Processes for User-Centered Design and Development: The Omeka Curator Dashboard Project

Susan Chesley Perry
University of California, Santa Cruz, USA

Jessica Waggoner
University of California, Santa Cruz, USA

ABSTRACT

The authors discuss user-centered design and agile project management using the development of the Omeka Curator Dashboard as a case study. The University of California, Santa Cruz University Library developed a suite of 15 plugins for the Omeka open source content management system. This chapter describes the library's use of agile principles and methods for the management of this project, detailing the creation of user stories and acceptance criteria. This chapter also outlines the usability testing conducted by the library in the form of online surveys and moderated field tests. The authors conclude that user-focused, inclusive, and iterative development are key components to the success of the software development process.

Keywords: Agile Project Management, User-Centered Design, Usability, Omeka, GitHub, User Stories, Acceptance Criteria, Omeka Curator Dashboard

INTRODUCTION

This chapter discusses user-centered software development projects and their management using the development of the Omeka Curator Dashboard by the University Library at the University of California, Santa Cruz as a case study. This chapter introduces the reader to Omeka, an open source content management system designed by the Roy Rosenzweig Center for History and New Media at George Mason University for digital asset management and digital exhibit creation. Omeka is used by a variety of libraries, museums, and academic researchers.

The University of California, Santa Cruz University Library developed a suite of 15 plugins to expand the functionality of Omeka in the areas of collection building, description, management, and preservation. The library managed the development project using an agile project management approach. This chapter describes the benefits of using agile for this type of development work and details the library's utilization of user stories to support a user-centered approach to plugin design. The library also conducted usability testing to reveal problematic aspects of plugins and discover potential future enhancements. This chapter describes the library's use of both online surveys and moderated field testing.

BACKGROUND

Libraries and archives are increasingly focusing their staff expertise on curating and disseminating the digitized primary source materials in their archival collections. As part of this effort, archivists and curators see an increased need to engage and connect researchers directly to those materials (Theimer, 2010). Crowdsourcing transcriptions and other descriptive information is one way to engage users, but as Trevor Owens (2012) of the Institute of Museum and Library Services (IMLS) stated in his blog post, crowdsourcing transcription or tagging should not be considered the ultimate end goal. Instead the goal of participatory archives projects should be to improve user engagement and understanding of the institution and its collections. The *2015 NMC Horizon Report: Museum Edition* states that two short-term trends for museums and archives include the expansion of the concept of patrons to include both in-person and virtual visitors and the increasing focus on participatory experiences on-site and online (p. 16). The report further explains one of the significant challenges facing museums is the need to improve digital literacy of museum professionals (p. 24). The authors of this chapter assert that libraries and archives face that same challenge. The director of the Scholars' Lab at the University of Virginia Library, Bethany Nowviskie calls on libraries with research and development teams to engage in "creative, iterative, unfettered, informal, (even gonzo?) development of digital scholarly interfaces and content" (p. 55). Digital humanist Cris Alen Sula describes a model where libraries can truly engage digital humanities scholars and support their projects by promoting skills and leveraging the user-centered service paradigm accepted by most libraries (p. 24). Faced with the challenge of developing systems to support digital humanists and to engage users with digital content while keeping that development creative and iterative, the UC Santa Cruz University Library chose the agile development approach to managing software projects. Agile methodologies assert that the user is always the final authority on product quality, and quick iterations of developed functions allow for frequent and prompt feedback from those users (Beyer, Holtzblatt, & Baker, 2004, p. 56). Following is an explanation of the functions the UC Santa Cruz team developed in order to facilitate engaged user participation with the digital archive and a step-by-step approach to developing those functions for the Omeka open source content management system.

CASE STUDY: THE OMEKA CURATOR DASHBOARD DEVELOPMENT PROJECT

What Is Omeka and What Can It Do for Me?

Omeka is a free, flexible, open source Web publishing tool used largely by libraries, museums, and scholars. Omeka has a unique hybrid disposition as both a fully featured digital exhibit platform and a capable digital asset management system. In this chapter, the authors will outline the development process for creating plugins, or additional modules that extend the core functionality of the software. The downloadable Omeka software requires a LAMP-configured server (including the Linux operating system, Apache, MySQL, and PHP), along with ImageMagick for image manipulation. Omeka is a project of the Roy Rosenzweig Center for History and New Media at George Mason University, and the creators provide several options for institutions without system administration resources. They list suggestions and options for running Omeka as a virtual machine on Mac and Windows servers or in the cloud on a variety of fee-based hosting platforms (“Preparing to Install,” 2016). A hosted version of the full downloadable version of Omeka is best for institutions with limited server and system administration resources. Omeka.net is a third option with a fully hosted and supported instance. Omeka.net has a free basic plan that includes 500 MB of storage, one site, 15 plugins, and five theme choices (Corporation for Digital Scholarship, 2015). Other fee-based tiers allow additional sites, files, and features. Omeka.net is the best for institutions looking to test-drive the software before fully committing the resources, or those with very limited IT resources.

The Omeka system itself is a structured database of files with metadata assigned to those objects in element sets. It handles most file types, including media. Each item can have multiple files attached to it, such as alternative views for images. Items are described using the Dublin Core metadata standard with additional metadata element sets for several different item types. These additional element sets can be modified by the institution. Items can be organized into collections and exhibits. Plugins are optional features that extend the functionality of Omeka, and they are developed by the Roy Rosenzweig Center for History and New Media as well as the open source community of developers using Omeka. Examples include the Geolocation plugin, which allows users to pin items to a Google Map and capture the geographic coordinates for that item in the metadata, or the YouTube plugin, which embeds video from the YouTube site and imports metadata such as creator and titles into the Omeka collection. Omeka themes control the look and feel of the site, including the colors and fonts. The Exhibit Builder plugin is a major feature of the system, which allows users to create Web pages that combine items from Omeka collections, including narrative text providing connections and context for items. Creators can incorporate visualizations such as maps and timelines to further illustrate their points.

The UC Santa Cruz University Library began using Omeka as a digital asset management system for the Grateful Dead Archive Online (GDAO). This large collection includes over 35,000 items, including images, documents, audio, and video. The site makes extensive use of the Solr Search plugin to index and create facets for the many categories of content. The online archive also accepts online contributions of materials from fans and scholars using the Contribution plugin. Library administrators chose the Omeka platform based on evaluations from other institutions such as the Metropolitan New York Library Council's experience with Omeka for their digitalMETRO project (Kucsma, Reiss, & Sidman, 2010). The Grateful Dead Archive Online is a blended collection of curated materials donated by the band along with artwork, images, and text uploaded by the community. In this sense it works as an exhibit platform, a system for collecting user contributions, and a digital asset management system for both types of content. The Omeka core code's flexible approach to metadata representation was also particularly appealing. The Grateful Dead Archive is a significant 20th-century popular culture collection, and the project team needed the flexibility to create and incorporate local vocabularies specific to the collection. The Grateful Dead community members are truly the content experts, and the archive wanted to leverage their knowledge of names, places, and events. Omeka allowed the team to maintain a standards-based system employing Dublin Core metadata and controlled vocabularies along with crowdsourced tags and other content.

Archivists, librarians, and instructors at UC Santa Cruz also use Omeka to highlight materials from a range of collections held by the library's Special Collections and Archives departments in digital exhibits. The purposes of these sites have varied from digital components of physical exhibits, to classroom projects curated by students, to online-only sites with interactive elements and community contributions. Much of the Omeka development done by the UC Santa Cruz Library team supports these functions and will make up the case study in this chapter.

Several content management systems are available to libraries and museums, and the team of librarians and developers has experimented with several different packages and sites in addition to Omeka, including WordPress and Scalar. The basic functionality of these systems is similar, and they all meet the basic needs to display digital items along with contextual content. Omeka's strengths are the contribution functionality allowing users to upload files to the collections, its structured exhibit templates, and its wide range of plugins to manage content in different formats. Currently the community has created over 60 plugins to enhance functionality. At the UC Santa Cruz University Library, librarians, archivists, and students all use Omeka and have found it easy to use. Evaluations from other institutions praise its "low barrier to entry" (Marsh, 2013, p. 2). WordPress has a very large community of developers and is a common blogging platform, which makes it familiar to many users. However, it lacks the predefined Dublin Core metadata schema for items in the collections and the templated exhibits. Scalar is a similar digital exhibit platform and performs many of the same functions. The strength of Scalar is its ability to draw connections between objects and embed objects from external sites. Librarians and archivists often start with a collection and provide context, background, and visualizations based on those items. Researchers may start with a text or a thesis and then add objects and visualizations to illustrate that point. Scalar natively supports that workflow. In his evaluation of Scalar at the University of Illinois at Urbana-Champaign, Daniel Tracy (2016) highlights Scalar's ability to create multiple paths through a set of research objects (p. 164). As the Digital Scholarship and Digital Initiatives departments at the UC Santa Cruz library have evolved, they've begun to focus on the needs and research questions of faculty and curators in order to make a recommendation for the best digital exhibit platform. The user-focused agile development models used by the project team can be implemented for design and development of functionality in any platform.

Developing for Omeka

Although Omeka can be used as a turnkey solution for Web publishing and still allow the user a great deal of flexibility in the appearance and structure of the site through the use of themes and plugins, one of the benefits of Omeka is its open source nature and robust development community. After the library's work on the Grateful Dead Archive Online, archivists and librarians began conceptualizing use of Omeka as a general exhibit platform for all of the digital collections. However, Omeka lacked workflow and curatorial management tools that would have optimized this more expansive use of the exhibit platform. As use of GDAO progressed, the exhibit creators also began to recognize a need for additional tools that would enhance their ability to build, maintain, and preserve the collection.

The library administrators decided to take advantage of Omeka's open source code and began the initial planning of an Omeka development project in which a team of programmers, designers, archivists, and librarians would design, develop, and evaluate these tools in-house. At this point in time, the library did not have an applications developer or a dedicated project manager on staff; the dedicated developer and project manager positions for the Grateful Dead Archive Online project were temporary and had been funded through an IMLS grant. Therefore, pursuing a major development project would require the library to seek additional funding. However, library managers were confident that the Omeka tools they envisioned would be desirable and useful to the entire Omeka community, making this development work suitable for grant funding. In 2012 the UC Santa Cruz University Library was awarded a second IMLS national leadership grant to assist the team in creating new Omeka tools, which were dubbed the Omeka Curator Dashboard (OCD). The OCD would be a suite of 15 Omeka plugins delivering functionalities in the areas of collection building, description, management, and preservation.

The development team relied on a large number of library staff serving in diverse roles across the organization in order to support the development project. The IMLS grant provided funding that allowed the team to hire a project manager, who was funded to dedicate 50% time to the management of the OCD project, and a full-time applications developer. However, a development project of this scale would not have been possible with a project team composed of only these two staff members. The library recruited staff from the Digital Initiatives department to engage in usability testing and serve as subject matter experts for digital library workflows, from the Special Collections and Archives departments to advise on the needs of exhibit curators, from the Metadata Services department to instruct on the needs and workflows surrounding digital object metadata, and from the Information Technology Services department to design and build an Omeka prototype Web site for digital exhibits and for the system administration infrastructure necessary to support the prototype. As work progressed and the scope of the project expanded to include instructional use of Omeka exhibit building, the team also tapped the library's CLIR (Council on Library and Information Resources) Digital Humanities Post-Doctoral Fellow to shepherd the inclusion of pedagogical Omeka exhibit sites. The core team consisted of the project manager, developer, and the Digital Initiatives department manager who communicated about project and budget status regularly with the library administration. Other staff were pulled in as their expertise was required for determining their needs and testing the software.

Assembling a project team composed of staff from departments across the library can be a formidable task. Obviously, the departments from which staff members are pulled incur the opportunity costs, prolonging the timelines on some of their own departmental projects, or possibly forgoing them altogether. This makes advance buy-in a crucial element to this type of library development project. Library administrators and the project manager were able to cultivate buy-in among department heads and staff by framing the project as an opportunity for their staff to develop or enhance skill sets. Project team members would also have the opportunity to shape the tools that they would be utilizing in the future. The library dedicated funding for conference attendance for staff involved in the development project as a portion of the institutional cost-share for the grant. This funding served two roles. First, it allowed staff to be project ambassadors at national conferences. They met with other Omeka users, who shared details about their institution's Omeka implementation, and they informed these users about the OCD development project, possibly even soliciting the Omeka users as future beta testers of the OCD tools. Second, this conference funding allowed staff to attend sessions that developed or enhanced their skill sets, facilitating their work on the OCD development project and other future library projects.

The library used the experience gained during the Grateful Dead Archive Online project to estimate the development staffing needs for the OCD project. The library administration had hoped to hire the same programmer they had hired to develop for GDAO, but that person had moved on to a permanent position at another institution just as the grant funding ended for that project. Losing skilled staff is a definite risk when one-time funds are used for development projects. Hiring another experienced programmer into a temporary position for the OCD made for a challenging and long recruitment, which delayed the project by several months. Though it was a difficult lesson learned, it was helpful in making the case to library administration for a permanent programmer position. By the conclusion of the second

IMLS grant, the project team was able to justify transferring the developer position from temporary to permanent, not only to pursue new development efforts, but also to sustain the existing projects.

Project Timeline

Figure 1. Timeline for the OCD project

Assessing Development Needs

The UC Santa Cruz University Library took several steps to assess the library's development needs for the next phase of digital collection management and exhibits in Omeka. The first step included a retrospective evaluation of the construction and deployment of the Grateful Dead Archive Online. The postmortem of the project launch revealed a significant challenge to importing large batches of materials into Omeka. The project team also conducted interviews with GDAO stakeholders and administrative users. Now that the archivists and metadata specialists were using the system on a daily basis, they had specific recommendations on how to improve the site. The interviews revealed a host of unmet needs and ideas for increasing efficiency in object management and curation. For example, the metadata specialists discovered that they needed a way to edit metadata in bulk across the Grateful Dead Archive collection. This bulk editing included the need to delete, find and replace, append, and add new metadata to existing fields. Archivists also desired a history log for each object that would expose any curatorial updates made and the username of the staff member who made the change. Collection managers needed a mechanism for effectively pulling digital objects, both files and metadata, from our larger digital collection housed in CONTENTdm. Asking our exhibit curators to manually enter CONTENTdm metadata into Omeka and then find and upload the corresponding file was inefficient and tedious. The team wanted to provide curators with tools that would allow them to incorporate digital objects from collections outside the library systems, specifically image and video hosting sites such as YouTube, Flickr, and Vimeo. Facilitating the addition of these externally sourced materials would allow curators to add a great deal of richness to their digital exhibits. Another challenge the collections managers faced was the archival preservation of the digital files and metadata in the Omeka collections. Although the library used

CONTENTdm to manage materials digitized in-house, Omeka was managing an increasing number of unique user-contributed files. Archivists needed a tool that would allow them to transform these Omeka items into METS (Metadata Encoding and Transmission Standard) and push them into an archival repository, such as the California Digital Library's Merritt digital preservation repository.

Once the project team had a draft list of functional needs, they consulted the main developer of Omeka, The Roy Rosenzweig Center for History and New Media at George Mason University, to compare these needs with the core manager's development roadmap. The Center for History and New Media staff were also very knowledgeable about projects in development at other institutions across the Omeka community. The UC Santa Cruz team wanted to be sure their plans complemented rather than duplicated the plans for core development. Because they were supported by federal grant funding, they also wanted to be sure their development would meet the needs of the larger Omeka and digital library communities.

The project team also consulted with the University of California Curation Center (UC3) at the California Digital Library to determine the feasibility of depositing objects from Omeka collections into their hosted repository as well as other standards-based repositories.

The team engaged in conversations with experienced Omeka developers around the country and their own administrative and IT departments to determine the resource requirements for such a development project. The assessment culminated with an environmental scan of other Omeka projects to determine how the development landscape had shifted between the deployment of GDAO and the awarding of the second IMLS grant. Through this assessment work, they identified broad requirements for project deliverables.

Managing Development

The UC Santa Cruz library had previously used a traditional "waterfall" project management approach for development projects. Although this method provided project teams with high levels of predictability and a more formalized structure, they found that it concentrated the development phase of the project timeline in a manner that did not accommodate any late changes in functional requirements. When employing a user-centered approach to software development, it is essential to adapt functional requirements to changing user needs. This is particularly true when working with open source software. As new versions of Omeka were released and other developers in the Omeka community published new plugins, the team found that some of the functions in the original grant proposal were already in development at other institutions. The team found it necessary to consistently track the work of the Omeka development community and collaborate on similar plugins in development by other groups. Using a traditional project management method would have hindered the team's ability to quickly adapt to these changes in the development environment. As H. Frank Cervone (2011) opines when describing the disadvantages of traditional project management methods, ". . . requirements definitions are often so labor intensive and protracted that the requirements for the project have changed before development even begins" (p. 18). For these reasons, the team chose to use an agile project management framework for this development project.

Agile project management grew out of the Agile Software Development movement spearheaded by Beck et al. Agile software development articulated four key values for software development work: 1) individuals and interactions over processes and tools; 2) working software over comprehensive documentation; 3) customer collaboration over contract negotiation; and 4) responding to change over following a plan (Beck et al., 2001a). In addition to the four key values, Beck et al. laid out 12 principles to support the values declared in their Agile Manifesto. Agile project management methods reflect the four values and supporting principles. Although there are a variety of more prescribed agile methods (Scrum, extreme programming/XP, etc.), the UC Santa Cruz library chose a less formal approach for their small team. They embraced the two core components of the agile approach, namely its iterative development cycle and regular direct communication and collaboration between stakeholders and developers. Cervone (2011) describes the importance of these elements, "The reasons these two concepts are emphasized is simple: both help a project team adapt quickly to the unpredictable and rapidly changing requirements most development projects are carried out in" (p. 19). The team also aligned their project with the principles behind the Agile Manifesto, particularly 1) early and continuous delivery of valuable software; 2) welcoming changing requirements, even late in development; 3) delivering working software frequently; and 4) working software as the primary measure of progress (Beck et al., 2001b).

The timeline for the development component of the Omeka Curator Dashboard project was one year. For the suite of 15 plugins, that allotted two to four weeks per plugin iteration to achieve the one-year timeline. Jim Highsmith, one of the creators of the Agile Manifesto, refers to these feature development timeframes as "timeboxes" and suggests that two to four weeks is an ideal length (2009, Kindle Location 2731). To decide whether a feature would require a two-, three-, or four-week timebox, the project manager and the developer reviewed the functional requirements to determine an estimated level of effort (LOE) for each complete plugin. Each plugin was assigned a low, medium, or high LOE corresponding to a two-, three-, or four-week timebox. The work performed inside the timebox was building/development time. Design and testing phases were also iterative in nature, but were not bound within the development timebox.

The project manager served as both the main stakeholder and the project facilitator. The project manager was responsible for working with subject matter experts (analogous to customers in an enterprise setting) to assess user needs for functional requirements expressed as user stories, communicating with the developer about features and user stories, developing acceptance criteria, conducting acceptance testing, and liaising with in-house users for feedback after the deployment of beta iterations. The developer was responsible for communicating with the project manager for information and clarification on requirements and acceptance criteria, giving feedback on LOE estimates for user stories and features, writing technical specifications based on functional requirements, developing the plugin iterations, fixing bugs and other issues found during bench testing, and providing LOE estimates for subsequent iterations.

The development cycle for each plugin had four distinct phases: 1) design, 2) develop/build, 3) implement/bench test, and 4) deploy/formal evaluation. During the design phase, the project manager created functional requirements, expressed as user stories, based on the needs of subject matter experts. Once these user stories and accompanying acceptance criteria were complete, the project manager and designer met to assign a level of effort to each plugin. As described earlier, this LOE estimate determined the size of the development timebox. The developer would then kick off the development phase by writing technical specifications. As agile emphasizes “working software over comprehensive documentation” (Beck et al., 2001), these technical specifications were often minimal. Rather than create extensive technical documentation, the developer ensured that the code itself was well commented. While the developer was building the plugin, the project manager created acceptance testing scripts based on the previously documented acceptance criteria. This simultaneous work allowed for a faster development cycle. Once the developer had a working iteration of the complete plugin, or for larger plugins a single function, the project manager bench-tested the iteration using the acceptance testing scripts. The developer would then fix any bugs or significant issues found during bench testing. The deployment phase involved use of the plugin within a production Omeka environment used by curators within the library. Curators gave feedback about the plugin to the project manager, who either shelved the feedback as a future enhancement or forwarded the issue onto the developer to address in a new iteration. This final phase also included a more robust evaluation of the plugin’s usability. The phases of the project are indicated in Table 1.

Table 1. Tasks by project phase

Phase	**Project Manager Tasks**	**Developer Tasks**

Design	● Write features and user stories ● Write acceptance criteria ● Determine LOE ● Set timebox size	● Review features, user stories, and acceptance criteria ● Determine LOE
Develop	● Write acceptance testing scripts	● Write technical specifications ● Write code iterations
Implement	● Bench-test iterations ● Report problems	● Revise code to fix problems found during testing
Evaluate	● Communicate with end users ● Document potential enhancements ● Report problems requiring immediate fixes ● Conduct formal usability testing	● Revise code to fix problems found by end users

The team used the "freemium" (meaning free as well as paid tiers are available) project management web application Asana to assign and manage development tasks. Because the phases were identical for each plugin, the project manager created a template in Asana that was used for each plugin. The tasks for each plugin were assigned to a member of the project team (generally the project manager or developer) and given due dates. Asana was chosen over other more robust project management software options for several reasons. First, its deployment as a Web application meant that it could be accessed on any computer without the need to download software. Second, it was deployable in minutes, requiring very minimal setup by the project manager. Third, it had a clean intuitive interface requiring no training prior to use by the project manager or other members of the project team. Asana is a lightweight task-based solution and does not allow for the creation of Gantt charts. Therefore, Google Sheets was used for overall project scheduling, whereas Asana was used for task tracking and more detailed development due dates.

Designing the OCD Plugins

The development team began the design process armed with the broad deliverables outlined in the initial assessment. To create functional requirements for the Omeka Curator Dashboard plugins, the project team members interviewed potential users, examined their current and desired workflows, and discussed their specific objectives for using the broadly described OCD plugins. The project manager identified subject matter experts in departments throughout the library, ranging from exhibit builders in Special Collections to metadata experts in Metadata Services.

For example, one OCD project deliverable was a plugin that would import content from a video hosting site such as YouTube. The project manager met with exhibit builders from the Special Collections and Archives departments to examine the curatorial needs of such a plugin and the workflows surrounding the incorporation of externally hosted videos into a digital exhibit. The curators explained that although Omeka natively allowed them to embed videos within an exhibit, they sometimes preferred to have a video be its own distinct Omeka item. As an item, curators would have the flexibility to add the video to a collection, connect the video to other items throughout the Omeka site through indexed fields such as subject headings or tags, and expose the video's metadata. They wanted a plugin that would quickly and easily import a video as an Omeka item and include the video's metadata from its source. The import of metadata from the video hosting sites YouTube and Vimeo required a crosswalk between the external site's metadata schema and Omeka's basic Dublin Core fields. The project manager consulted with experts in the library's Metadata Services department to design this crosswalk.

Based on the information gathered in interviews with subject matter experts, the project team developed user stories detailing the desired capabilities of each plugin. Mike Cohn (2004), co-founder of the Agile Alliance, defines user stories as "short descriptions of functionality—told from the perspective of a user—that are valuable to either a user of the software or the customer of the software" (para. 1). Rather than describe user interface elements or detail system behavior, user stories briefly explain what the user should be able to do when interacting with the software. User stories should also be very specific and descriptive; ideally, two people independently reading a user story would share an understanding of what the user should be able to do even if they have different thoughts on how that functionality might be achieved. Cohn (2004) points to imprecision in language and the risk of differing interpretations between customers and developers as a key advantage in utilizing user stories (para. 5).

Due to the detailed nature of our user stories, the project team found it necessary to wrap them in a broader hierarchical context. In his book on agile project management, Jim Highsmith (2009) refers to these hierarchical elements as “stories” and “features,” wherein features may be complete functions, whereas stories may be smaller components of a given feature, delivering useful but not complete functionality (Kindle Location 2817–2819). An example of this feature and story division can be seen in the Item Review plugin depicted in Table 2. The feature for this plugin stated, “As a curator/admin, I want to be able to view items created by Contributors and Researchers and approve them for publication.” Alone, this feature would not be adequately specific for initiating development work so the project manager divided it into three distinct user stories: “As a Super User, I can determine which roles require approval for publication,” “As a curator/admin, I can see a list of all objects pending approval,” and “As a curator/admin, I can approve items for publication.” Expressing the functional requirements as user stories allowed the team to pursue a more user-centered approach to the design by employing the user’s perspective when envisioning and describing desired functionality. The user story approach also facilitated an iterative development process by concentrating development efforts on one user story at a time, then testing the completed user story while simultaneously beginning development work on a next user story.

Table 2. User story for the Item Review plugin

ID	**Title**	**Feature**	**Priority**	**User Story ID**	**User Story**	**LOE Est.**

17	Item Review	As a curator/admin, I want to be able to view items created by Contributors and Researchers and approve them for publication	2	17.1	As a Super User, I can determine which roles require approval for publication	Med
				17.2	As a curator/admin, I can see a list of all objects pending approval	
				17.3	As a curator/admin, I can approve items for publication	

The project manager derived acceptance criteria for each plugin based on the articulated user stories. Acceptance criteria are the minimum requirements for a given feature; once the developer feels that these criteria are met, the feature exits the initial development phase and enters the testing and refinement phase. Agile coach Sandy Mamoli argues that good acceptance criteria should not be overly detailed, should "state an intent not a solution," and should not be implementation specific (Mamoli, n.d., para. 3). Based on these guidelines, the acceptance criteria did not contain any technical specifications constraining the developer in his implementation of the user story, nor did they include any detailed instructions on the user interface design. Refinements to these aspects of the user experience would be made after acceptance testing, in line with the iterative approach to development.

To return to the example of the Item Review plugin, one acceptance criterion was "Users in designated Reviewer roles can approve items that are pending." This criterion did not specify details such as the wording, color, or placement of the approval button. Avoiding this level of detail facilitated the iterative development process by allowing the developer to focus on delivering functionality. The project manager then identified refinements to these elements during acceptance testing.

Acceptance testing ensured that all acceptance criteria were met. The team developed detailed testing scripts enumerating the steps taken to test each criterion. Returning to the example for the Item Review plugin, the test for determining whether reviewers could approve pending items (approval of items being just one of the articulated acceptance criteria) was as follows:

1. Log in as a Reviewer-level account.
2. Find items for review.
3. Approve item(s).

The team referred to this acceptance testing as the "bench testing" phase for each plugin. During this phase, the project manager produced a bench testing report, reporting bugs, identifying necessary changes to user experience elements, and recommending possible future enhancements to the plugin. The developer then fixed any bugs and incorporated changes identified by the project manager during bench testing. Enhancements were either made at this point or shelved for future versions of the plugin, depending on available slack in the project timeline and the level of effort required for the enhancement.

After this second round of development work, the plugins were considered to be in "beta." Beta versions of the plugin were deployed in the library's own Omeka sites for use by exhibit curators. Curators subsequently suggested additional improvements to experience elements, such as buttons and wording. The project manager gathered these suggestions for potential incorporation into future versions of the plugin. May Chang (2010) describes this agile process for incorporating user feedback as follows: "Requirements elicitation is an ongoing activity, because needs are constantly changing, and user feedback is collected at the end of the development cycle. The development team is often willing to go back to the design phase to redesign the system around user feedback" (p. 673). This iterative cycle of development and refinement allowed the team to quickly deploy beta plugins for use in production Omeka sites and incorporate improvements into subsequent iterations on user feedback.

Because the library was already using GitHub as a repository for the OCD plugin code, the development team found it most efficient to use GitHub Issues for reporting bench testing outcomes. The project manager opened separate issues for bugs, user interface changes, and potential enhancements.

Communication between the project manager and the developer about each issue occurred outside of the GitHub platform in greater detail. However, the team found GitHub Issues to be the most efficient way of tracking and monitoring the status of changes made after bench testing. Because the code was openly available in GitHub, other Omeka users at other institutions could also download and use the beta plugins. These users also opened GitHub Issues if they found bugs. Placing the official bench testing findings in GitHub gave the developer a single platform for storing and tracking all refinements.

Evaluation and Usability Testing

The Omeka Curator Dashboard project included more formal usability testing conducted outside of the development timebox after development work had been completed and plugins were ready to be used by the larger Omeka community. Usability is a subjective concept, with many definitions offered by a variety of authors. Most definitions center around the concepts of effectiveness (Can users do what they need to do?), efficiency (How easily can users do what they need to do?), and satisfaction (How do the users feel about using the tool?). Steve Krug (2014) offers a concise definition of what makes something usable: "A person of average (or even below average) ability and experience can figure out how to use the thing to accomplish something without it being more trouble than it's worth" (p. 9). Usability testing seeks to assess how usable something is and uncover changes that would make that thing more usable.

The project grant funding gave the team the opportunity to hire a usability professional. The consultant created a usability toolkit for use in testing the Omeka Curator Dashboard project plugins and future technology projects. Formal usability tests can take a variety of forms, ranging from asynchronous lightweight testing through surveys to more labor-intensive moderated field testing where testers observe and document users attempting to accomplish a task on a Web site. The toolkit focuses on four methods: focus groups, A/B testing, online surveys, and moderated field tests. Guidance is provided for determining which test(s) to use based on the type of development project, research questions to be addressed, and point in the development cycle. Playbooks that provide detailed instructions and templates for conducting the test are included for all four methods. This toolkit will be made publicly available for use by other libraries seeking guidance and templates for conducting their own usability testing.

For the Omeka Curator Dashboard usability testing, the project team determined that online surveys and moderated field tests would best meet their testing needs. In their *Handbook of Usability Testing*, Rubin, Chisnell, and Spool (2008) suggest that although surveys can be used at any point in the development cycle, developing language that elicits a common understanding among all survey takers can be challenging, and surveys cannot provide the quality of empirical data generated through moderated field testing (p. 18). Surveys allow testers to potentially gather data from a large number of participants. Whereas the resources required for moderated field testing may restrict testing to three to five participants, surveys can reach tens or even hundreds of users. Surveys are time intensive on the front end: the survey designer should ensure that questions are understandable by testing the survey itself prior to release. This testing phase might result in multiple revisions of survey questions. Once finalized, testers will need to publicize the survey in order to generate sufficient responses. Moderated field testing, referred to simply as "usability testing" by Rubin et al. (2008) involves developing a series of research questions or test objectives, recruiting test participants who represent the end user, observing participants conduct a series of tasks, and collecting quantitative (e.g., time to complete a task, number of clicks) and qualitative (e.g., post-test interview questions measuring satisfaction) data (p. 25). These field tests generate a wealth of usability data, but are very time and resource intensive to conduct.

The project manager developed separate online surveys for all 15 of the Omeka Curator Dashboard plugins. Surveys were designed to address 1) the effectiveness of the plugin: was the user successful in making the plugin do what it was designed to do; 2) the efficiency of the plugin: how easily the user found the plugin to use; and 3) the overall satisfaction of the user: how well did the plugin meet the user's expectations. The team attempted to minimize the number of questions in each survey, avoiding any unnecessary "nice to know" questions. The surveys employed Likert scales, where 1 meant "Strongly Disagree" and 5 meant "Strongly Agree" in the required questions and optional open questions where users could enter context for their scaled answers in a text box. They publicized the surveys in several ways. First, they included links to the surveys on the Web pages where users download the plugins: on the official Omeka.org list of plugins and/or the plugin's page in GitHub. Second, they included links to the surveys on the Omeka Curator Dashboard project Web site. Third, they sent links to the surveys out to the Omeka developer group and to a variety of library technology listservs.

Due to the resources required to conduct moderated field tests, the team limited field testing to a selected group of plugins rather than all 15. They selected plugins to test based on their levels of popularity and complexity. YouTube is most popular plugin and was the first plugin for which the team conducted moderated field tests. They recruited three participants to test the YouTube plugin. The participants included two professional online exhibit curators and one undergraduate student with experience creating online exhibits in Omeka. Testing was conducted in an empty office at the University Library, and two members of the project team were involved in each test, one serving as the test moderator and the other as the observer. The field tests were divided into two parts: leading the participant through a series of tasks and a debriefing with the participant once the tasks were completed. For the first part of the test session, the moderator led the participant through a series of tasks using the Omeka plugin. The moderator followed a test script and encouraged the participant to think out loud and ask questions while engaging in the tasks. The observer silently watched the participant's screen and took extensive notes on how quickly the participant completed the tasks, points of confusion, questions or statements made by the participant, and "errors" made by the participant (e.g., selecting an incorrect option from a drop-down window, neglecting a necessary form field, etc.). For the second part of the session, the participant was presented with a list of words (both positive and negative) and asked to select the words that they felt represented their experience with the plugin. These selections served as a starting point for the debriefing conversation. The moderator asked the participant to explain their choices and attempted to probe the participants for more expansive descriptions of their thoughts and feelings while completing the tasks. During this debrief, the observer continued to take notes on the participant's responses. Once all three test sessions were completed, the project manager examined the notes taken by the observer. Based on these findings, the project manager created usability reports outlining changes needed in each plugin to enhance usability. These changes were divided into immediate needs and potential future enhancements. The project manager then created GitHub issues for each immediate need to be addressed by the developer.

FUTURE DIRECTIONS

The general principles of regular communication with users and attention to their input and feedback have been beneficial beyond the project detailed in this case study. Before the Omeka Curator Dashboard project concluded, the Digital Initiatives department was able to successfully make the case to UC Santa Cruz University Library and campus administration to permanently fund the library applications developer position. The short development cycles and iterations repeatedly demonstrated the progress and value of the project. This position has opened the possibility for continued development of Omeka and other open source software applications. The authors believe the user-focused agile project development method was a key component of the success of the project, and it will be used for all future development projects. The library continues to support Omeka for digital exhibits of archival materials, and the team has been working with researchers across the campus to create Omeka sites for courses. They've also developed additional Omeka plugins to support large research projects created and managed by faculty. This expanded the service, which includes graduate students, and has given the team insight into the different workflows and creative processes for different types of content creators. The team plans to include better support for helping researchers articulate their project goals during the needs assessment phase before the design and development stages begin.

Because usability principles are a major component of user-centered design, there is a growing movement to better incorporate usability features into the development cycle. Moreno and Yagüe (2012) argue that usability principles should be built into functional requirements for agile projects (p. 169), either as tasks within an existing user story or as separate user stories focused on usability (termed "usability stories") (p. 170–171). In future development projects, the UC Santa Cruz team will attempt to improve our functional requirements by incorporating usability principles. This might include the description of a button, the wording of a menu option, or the placement of explanatory help text.

CONCLUSION

The core project team at the UC Santa Cruz University Library found that involving a large team of staff stakeholders in the development of requirements and testing benefited the project in many ways. It increased the staff buy-in and professional investment in the project. Because the quick and iterative development cycle in agile project management focuses on delivering working software, staff were often able to see the results of their feedback within weeks of the initial needs assessment. The institution used the software as it was developed, and iterative development cycles supported that use. Users' stories were also extremely useful communication mechanisms, which helped the developers understand the needs of staff users of the software and helped the nontechnical users articulate their needs. Project team members can be so immersed in the details and iterations of a software function that they sometimes no longer see obvious flaws in the design. Usability testing helped the project team see their products through the eyes of a novice user. The project team also found that GitHub worked well for storing code and tracking issues from a variety of staff and sources. The GitHub Issues function gave the developer a clear list of tasks that could be prioritized by the rest of the project team. The team also found that the documentation from the deliverables, features, user stories, and acceptance criteria served as an extremely useful communication mechanism for all members of the project, including the project sponsors and library administration.

REFERENCES

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., …, Thomas, D. (2001a). *Manifesto for agile software development*. Retrieved October 25, 2016, from http://agilemanifesto.org

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., …, Thomas, D. (2001b). *Principles behind the agile manifesto*. Retrieved October 25, 2016, from http:// agilemanifesto.org/principles.html

Beyer, H., Holtzblatt, K., & Baker, L. (2004). An agile customer-centered method: Rapid contextual design. In C. Zannier, H. Erdogmus, L. Lindstrom (Eds.), *Extreme Programming and Agile Methods – XP/Agile Universe 2004.* Lecture Notes in Computer Science (Vol. 3134, pp. 50–59). Berlin Heidelberg: Springer. doi: 10.1007/978-3-540-27777-4_6

Cervone, H. F. (2011). Managing digital libraries: The view from 30,000 feet: Understanding agile project management methods using Scrum. *OCLC Systems & Services: International Digital Library Perspectives, 27*(1), 18–22. http://doi.org/10.1108/10650751111106528

Chang, M. (2010). An agile approach to library IT innovations. *Library Hi Tech, 28*(4), 672–689. http://dx.doi.org/10.1108/07378831011096303

Cohn, M. (2004, October 8). *Advantages of user stories for requirements* [Web log post]. Retrieved October 25, 2016, from https://www.mountaingoatsoftware.com/articles/advantages-of-user-stories- for-requirements

Corporation for Digital Scholarship. (2015). *Plans*. Retrieved November 3, 2016, from https://www.omeka.net/signup

Highsmith, J. (2009). *Agile project management: Creating innovative products* (2nd ed.) [Kindle version]. Retrieved from Amazon.com

Krug, S. (2014). *Don't make me think, revisited: A common sense approach to Web usability.* (n.p.): New Riders.

Kucsma, J., Reiss, K., & Sidman, A. (2010). Using Omeka to build digital collections: The METRO case study. *D-Lib Magazine, 16*(3/4). http://doi.org/10.1045/march2010-kucsma

Nowviskie, B. (2013). Skunks in the Library: A path to production for scholarly R&D. *Journal of Library Administration*, *53*(1), 53–66. http://doi.org/10.1080/01930826.2013.756698

Mamoli, S. (n.d.). On acceptance criteria for user stories [Web log post]. Retrieved November 3, 2016, from http://nomad8.com/acceptance_criteria/

Marsh, A. C. (2013). Omeka in the classroom: The challenges of teaching material culture in a digital world. *Literary and Linguistic Computing*, *28*(2), 279–282. doi: 10.1093/llc/fqs068

Moreno, A. M., & Yagüe, A. (2012). Agile user stories enriched with usability. *Lecture Notes in Business Information Processing*, *111*, 168–176. http://doi.org/10.1007/978-3-642-30350-0_12

New Media Consortium & Balboa Park Online Collaborative. (2015). *NMC horizon report.* Retrieved from http://cdn.nmc.org/media/2015-nmc-horizon-report-museum-EN.pdf

Owens, T. (2012.) Crowdsourcing cultural heritage: The objectives are upside down [Web log post]. *User Centered Digital History.* Retrieved November 1, 2016, from http://www.trevorowens.org/2012/03/crowdsourcing-cultural-heritage-the-objectives-are-upside-down/

Roy Rosenzweig Center for History and New Media, George Mason University. (2016). *Preparing to install*. Retrieved October 27, 2016, from http://omeka.org/codex/Preparing_to_Install

Rubin, J., Chisnell, D., & Spool, J. M. (2008). *Handbook of usability testing: How to plan, design, and conduct effective tests*. Indianapolis, IN: Wiley.

Sula, C. A. (2013). Digital humanities and libraries: A conceptual model. *Journal of Library Administration*, *53*(1), 10–26. https://doi.org/10.1080/01930826.2013.756680

Theimer, K. (2010). *Web 2.0 tools and strategies for archives and local history collections*. New York, NY: Neal-Schuman Publishers.

Tracy, D. G. (2016). Assessing digital humanities tools: Use of Scalar at a research university. *Portal: Libraries and the Academy, 16*(1), 163–189. https://doi.org/10.1353/pla.2016.0004

KEY TERMS AND DEFINITIONS

Agile Project Management: An iterative and adaptive method for managing projects that aligns with principles advocated by the Agile Manifesto.

Omeka: An open source content management system used for digital collections and exhibits. Omeka is a project of the Roy Rosenzweig Center for History and New Media at George Mason University.

Omeka Curator Dashboard (OCD): A suite of Omeka plugins that expand Omeka's functionality in the areas of collection building, description, management, and preservation. The OCD was developed by the University Library at the University of California, Santa Cruz.

Plugin: An add-on to the core Omeka code that expands the system's functionality.

Traditional/Waterfall Project Management: A sequential method for managing projects.

Usability Testing: Testing designed to evaluate the effectiveness and efficiency of a tool and the satisfaction of users when using the tool.

User-Centered Design: A design process in which the needs and wants of the user are considered prominently.