deal with Functional Specification Documents directly, there are many resources you can use for various aspects of spec writing. The following sites are just a handful of the really good informational sources out there relating to (in one way or another) the discipline of creating Functional Specs: usability, visual design, architecting information, web design.

xblog :visual thinking weblog
http://www.xplane.com/xblog/

elegant hack :information architecture
http://eleganthack.com/reading/index.html

peterme :interface design
http://www.peterme.com/index102698.html#Interface

metagrrl :Design Process in Online Environments
http://www.metagrrrl.com/design/proposal.htm

Usable Web :links to web usability articles
http://usableweb.com/topics/000505-0-0.html

A List Apart : web design
http://www.alistapart.com/index.html

Surfstation :web design
http://www.surfstation.lu/

Webword :usability
http://www.webword.com/

Use It :usability
http://www.useit.com/

Design is Kinky :graphic and web design
http://www.designiskinky.net/

Kaliber10000 :web design and all things cool
http://www.k10k.net/

Joel on Software :good look at writing functional specs
http://joel.editthispage.com/stories/storyReader$212

Builder.com :web design
http://builder.cnet.com/webbuilding/0-3880.html

Glassdog :web design
http://glassdog.com/design-o-rama/

Webmonkey IA Tutorial :information architecture
http://hotwired.lycos.com/webmonkey/design/site_building/tutorials/tutorial1.html

CoolHomepages :web design
http://www.coolhomepages.com/

Digital Web Magazine :web design
http://www.digital-web.com/

webdesign-l :web design e-mail list
http://www.webdesign-l.com/

SitePoint :web design
http://www.sitepoint.com/

Photoshop Workshop :Photoshop tutorials
http://rainworld.com/oe_99/forge/ps_tut/

Web Design Start :web design
http://www.webdesignstart.com/

Lynda.com :web design
http://www.lynda.com/