Five Steps to Great Software Requirements

For a company that makes software, software development is like the engine that makes the whole company go. And software requirements are like the fuel that feeds that engine. The better the fuel, the better the engine runs. Follow these steps to create high octane requirements that will kick your software into high gear.

1 Cover Lots of Ground

Good requirements touch on many aspects of your software. Make sure you hit many functional areas of your product. This means not just functional requirements, but technical and architectural ones. Also, think about ideas that will improve operational efficiency. Bring in ideas from a wide range of perspectives, including ideas from customers, subject matter experts, partners, Sales and Marketing, R&D, consulting, training, customer support, and finance.

2 Stick to the Facts

When writing requirements, people often get bogged down in opinion, argument, and justification. This does not provide developers the instructions they need to build good software. Instead, make each requirement a plain, factual description of what's needed.

Not a Good Requirement

"The marketing campaign process is too difficult and complicated, make it easier!"

A Good Requirement

"Modify the campaign process to four steps: 1) Plan; 2) Activate; 3) Monitor; and 4) Close."

3 Organize

Any list of requirements quickly becomes long and hard to manage. Organize your requirements into different categories or types of requirements. Choose a set of categories that reflect what is most important for your product. For instance, choose major functions plus other top considerations: Quotes, Orders, Shipments, Invoices, Payments, as well as Services, Architecture, and Channels.

In addition to categories, define the main benefit or return each requirement provides. For example: Salesworthiness, Competitive Parity, Feature Leadership, New Market, and Cost Savings are all valuable benefits. When the team understands the benefit you're aiming for, this guides design and coding decisions.

This benefit or return can be a much more realistic, intuitive, and achievable way to measure your return on investment (ROI) than trying to estimate a dollar figure for ROI on individual requirements.

Prioritize

There are always more ideas for what to build than there is capacity to build them. If you have collected a long list, this is good. But now you have to sort this list based on the relative importance of each individual requirement, so you can focus on the most important ones.

Aim for a simple system: High, Medium, and Low, with perhaps an additional Top priority for the handful of most important things your software needs.

Prioritizing is usually the hardest step, because you tend to veer towards giving every requirement a High priority. Take a cold, hard look at each requirement and just how important (or minor) it is compared to all the others on your list.

Start by making each requirement a Medium priority. Then consider whether it truly falls in the middle, or whether it is actually not that important (Low) or a High priority. Then go back through the list and pick out the handful of Top priorities.

It's likely your development team won't even be able to complete all the High priorities for the next release. You can also throw in a few Medium and Low priorities if they are particularly fast and easy to do (when you Plan the Latest Release, below) in conjunction with work on a closely related High priority.

Measure

Finally, estimate the effort needed to develop the requirement. Try to keep this simple, expressing this effort in terms of man-days or man-weeks, or if that's too difficult, in terms of Small, Medium, and Large. Use the same measure (weeks, days, or hours) for every requirement.

Now Plan the Next Release

Once you've completed these five steps, you've reach a hard-won milestone. Congratulations! Your requirements are now ready for you to pick which ones you will develop for the next release.

Try sorting requirements by topmost priority first, and within each priority, by development effort from easiest to hardest. Go down this list until you have enough requirements to fill the next development cycle.

Then go through the remainder of the list to see if there are must-have requirements that need to be substituted in for ones that were higher up the list. You might forego your topmost requirement because you can get five others done in the same amount of time. The tradeoffs won't be easy, but in the end you have a clear set of instructions to provide to the development team.

This is the high-octane fuel that keeps your software development engine firing on all cylinders to drive your product and your company forward at top speed.

Take the Quiz
How Does Your Company Measure Up?

How well are you doing software requirements today? Use the chart below to score yourself.

Check the Applicable Column. Do You:	**A**	**B**
Gather requirements from a couple individuals (A) or multiple people (B)?		
Collect requirements from a couple job roles (A) or from many functional areas (B)?		
Use wording that is factual and instructive (A) or write things like "make it work better" (B)?		
Write each requirement in a consistent format: "Do this and here's why" (A) or not (B)?		
Organize and sort your requirements into categories (A) or not (B)?		
Specify for each requirement a benefit or return from a standard list (A) or not (B)?		
Assign a priority to each requirement (A) or not (B)?		
Have far fewer higher than lower priority items (A) or have mostly higher ones (B)?		
Estimate the development effort for each requirement (A) or not (B)?		
Use the same measure to estimate each requirement – days or weeks) (A) – or not (B)?		
TOTAL for Each Column		
Total for Column A + (Column B x 2) = Your Score:		

10 – 12	**13 – 17**	**18 – 20**
Good	**So-So**	**Poor**

Note: Not sure which answer is best? That means you choose B.

Do You Need Help With Software Requirements?

I employ a highly effective method that helps companies develop requirements quickly and efficiently so that you can keep moving your product forward. Your development speed becomes an advantage over your competition.

If you would like to know more, send me an email or text/call me at +1-215-605-1088 (US Eastern Time) and let's talk about your situation and how to make it better.

– Jacques Murphy, ProductManagementChallenges.com