**Why Writing Software Design Documents Matters**

If you're an experienced developer, you've probably progressed from being a humble tester to a senior developer, and if you're a freelancer, you've made another leap, perhaps the biggest of them all, when you started working with clients directly. Some clients aren't in the software business; they're in an entirely different industry that needs a piece of software, and they don't have a clear and precise vision of what they want from you. This is a far greater challenge than it appears, and here's what you can do to improve client communication and project documentation.

[](https://www.toptal.com/resume/chris-fox)

AUTHOR

[**Christopher J Fox**](https://www.toptal.com/resume/chris-fox)

Chris has a BSc and 25+ years of development experience, including senior engineering positions at Microsoft and RealNetworks.

**1.6K**SHARES

Congratulations, you’re a [competent independent developer](https://www.toptal.com/freelance). From your humble beginnings, perhaps working as a tester, you’ve progressed to a team developer, then a senior developer, and now you’ve made another leap, the biggest of them all, to working directly with clients.

But where the other transitions were linear, this last one was exponential. While in the past you got your marching orders from an employer that worked with clients or was itself in the software business, now all those responsibilities that were once distributed between expert-testing, program management, etc., are all yours. And now you’re working with clients who are not in the software business; they’re in another business that needs a piece of software, and they don’t have a clear and precise vision of what they want from you. This is a far greater challenge than it appears.

*\*Note:\* Here, I’m describing smaller clients who want a one-man army from their developer. It’s not the only route a freelancer can take, and those aren’t the only clients we work with at Toptal, but it’s the route I enjoy most. Of course, if you’re working in a team and not*[*on your own*](https://www.toptal.com/freelance/how-to-work-remotely-and-still-be-the-best)*, some of the below won’t apply. For example, if you’re using*[*Agile methodologies*](https://en.wikipedia.org/wiki/Agile_software_development)*or*[*Scrum*](https://en.wikipedia.org/wiki/Scrum_(software_development))*, you’ll probably want to structure your milestones slightly differently.*

From your humble beginnings, perhaps working as a tester, you've progressed to a team developer, then a senior developer, and now you've made another leap, the biggest of them all, to working directly with clients.

You’ve all heard about the supreme importance of communication. You can’t work by getting a few sentences of terse description over Skype and saying *“See you in three months when I’m done.”* You have to be in communication with your client and at every stage of your work make certain that you have congruent ideas about the objective, because it’s rare indeed that a client will send you wireframes and a detailed functional specification. You will get a very general idea of what the software is supposed to do, look like, and flow. If you write an application based on the cursory description you usually start with, there’s almost no chance that your client will be happy with the result. At each stage, you must iterate your way closer to agreement.

You can't work by getting a few sentences of terse description over Skype and saying "See you in three months when I'm done."



Having worked for years at companies that were themselves in the software business, where everyone on the team was from the same culture, spoke the same native language, worked in the same hallway, met each other daily, etc., it was noteworthy that the company *still* didn’t get what it wanted half the time. Make no mistake: the challenge here is enormous.

Why Software Design Documents Matter

So, when you take on a new project, **before you even open Xcode or Visual Studio, you need to have clear and agreed-upon design goals**. And these goals should be established in a specification document. If the client hasn’t written one, you should write it, and submit it to them for review before you even open your IDE. And **if you encounter a client who says, “We don’t have time for design documents”, candidly, you should walk away from the project** because you have trouble ahead. The specification need not be particularly lengthy; it can be just a few pages, but at the very least it should lay out the user interface, include wireframes (if there’s a UI component), and set completion milestones.

*Without* this document, you’ll end up in a loop of acrimonious equivocation, clients disputing what they told you or what you told them, angrily sending cut-and-pastes of previous communications, interpreting and arguing until the time comes when the client demands that you make changes to bring the application into conformance with “what they actually asked for,” and expects you to make those changes without pay.

*With* this software design document, you’ll have an answer to any such quibble: when disagreements arise, you can refer to the specification which the client agreed to and signed-off on, pointing out that you have fulfilled it to the letter. Instead of angry arguments, you’ll make amendments and clarifications to the document. If anything, the client will apologize for letting the imprecision slip through in the first place.

We all want satisfied clients. We all want a friendly working relationship. And we all want the pride of a job well-done. But these can’t be achieved if there’s any vagueness whatsoever about what the job actually *is*. If your client says that a design document is too much extra work, it’s your job to explain to them that the *real* extra work will emerge when revisions need to be made due to some sort of misunderstanding. If the client *still* insists that you advance without such a document, you should accept the fact that you have an unworkable relationship and walk away.

What Should the Software Design Specification Actually Specify?

**At the very least, it should be a description of the desired application, criteria for completion, and milestones.** Remember, you are sharing what is best described as a requirements and function document, not an implementation specification. And unless a specific implementation is a stated client objective, how you make it work is up to you.

User Interface

Most projects are applications, not libraries or frameworks. But if you happen to have one of these as a deliverable, count yourself lucky because **the user interface is far and away the most problematic component of your design document template**, and almost always leads to misunderstandings. Many clients will send you perfect illustrations created in a graphic editor by a graphic designer who is not a programmer. But the problem is: these illustrations say nothing about animations, control states (e.g., Is this button disabled? Does it disappear when unusable?), or even what actions to perform when a button is pressed.

Many clients will send you perfect illustrations created in a graphic editor by a graphic designer who is not a programmer. But these illustrations say nothing about animations, control states, or even what actions to perform when a button is pressed.

Before you start writing the code behind these illustrations, you should be able to answer all of those questions. Specifically, you should know:

1. Are controls always visible and/or enabled? Under what conditions do their states change?
2. Looks like a bitmap—is it a button?
3. What transitions occur between these states and views? And how should they be animated?

If it’s up to you to generate the UI for the client’s concurrence, do the same in reverse: use a wireframe tool and create a complete set of screen layouts, including any variants that the views show in different application states. This can be exhaustive and tedious work, but you won’t regret it—it can save you from re-writing huge amounts of code and re-creating interfaces due to a minor misunderstanding with major implications. If you’re creating a dual application (e.g., for both [iPhone](https://www.toptal.com/designers/iphone/portfolios) and iPad), create separate wireframes for both.

Screen dimensions are important too. There are (as of writing) three sizes of iPhone screens. Separate wireframes for 3.5” and 4” screens are probably excessive, but you may have to make them; in most cases, you can simply change proportions.

If your client supplies you with graphics, make sure that they are correctly sized with the proper aspect ratios; morphing any bitmap that has text or objects (like circles) will introduce distortions. If they don’t match, tell the client to re-create them with matching sizes. Don’t presume that you can stretch a 3.5” splash screen into a 4” splash and just roll with it.

Functionality

Key questions to ask in the application design document:

* What does the application do, and how quickly does it do it?
* What are possible failure conditions and how are they handled?
* What one-time operations are done at the first execution (i.e., after installation)?
* If the user creates entries of any kind (e.g., bookmarks), what are the limitations?

Generalize these ideas, and be as detailed and thorough as you can—because errors or misunderstandings here will mean rewriting code.

Milestones

Your specification template should layout clear milestones. If your client writes the functional and user interface design, you should subsequently agree on a set of milestones. Sometimes these are billing thresholds as well, but at the very least they provide a clear metric toward completion. Milestones may be in terms of functionality and/or components; they may even be separate applications if the gig involves a suite of deliverables. When possible, milestones should be approximately equal in duration.

Software Design Specification Example

Here, I’ll layout the example structure of a proper design document. Of course, this template should be adjusted as-needed. For another example, see [Joel Spolsky’s sample specification](http://www.joelonsoftware.com/articles/WhatTimeIsIt.html), based on [this write-up](http://www.joelonsoftware.com/articles/fog0000000035.html). He approaches the document slightly differently, but shares a similar sentiment.

Statement of Goals

Include a short paragraph describing the project and its intended audience.

Functional Description

What does the application *do*? What application states (high-level descriptions of core user scenarios) will the user encounter?

For example, your functional description might look like:

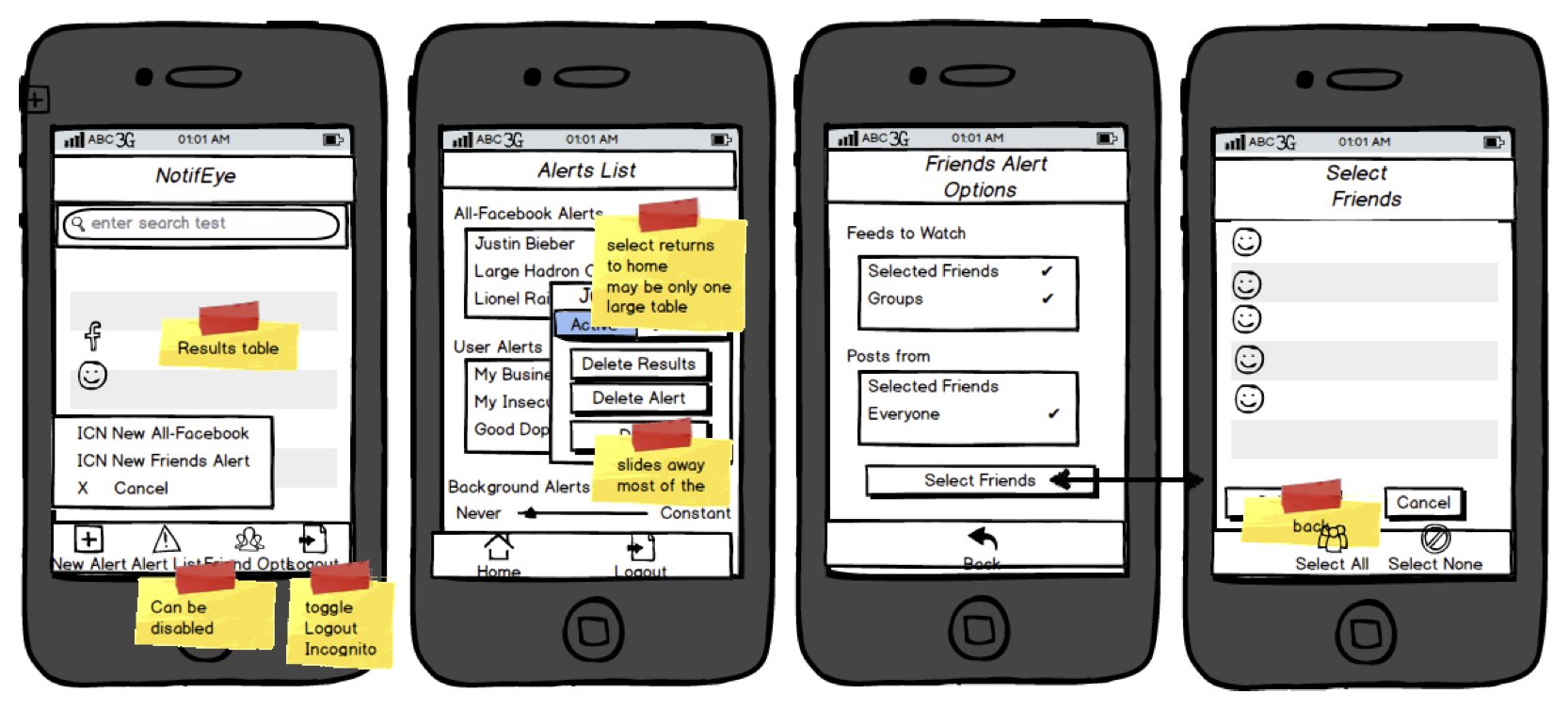
* First Run
* Creating a New \_\_\_\_\_ (game, search, etc.)
* Operations
* Background and Foreground Behavior

User Interface

Include wireframes for each page, with detailed descriptions of:

* Each control, including states (enabled/disabled/highlighted) and operations.
* Supported orientations and transitions between them.
* Functionality represented.
* Error handling.
* Dimensions and constraints.

Here are the wireframes related to my latest iOS app, NotifEye:



If you’re interested, I made these mockups using [Balsamiq’s wireframing tool](http://balsamiq.com/).

For example, your [UI description](https://www.toptal.com/ios/ios-user-interfaces-storyboards-vs-nibs-vs-custom-code) might look like:

* Navigation Bar
  + Left navigation control: return to home page
  + Title bar: current screen or operation name
  + New button: create a new Thing
* Table View
  + Section 0: Section title
  + Section 0 rows:
    - Row control 0 (e.g., image)
    - Text Line 0
    - Text Line 2

Milestones

As described above, deadlines for completion and expected deliverables.

For example, the milestones section in your design document template might look like:

1. Facade Application showing screen and with temporary transitions and example images/text
2. Communication Protocol: application connects to network/server
3. Functional Milestone 1: …
4. Alpha Application (with full functionality)
5. Stability
6. Release

Making Sure Software Documentation Remains Relevant

I don’t mean to imply that the design phase is over once you and your client have agreed upon a specification document. There will always be details that neither of you had considered, and both you and the client will, while looking at the intermediate results, encounter new ideas, design changes, unexpected design flaws, and unworkable suggestions.

The design will evolve, and the changes should be captured in your document. In my 25 years of experience, I have never once worked on a project where this didn’t happen—and that includes my own applications (i.e., where I was my own client). Even then, I created a design document with detailed specifications, and adjusted it as necessary.

Above all, keep in touch. At least several times a week, contact your client, report on your progress, ask for clarification, and make certain that you share identical visions. As a litmus test for your communication, try and ensure that you and your client give the *same* answers to these three questions:

1. What was the developer just working on?
2. What is the developer currently working on?
3. What will the developer work on next?

UNDERSTANDING THE BASICS

**What does the acronym SDD stand for?**

SDD stands for software design document or software design description.

**What is a functional design document?**

A functional design document describes a software product's capabilities, appearance, and functions it needs to ultimately perform. Design documents are also referred to as functional specifications or functional specifications documents (FSDs), or functional requirements specifications.

**What is a high-level design document?**

A high-level design document (HLDD) describes the architecture used in the development of a particular software product. It usually includes a diagram that depicts the envisioned structure of the software system. Since this is a high-level document, non-technical language is often used.

**What is in a software design document?**

The software design document (SDD) typically describes a software product's data design, architecture design, interface design, and procedural design. The content and organization of an SDD is specified by the IEEE 1016 standard.