How to Write Requirements

June 9, 2020

>> DEBBIE LUCHENBILL: We are now hosting the Zoom track for track two today and I am excited to also think Equinox for sponsoring the closed captioning and are fantastic Captioner, Karen. And am pleased to introduce to you Andrea Buntz Neiman. Who is sponsoring How to Write Requirements or You Can't Always Get What You Want but You Can Get Close.

>> ANDREA BUNTZ NEIMAN: I don't seem to have the ability to share my screen, I am not seeing that.

>> DEBBIE LUCHENBILL: Try now, I just stopped sharing mine.

>> ANDREA BUNTZ NEIMAN: Pin video to second screen... hm... Technical difficulties, is probably my fault. I'm just not seeing an option anywhere to --

>> DEBBIE LUCHENBILL: All panelists should have sharing. I can try to make you the host and see if that works.

>> ANDREA BUNTZ NEIMAN: Sure, why not [Laughter].

>> DEBBIE LUCHENBILL: There you go.

>> ANDREA BUNTZ NEIMAN: I have more options --

>> DEBBIE LUCHENBILL: Mine was a big green button next to chat.

>> ANDREA BUNTZ NEIMAN: It is there! how many librarians does it take to figure this out. Okay, I think we are good, can you all see my screen, how to write requirements?

>> DEBBIE LUCHENBILL: Yes.

>> ANDREA BUNTZ NEIMAN: Sorry about that. Let's roll. For those of you who don't know me, on the project manager for -- software development at Equinox and longtime Evergreen member. Today we'll talk about some of the best practices for how to write software requirements. If you attended might talk in March with the cataloging working group this is pretty much the same - if you attended that, this is going to be a lot of the same content.

Why are requirements important? I will start with a quick story. A lot of us probably remember this happening in real time, but in 1999, the Mars orbiter burned up upon atmospheric entropy the cause of this was traced to Lockheed Martin's failure to convert English units to metric units. In its software. The software was assuming metric units. The error combined with misplaced assumptions to be fatal for the spacecraft. NASA followed its undergarments but Lockheed Martin did not. However, NASA seemed the requirement had been read and did not double check. What is the lesson?

The lesson is there are negative consequences to port requirements mitigation unfortunately in the library world these consequences are not nearly so dire but in understanding of that reform is process and how to work through it is important if you are involved with library software development.

Here is a brief outline of what I will talk about-- it much going through all these things. I will give you time to ask questions at the end. If you're not tired of hearing about requirements.

First, when you think beyond bug reports. Bug report surg a great way to start making software better, creating them, commenting and adding heat and bug fixes versus full development is a degree of difference rather than kind. Software-- sometimes developing items are more wants or wish list items that needs but that can be a there is differences between the two in terms of effort, process etc. Large improvements require larger depth and more development.

Fitting new development into a mature software product like Evergreen can be quite tricky. There's a lot of potential pitfalls and not one person knows everything, not even Galen. Larger development means more decisions and more avenues for misunderstanding. How do we navigate this and mitigate the misunderstanding?

Steps in the development project can be broadly grouped into two sections. There is the defining and researching section and the coding and execution section. For this presentation I am going to be talking about the first half of the process. Development starts from an idea or problem needing a solution.

Why do you want the new feature? What problem are you trying to solve? Requirements which is what I will get more in-depth about today is about what you specifically need the future to do. The specification takes the "why" and the "what" and translates it into "how" the Equinox specification can involve detailed interfaces, database changes upgrade considerations etc.

Once you've done the early work, the coding can actually begin the development phase. This is followed by testing, internal testing, external texting, community testing and the community acceptance and release process -- that is whole sensation in and of itself and I will not be getting into that.

What do I mean by iteration? First of all, there's a caveat here -- iteration is desirable but not always possible. Each of the six steps in the preceding slide and sub steps within them, should have some kind of internal feedback loop of discuss, revise, refine before you move to the next step in the process. Ideally, this involves us of, mitigation between libraries, library workers, librarians, and developing staff.

One of the biggest problems I see in requirements is over specifying technical implementation details. That is the "how" partnered requirements about what. What does your user need to be able to do, anyway I will frame this that I will talk about a couple slides is through user stories? When you talk about what you want software to come abstraction is important. come you constrain developers and you run into the ever increasing risk of asking for things that are impossible. Stepping through workflows as part of a vocal what" referring to specific API call starts to get more into a "how".

Want requirements to be atomic requirement statement should cover one thing. Correct, which means possible related to extant interfaces if it is -- make sure the interface actually exists. unambiguously want them to be clear, concise, avoid vague language. These are not -- if you are not sure about the defined terms, define them, success harkens back to the user stories

presents a problem statements, should be task or goal oriented. What are you trying to do? This is an element of user stories. A pattern because they are communication tools. They create shared understanding among end-users, contracted parties, testers, project managers, that is me, and they give this diverse group of stakeholders hopefully clear set of parameters with which the project will go forth and when well written they alleviate further misunderstanding and arguments and mitigate unhappiness among one or all of those stakeholders.

This is where you all come in if you are listening from a nontechnical perspective. One of the biggest sources of bed requirements is not understanding the end users need, and this harkens back to what John talked about earlier this afternoon. Anyone who writes end-user requirements without talking to the real life end-users is doomed to failure making things very difficult. End-users on the comments rating team is better particularly if they have specific expertise in the area you're covering, and went to talk to circulation people for those requirements and catalogers for the catalog requirements.

They are gathered by talking to people. Focus on one specific set of related actions, example I will use in the next slide is Wikipedia style edit log for catalog records. For user stories you want to use these to frame your requirements bid these should be as atomic and specific as possible what is likely for a user story to contain several individual requirements. When several are gathered, you can figure out their position within existing workflows. Or for use in multistep workflows. This is a useful framing device we will talk about.

Manage expectations. So, a joke we have here at Equinox is the gospel according to Mike is through all things all things are possible. Corollary is don't use development to solve a training or policy problem. And the second is price, speed, quality pick any two. You want to manage your own and others' expectations. It should be noted that clear your requirements, the more likely your expectations are to be met.

Knowledge gaps. A little knowledge is dangerous thing. One of the biggest pitfalls and over specifying requirements is you probably do not know everything you think you know. That is why you should focus on what you went to the page when you drift off into "how" it becomes too easy to fall victim to incomplete knowledge. This is okay, it is not a failing of any of us, Evergreen is really complex software with a long development history. And hopefully you are dealing with experienced developers who know more about how to make your "what" happen.

Reality check. In constraint with the above two bullet points, constrain yourself to reality, I am sorry for being a bummer about that. Think about your needs, your budget, your timeline and how these interact with your requirements. Think of the larger project, does what you are doing interact with other parts of the software? There's more than one set of users and more than one kind of -- workflow. The gold everyone should be to produce working software. Software that is perfect always is not realistic. Big bummer for me also.

Requirements can be functional or nonfunctional. Functional requirements related to specific actions and workflows. Nonfunctional -- quarter requirements are answered to use cases and must include documentation or must integrate with Evergreen testing utilities, these are two common nonfunctional requirements I see a lot.

Finally, talk to other humans, talk to community members, to staff developers if you've got them. Or people familiar with the code. Talk to other people in the same staff roles. Get a range of people to provide input. If you design for accessibility talk to people who use accessibility tools. List serves and community working groups are great channels for soliciting input. And remember open source a porous vacuum and specific workflow may not be the same as others'. Requirements for-- should consider software as a whole as well as specific needs.

I do want to spend the whole slide on user stories because I think it is a helpful framing device. To get to that "what". I have laid out the general form of the user story on this slide, typically it is who wants to do what. And the why is helpful but unnecessary. This is a real world example, it is not a perfect example but it is a pretty good one and when I was able to dig up from launchpad.

Jeff Davis of ABC co-op broke down the larger Wikipedia style catalogs into specific requirements and reframe the first of his recrement is a user story. Atomically, this is four requirements webpage display the chronological display, user display, timestamp display but all are correct, testable, unambiguous and state what you want to do and why.

The user story is a way to focus on workflow outcomes and processes. Without letting yourself get bogged down in the details of how to develop that note the user story does not refer to specific permission calls, table names, really finicky display details specifics outside of, this is what we want to be displayed. So we can track who edits the record and I think that's a good

example from the accident--extant Evergreen launchpad outside of the user requirement that can be framed as a user story.

You've gathered your user stories and requirement needs in a concert translating those to actual requirement statements. Requirement statements should be stated in the positive, you want to say the software shall or the future shall versus shall not. Your language should be as clear and unambiguous as possible. If you can come have several different stakeholders read the requirements and see if all -- reached the same conclusion. If they all read what you wrote and you get a couple different variations of what you are trying to say, that is a good hint to edit and revise.

Define your terms. You do not want to make people assume what you are talking about, nor do you want to assume they know what you are talking about. Make sure you are clearly referring to a specific interface. This can be hard in Evergreen because there's not necessarily canonical names for interfaces. Sometimes developers know it interfaces by different names than an end-user would put on them.

So the web client, the web client site, it, it usually has a unique URL. So it is absolutely clear what you are talking about. Remember to keep focusing on the abstract, the "what" rather than implementation.

Want to make sure the true user stories and requirements .2 things which are testable. Walking that tricky line tween -- excuse me, walking the line between abstract enough for a requirement but specific enough to test is a tricky line to walk but vague statements do not help anyone. You want to check in frequently against your scope definition you said before and make for you do not wander off course for what you want to do.

Want to make sure you consider potential errors in your workflow. Either user errors or software errors but what do you want the system to do if it encounters an error? While testing can never account for all the real-world ways that humans can break software or it can break itself, thinking about how you want to handle errors false is important and can be instructive in clarifying user stories and workflows.

There is no perfect user. Trust me. You can consider using identifiers in your requirements. So stay cold as can refer to requirement 3.B.1 or something like that. And remember if you are struggling to describe something in words, a wireframe can be helpful. Sketch out large interface elements to clarify your thinking or to illustrate your thinking to others but just as a note of caution, don't become overly tied to wireframes, there meant to eliminate the requirement but the requirements themselves should contain the definitive version of what you want.

I said a lot of words in the last slide, this is the short picture version of all of those words I said in the last slide. All of these individual steps, remember you should be communicating with your stakeholders and working through the iterative feedback loop of discuss, revise, refine.

This is also a summary of what I talked about in the preceding few slides. I will not read every bullet point, but this gives the word version of what makes requirements good and bad in my humble opinion. On the previous slide, I showed you the good and better now I will show you the ugly.

Why are these ugly? Some of them look sort of good. But they lack context or other details. Some are vague and untestable. What does easy to use mean? What does robust mean? Need to think about these things in a more concrete, specific way and spell them out a little more. And some are impossible please do not write impossible requirements.

In conclusion, but requirements are clear goals that make for happy developers, users and happy everyone, it is the key to happiness, you guys. Since this is library software at the least we do not have to worry that poorly communicated requirements will result in the loss of a very expensive orbiter. When-when, guys. I managed to go through the slides in record time. I apologize if I was speaking too quickly.

I am happy to take questions if anyone has questions. Please use the chat box. Thank you, Debbie for echoing, don't use development to fix a training problem, that is practically my motto. If you worked with me in software development I have literally said those words to you at some point.

If don't have questions we can let everyone sign off and go 30 minutes early. Thank you Debbie for hosting and Karen for captioning.

>> DEBBIE LUCHENBILL: Thank you.

>> ANDREA BUNTZ NEIMAN: We have a question about Africa want to be clear I'm speaking, I am not speaking officially about the Evergreen community, about what makes for good recrement, this is just my experience as a project manager at Equinox. Requirements of a new feature -- no, we do not have an Evergreen standard place to post these kinds of things, there are some words about this on wiki, I think Kathy Lucere wrote them but not on the actual website, course the presentation will be posted there with the other presentations.

What can we do to encourage specific and clear requirements on a launchpad? Okay, that is a huge question. Because launchpad is a way where new community members can get involved with filing bug reports and I and others have talked over the years of what makes for a good bug report and there's actually a good deal of similarity between what makes for a good bug report and what eventually becomes requirements, which is that you are clear, are workflow focused, you are concerned with reporting the facts on the ground. And also what you would like it to do to be fixed.

Encouraging specifically requirements on launchpad --That is really just getting people to frame like I keep-- hammering this but it's an agile development concept that we do not really do the full agile element framework but this concept, the user stories concept comes from agile and I think it is a really useful framework for end stories similar to what makes for a good remark for bug reports, where we ask people what were you doing, what happened and what did you expect to happen? That is the basic framing for bug reports and then for acquirements, the user story is, I want to do X because this particular context of user, like a cataloger I want to be able to do a thing because of this reason. So having that framing device and maybe asking clarifying questions to tease that out if people are ambiguity. It is easy -- I include myself, this is much an instruction for me as everyone else, do not get hand wavey think clearly about what you want and the frame that as an atomic specific element, that is a great question, Mike. Thank you.

This presentation makes a great proofreading document for potential proposals. Feel free to use it as such. We are an open source community, steal and share alike and all of that. Like I

said, the slides we posted, something I think about a lot, something I interact with a lot in my day to day work. Of course, since I came from the end-user side of things, is a decade long user of Evergreen I constantly think about that from the end-user perspective.

Oh, I'm sorry Blake, I misunderstood you twice, like mentioned a place to put proposal for a new feature. Wiki used to be a place for that and there's still a defunct page for developing ideas but I think the standard place for that has kind of now become the listserv, if you're looking for more, the open I list dev list, and today we will be sending a message out with some public testing information for the curb side feature because were interested in technical developer feedback or if you are looking for general community input I would say the general listserv would be a place for that. Is there a page on the website for that? I'm going to look now. There is a page on the wiki... It is linked from the website so -- I just sent that only to Debbie, let me change that and send that to all.

There you go, that is linked from the webpage under "get involved", there is a link-- for proposed development project and takes you out to the development page it looks like this was last updated almost exactly one year ago.

I believe that project listed on their as course materials module -- has actually not only solicited quotes but moved forward with a quote, or moved forward with work on that so that is outdated. But you know, we can -- you can single-handedly bring it back, Blake. In all seriousness, that is the place to do that. But since it's probably not super actively monitored, if you're going to put things on that wiki page, I would go ahead and shout it out to the lists or even IRC, just to draw attention and be like hey, guys I posted this, who wants to look at it kind of thing.

Oh, yes, I, Debbie said Jane was presenting on the course materials, was that the student success working group?

>> DEBBIE LUCHENBILL: Yes.

>> ANDREA BUNTZ NEIMAN: Yeah, I popped into that briefly but I didn't stick around so I didn't see that. That has moved well past that seeking quote phase.

>> DEBBIE LUCHENBILL: We have a question in the Q&A -- development for ability to search multiple languages for bilingual items was done that requires knowledge of evergreen term such as item, lingered, it might be better for you just to look at this -- [Laughter].

>> ANDREA BUNTZ NEIMAN: Well, there is a staff interface way to do that, excuse me, and OPAC interface way to do that. You can actually, from advanced search, click or control click to select multiple languages like from the interface. The way it translates that into the search box is that item line, SPA ENG, but for the staff, for the users you can -- actually do that just from advanced search without having to type that specific thing in the box. So you just show them the advanced search and they can -- by default, I don't know if your library has this turned on or not but by default there's an item language selector and that is what you can use to do that. Con is a couple things you have to make sure your administrator does on the backend like make sure the correct fields from 041 -- my -- cataloger days or so long ago did whatever that mark feel -- an item you want to make sure the subfields are indexed correctly. So you can search them but there's a way to do them from the advanced search OPAC interface without having to type item_lang in there. All the catalogers are chiming in, thank you for that.

But yes, so, I hope that answers your question. That you can do that through the OPAC interface under advanced search. I saw that Rogan -- yes, Rogan has a good point about the wiki, it is not used a lot because not everyone has editing access on the wiki. This is true but it is easy to request edit access on the wiki. So I think, I know that Galen has power to elevate users. I know there are a couple others in the community who can give edit permissions in the wiki. And I think there is a way to request that, to just request edit permissions so if you're a wiki member and you want to do a wiki page, we request edit permissions and go forth. Are there other questions or follow-ups. Yes we can do that but it is not the same -- hmm, I mean -- I can show you, Debbie if you could -- if I can get -- know I have screen sure, I still have screen share powers. I'm not sure why -- I'm not sure why I was having problems earlier. It was probably all my fault.

I will just go to curb side -- that is the most recent test system I have. Here is the language that I was talking about, so you can select -- I am doing click to multi-select, control click to multi-select rather and -- So this puts the language down here, if you are still seeing it up there, in the search box, I think that was a later bug fix, now that I am talking through this, walking through this interface, so what I am looking at in this curbside demo is a recent version of master but I think- it used to be that the search terms -- that will still show up -- anyway, you can, you used to be able to type that syntax in as well and it might be that-- that is an older version because

I'm thinking no -- oh, doing or/not/and -- you can make it do or/or/and -- yes. That is just if you do-- two pipes is "or".

So double “&” is and two "|" is "or".

I would really have to go back and look at the documentation for that. It is been a couple years since I worked on that directly.

You are still seeing my screen but let's see if I can drop this link into the chat as well. That must've been about that was fixed. The link I dropped in the chat was the original documentation when this went in in 3/1 and that talks about what it indexes and then down here is where it tells you to do -- so here you can do Boolean or -- etc. -- Boolean and -- and then on advanced search, yes, it applies the or operator on advanced search. So, there is no clicky way to do an and search on interface but if you type it into the search box using the double & and that syntax, I put a space, that is why it didn't work.

Let's see. So that is bringing up -- arias with the language Italian food so if I wanted to do AND what let's say wanted to own OR-- wanted to do and OR -- to be of German arias to we have German arias? That brought up things with a keyword, arias, that have either Italian or German so -- that gives you an example. I know that is not super N user-friendly to type that in but there's a predictable short context for that, so I hope that helps. The person that I think the problems in the development requirements not being exact that could have been the case. Equinox did the work for that but I may have been that specific have met that requirement was not specifically stated to have a way to do that via clicking in the advanced search menu. Again, I'm judging up 2.5 years ago memories.

That is a very good point and it brings it all back around which is if you want that interface to be -- if you want a specific interface element, state it. Like -- and it may have been that we talked about adding too much complexity because there's also, it comes back to constrained existing interfaces -- there's a lot going on in the advanced search interface, you have to have a conversation about where you're going to stick that and that this is going to become a submenu where you want to say Italian or Spanish but not German, like how deep do you want to get in the complexity of what you are doing to the interface and if there is a way for a power user will want to search like that to do it in an existing manner, i.e., by typing it you might end up in the

diminishing returns circumstance. That is again why you want specific requirements and talk about them.

And talk to your power users, do they expect to see that there and are they comfortable typing it in a box? That is a good point. Anything else? You got a bonus impromptu demo, so thank you for asking the questions.

>> DEBBIE LUCHENBILL:. Thank you all for coming unhelpfully we will see you again tomorrow and Thursday. Again, thank you to Equinox for sponsoring captioning and for Karen our captioner and for Mobius sponsoring the track. Good bye.

>> ANDREA BUNTZ NEIMAN: Goodbye everyone. Thank you, Debbie.