Tips for Writing Software Development Specs

Revised: February 2018

This document provides tips and examples of successful techniques for writing end-user requirement specifications ("specs") for a software development project.

Specs vs. Research Specs: What's the difference?

This document specifically relates to the detailed technical spec that will be used by a programmer to code the software changes. If you are at the stage of trying to gain support for a proposed project, or if you need to get estimates of the amount of effort involved for the purpose of evaluating costs, a ***Research Spec*** *may be more appropriate.*

We often start our custom projects for credit unions, as well as our major development projects, with a research spec overview. That way we can get buy-in and make sure everyone is on the same page and agrees to the project scope, before spending the time and energy on detailed technical specs. For instructions, refer to the Research Spec template.

Contents

What Makes a Good Spec?

Writing a project spec requires the ability to imagine and clearly explain a big-picture, high level overview of a project, as well as get down to the nitty-gritty details that define how the software will behave and what the end-user experience will be.

Key Components Your Spec Should Contain

Part 1:	Introduction/Project Overview	This is the elevator speech that defines the overall scope of the project, introduces key concepts, and explains the overall purpose and goals of why the project needs to be done. Imagine explaining your vision to a colleague at another credit union. Items to include in this section: • Purpose and goals of the project; expectations from an end-user's standpoint • Timeline and general deadline, if any (i.e., reg. change date, etc.) • Outline of any future enhancements anticipated in the long term • Suggested content for the Release Summary (marketing spin)
Part 2:	Program Changes Overview	A quick overview of what new tables, new columns. or program changes, if any, are anticipated with the project.
Part 3:	Changes to Screens, New Screens, Workflow, Field Specifications	This is the meat of the spec and includes mockups of how the screen(s) should look, changes to screen layout, instructions on workflow, detailed field specifications, descriptions of how all command keys should work, and so on. Every screen throughout CU*BASE that will be affected must be detailed. Do Account Inquiry screens need to change? Which ones? How? Do the main Phone Op, Inquiry and/or Teller Verify

		Member screens need to be modified? How? What about maintenance screens such as Update Membership Info or Update Account Info? Which screens and how? All screen changes must be clearly mocked up and included in the spec document itself (not screen prints with handwriting on them). These sections should also include specific messages and verifications the program should use to prevent common user mistakes. We call these edits and they are generally instructions like this: "If Activate... is set to Y, also require a valid, eligible account # to be entered."
Part 4:	Changes to Reports and Report Selection Screens	If a report is involved, whether a new one or changes to an existing one, include a mockup of the changes to the report selection screen. Also include a sample of the printed report with specifics on how the data will lay out, mocked up with sample data.
Part 5:	Tool Changes	A list of new tools that might be needed, with suggested tool title, description and program information.

Consider the following examples:

Sample Project Overview

Instead of:	Try something like this:
Create a debit card round up program to round up purchases to the nearest dollar and deposit them to a savings account.	This project represents our take on the KeepTheChange© program from Bank of America and similar programs that help members save by automatically rounding up each purchase made via a debit card and depositing the extra money into a designated savings account. The most important element of this product is one that might not be obvious at first glance when you read marketing materials from BofA and other financial institutions: **The round up process is *not* posted immediately, on a per-transaction basis.** Instead, a daily process will calculate the round up amount for all debit card transactions posted that day and post a single transfer from the checking account to the savings account. To put it another way, if you go to Macy's and spend $94.73, the transaction that posts to your account will be exactly $94.73, not $95.00. At the end of the day, that extra 27¢ (along with any other amounts calculated on other purchases throughout the day) will be transferred from your checking to your savings account.

Sample Programming Instruction

Instead of:	Try something like this:
Write out a Tracker record	• When the user presses F10 to save the CTR at any point (every time it gets edited, whether the first time through or later on), write a note to the member's Audit Tracker (Type AT) - don't need to display any Tracker screens for user input or provide any confirmation message; just create the note behind the scenes: • Use Memo Code CT (code description "CTR form") • CONVERSION NOTE: Doesn't look like this is one of our standard ones, but will still need to verify all CU configs for this memo code; please advise if this is already used by any CUs, then arrange for a conversion program to create this code in all CU libraries upon implementation

	• Use the following text in the Tracker note: “CTR form created / modified” and include the date and time, Employee ID, and CTR form number.

See the difference?

Other Things That Lead to Good Specs

- Use examples to illustrate complex calculations or scenarios that the software is intended to handle. If you are mocking up a screen sample, put in some realistic data to help illustrate how it will look and be used.
- Include more detail than what you think the programmer needs. Don't make assumptions. Better to be too clear than to leave something out.
- If you are making a change to an existing screen or feature, showing both a before and after view can make it much easier to see what is changing, especially if there are a lot of new features or elements are being rearranged significantly.
- Remember that the project must be testable, so thinking about how you might test the changes can help you write clearer, more complete instructions in the specs.
- Include notes about converting existing data to any new uses/formats.
- Include documentation/end-user notes to explain difficult concepts or to justify a particular method selected (i.e., why did you do it that way?).
- Keep the end-user in mind and anticipate their questions to make the workflow, field labels, and layout as intuitive and easy to use as possible. Longer field labels are better than short ones if they eliminate questions and keep someone from having to open help or call a CSR.
- Follow standard screen layout conventions for consistency and to make the software easy to learn. (Refer to the User Interface Style Guide for current standards for CU*BASE host screen elements.)

Tips on Constructing a Spec Document

Using Our Word Template

You can organize and format your spec document any way you wish, but if you want a sample layout to get started, start with the Project Spec template that our designers use. Then simply copy, delete, rearrange, and modify the sample content from this template as needed.

Techniques for Making Host Screen Mockups

Use the method that our spec writers do to mock up a host screen change. If you right-click on the title bar of any CU*BASE GOLD screen and choose Show Emulator you'll see what a host screen looks like. By copying this plain-text data into a properly formatted Word document, you can manually type field labels and rearrange screen information as needed to illustrate how the user interface should look. A sample blank screen is provided in the Word template. This format is particularly handy because it makes it easy to show field lengths.

If you are modifying an existing screen, follow these steps to pull the existing screen into your spec:

1. Navigate to the screen in CU*BASE GOLD

2. If you haven't already, right-click on the title bar and choose Show Emulator to view a separate window showing the host layout
3. With your mouse, click and drag to drag a rectangle around the entire colored area of the screen. When you let go of the mouse, everything in the outlined area is automatically copied to your clipboard (you don't have to choose copy)
4. In your Word document, position the cursor where the screen should appear
5. Apply the **ScreenShots3** style
6. From the Edit menu, choose Paste

Other Helpful Resources

If you want to see some samples of specs from recent projects, contact Dawn Moore at dmoore@cuanswers.com. We can pass along the spec from a specific project or just provide a sampling of several different-sized projects, including some that have been implemented as well as ones that are currently in development.