Reviving Digital Projects

What do you do when you are in charge of assessing and reviving an abandoned digital project you had no part in creating or implementing? This article will talk about the unique challenges and issues involved in such a project, drawing from a specific example at the University of Michigan Library. We contended with unfamiliar software, limited technical documentation, proprietary file formats and platform migration, and will discuss how we approached each of these specific technical issues. After reviving our project and reflecting on our process, we put together a list of guidelines that we feel will help assist others who may find themselves in similar situations.

by Dianne Dietrich, Jennifer Doty, Jen Green and Nicole Scholtz

Introduction

Librarians and professionals working in libraries are inheriting digital projects that they had no part in creating or implementing, including projects that have long since been abandoned. Given the unique challenges of digital information and data, including hardware and software obsolescence, a lack of transparency, and incomplete or insufficient metadata, reviving digital projects is a complicated and complex task that will require new technology-specific approaches and methods.

How do libraries address the assessment and revival of digital projects? What significant issues do they face throughout the process? Often, when digital projects are abandoned, very little documentation survives for those who will ultimately take over the project. The original creators may have left the institution, and it can be difficult to find individuals with the specific technical expertise needed to revive the project. When librarians and information professionals encounter a project that has long been abandoned, how should they determine the immediate next steps for the project? How do stakeholders determine if a project is worth saving, and how are soon-to-be revived projects technically evaluated? If significant changes in computing trends suggest new technologies better suited to the type of project, how do libraries handle this shift?

We will explore these issues and concerns, drawing from an example of a successfully revived digital project at the University of Michigan Library. In the winter of 2008, the ArcIMS service that ran the Online Atlas of Michigan Plants was migrated from an aging Sun server to a new server. The original database was established a few years prior, and its creator was no longer at the University of Michigan. Additionally, technical documentation for the ArcIMS software was limited at the outset. A team of librarians and interns evaluated the soon-to-be decommissioned database, determined the best methods for transferring the existing data, and finally migrated an improved version of the database to a new installation of ArcIMS on a new server.
In reviving our own digital project, we considered ways libraries and librarians can better restructure the lifecycle of digital projects to make it easier for their successors to continue established projects. This includes strategies for documentation that specifically address the complexity of digital data, methodology for determining maintenance needs for newly established projects, and finally how to effectively delegate responsibility for the care and feeding of these projects, so that they can persist and add value to the services libraries provide.

**Reviving the Database**

**History and background**

The Online Atlas of Michigan Plants[1] originated on an installation of ArcIMS that ran on a Sun server housed at the Graduate Library at the University of Michigan. The online service provided a web-based interface for users to explore the geographic distribution of plant species throughout the state of Michigan. Its audience included researchers, students, and others interested in botany.

In late 2007, the Sun server was slated to be decommissioned, so it was crucial to determine the fate of the Online Atlas of Michigan Plants. There was still demand for the service, as Spatial and Numeric Data Services at the University of Michigan continued to receive emails with questions about it, especially if the site experienced downtime. Since there was a demonstrated need for the service, it was clear that it should not be discontinued when the Sun server was taken offline.

Reviving the project, however, posed some considerable challenges. Its creator, Rachel Simpson, was no longer affiliated with the University of Michigan. While Simpson did provide extensive documentation related to gathering and digitizing the spatial data included in the service[2], there was a lack of specific technical information pertaining to the setup of the ArcIMS software. The librarians in Spatial and Numeric Data Services, while skilled in the use of a number of GIS applications, were not familiar with the ArcIMS software. It was also necessary to locate a suitable new home for the service, but it was unlikely that another Sun server would be available. Migrating the service would definitely mean migration to a new platform.

Given these challenges, it became necessary to formulate a strategy to migrate the Online Atlas of Michigan Plants to a new home. There were several ArcIMS services running on a Windows server at the Center for Statistical Consultation and Research (CSCAR) at the University of Michigan. This, then, was the ideal choice for the service’s new home. Once the new location of the service was established, our group convened in early 2008 to discuss the immediate next steps for the migration. Our plan was to have the service migrated to the CSCAR server by the end of the term, in April 2008. We scheduled a weekly meeting time to keep all members apprised of the status of the project, and established a group blog to post questions, notes, and future plans to share with the team.

**Understanding the original service, from the outside in**

Our first objective was to determine as much as possible about the original service while it was still active on the Sun server. This posed a significant
challenge, as none of us had direct access to the Sun server. In order to capture as much information about the service while it was still online, we opted to use TechSmith’s Camtasia screen capturing software to record a video as we explored the service’s web interface. We believe that video provided a significant advantage over static screen captures of the database interface, as it allowed us to record the sequence of events as we queried different parts of the database.

We also received a collection of files from the original server, though we were not sure which files belonged to our service and which did not, as we had no documentation or metadata available for them. We knew, from our experience using other ESRI products, how to identify several of the proprietary file formats, and how to view their content. We viewed as many of the files as possible in a plain text editor in an attempt to make sense of their contents. To keep track of all of the files, we kept a list with the name of the file, a summary description of its contents, plus notes about our own assumptions as to how the files might be related to the Michigan Plants service. As we learned more about the ArcIMS software, we became better able to understand how these files fit into the existing service.

Next, we faced an important question: was ArcIMS the best choice for this service, given that online mapping software and web-based interfaces had changed dramatically since the service’s inception? ArcGIS Server (also developed by ESRI) and the Google Maps API were two of the newer, more popular options for providing web-based access to geographical data. Based on the functionality that we observed on the original service, the needs of the community that used the service, and the existence of other ArcIMS services on the CSCAR Windows server, we decided that, despite the newer options, ArcIMS still remained the best option for this migration. Keeping with the same software would allow us to provide continuity to our users, ensuring that the functionality they needed would not be lost in an unfamiliar interface.

**Learning about ArcIMS**

In order to gain more experience using ArcIMS in its future server environment, we set up a virtual machine using VMWare that duplicated the environment of the CSCAR Windows machine. By using VMWare, we were able to easily restore the virtual machine to a pre-install state, so that we were able to practice installing and configuring each of the required components for ArcIMS (including Apache and ServletExec). Once the virtual server was set up to our standards, we were able to create videos that captured every screen throughout the process. We also created detailed documentation describing each of the steps necessary to configure the ArcIMS environment.

Our group blog served as an important record of our progress, as we noted any difficulties, frustrations or issues we encountered while working on the ArcIMS service individually. It also served to provide the entire team with detailed explanations of how we resolved errors during installation and configuration of the service. We incorporated some of this information into our reference documentation for future maintainers of the service.
Figure 1: Digital archiving can be tricky, and the blog enabled us to record our progress in deciphering the old service. [View full image]

We were able to locate training and online tutorials for ArcIMS, and we each completed the ESRI “Virtual Campus” series for the software. By completing the training modules, we developed a much better sense of how ArcIMS functioned and, most importantly, we learned which files were used to create a complete, functioning ArcIMS service. We were then able to review our original list of the files from the Sun Server, and determine precisely which ones we needed to re-institute our service.

Putting it all together

Once we had completed the online training for ArcIMS, we had a good sense of what we needed to do in order to get the service up and running in its new environment. First, we created a test service using our virtual machine, comparing its functionality to the video we created of the original service on the Sun server. When we were certain that we had captured everything from the first incarnation of the database, we easily transferred all of the map and configuration files to the new server and alerted the appropriate administrators to ensure that the service was visible on the web.

We sent an email to a select number of users of the system, asking them to evaluate the possible changes and improvements we were considering for the service, and to provide us with any additional feedback.
Customizing ArcIMS

We communicated what we learned about ArcIMS that would be helpful to the rest of the team.

Figure 2: We communicated what we learned about ArcIMS that would be helpful to the rest of the team. [View full image]

Future-proofing the project

We organized our ideas for potential improvements to the service.

Figure 3: We organized our ideas for potential improvements to the service. [View full image]
As the project came to a close, our last task was to provide documentation so that the future maintainers of the project could continue to keep the database alive. Our group blog already provided us with a wealth of information regarding the installation and configuration of ArcIMS, the structure of the Michigan Flora data, and suggestions for future improvements, but we knew we needed documentation that was streamlined and directly addressed the needs of the future maintainers.

We wrote three separate documents that addressed how to install the software and configure it specifically for the Online Atlas of Michigan Plants, and how to create new services for additional map layers of data, as well as steps to follow to make modifications to existing and future instances. We created Camtasia videos with narration that stepped through the process of installing ArcIMS and configuring and customizing the software. We included a comprehensive list of all of the files necessary for the service, including the directory structure, and a short statement of their purpose. These artifacts provided all of the information one would need to reestablish the service.

Figure 4: We originally discussed the idea of using video documentation on the blog.

We archived all of our blog entries, so that they could be consulted as a reference in the future.

Generalizable Issues When Reviving Digital Projects

Though abandoned digital projects can take many forms, by reflecting on our experience, we can generalize some of the issues we faced in reviving our digital project and provide a list of important questions to consider when tasked with reviving an abandoned digital project.
Evaluation

When considering whether to revive a particular digital project, an important question is how to evaluate the current service provided by that project. Evaluation includes determining if there is still a need for the service. If the project is located on the web, is it possible to obtain site traffic statistics to get a clearer sense of who is still visiting the page? Is anyone still sending queries about the service to the posted contact person? Who was the primary intended audience? Are they likely to still use the service? (If the primary audience was a class, perhaps there is very little current use for the service.) Another critical component of evaluation includes identifying limitations to the current service that must be changed if it is to be migrated. It might be necessary to initially pass over minor changes to the interface or usability issues, but errors that significantly affect the functioning of the current service must be resolved if the service is to be revived. It is important that updates and changes be prioritized, so that the migration and revival process goes more smoothly.

Technology

Another important issue facing professionals tasked with reviving a digital project includes determining if a new technology should be used. Often, when a project has been abandoned for an extended period of time, the technology used to originally create it has become obsolete.

Many projects created from third-party applications soon become obsolete as technology changes. The software vendor may have discontinued the software, or the number of updates since the version used in the project causes a complete incompatibility between files and data. In this case, it is critical to determine how using an outdated piece of software may affect the future maintainers of the project. If the vendor is no longer providing support, is there still a user community present for assistance? (This user community may exist at one’s home institution, or may be located on the web.) If it eventually becomes necessary to find a new piece of software to house the project, how easy is it to extract the needed files and convert them to the formats required by a new piece of software? (Some configuration files are written in plain text, or in XML, making them more portable than configuration files in binary file formats.)

For certain digital projects, significant changes in technology trends suggest new technologies better suited to a particular project. A regularly updated page, previously maintained by client-side HTML editing software, may be better served by blog software installed on a web server. In this case, it is critical to have tools in order to batch process multiple files for migration. It becomes important, when facing projects such as this, to survey the technological landscape, and examine the multiple new ways communities provide similar services. Which are closest to the spirit of the original digital project? What new features are available now, that were not available when the service was created? Is it possible to incorporate these new features with the original service?

Issues related to programming languages and platforms become important here as well. If the project consists of old code, what language is it in? Is there still expertise (and anticipated continuing expertise) in that language at one’s institution? Since the project’s inception, has there been policy enacted that standardizes the programming languages used in digital projects? How much time and effort would it take to convert a project from one programming language
to another? Can the project continue in its current environment or is it on a particular piece of hardware or software that is now outdated? If the project is to be revived, will there be compatibility issues between the old environment and the new environment that will complicate migration? Will it be possible to find a new home for the service?

**Migration**

When individuals inherit abandoned digital projects, often they have limited access to the original project. One of the most difficult tasks in the revival process is capturing the functionality of the older service without having direct and full access to it. It may be necessary to reverse engineer the service, by testing out each piece of it. It may be helpful to invest in screen capture software to record the results of interacting with the service. This may be even more helpful than notes, as it doesn’t rely on interpretation, and what was cryptic in the past may become clearer as one gets a more complete understanding of the project.

Is it possible to obtain all of the files used for the project? It may be necessary to examine, with a text editor, every file that can be read in that way. Diagrams of directory structures and notes for each file will prove invaluable later in the documentation stage. For binary files of unknown type, it may be possible to use utilities (such as “file” or “strings” in Linux) to ascertain the identity and content of any files that do not contain plain text. Lists of filetypes, including older and obsolete extensions, are readily available on the web.

**Restructuring the process**

How can libraries and librarians better structure the life-cycle of digital projects to make it easier for successors to continue established projects? Reflecting on the information we needed to revive the Online Atlas of Michigan Plants, we can offer a few suggestions for those implementing new digital services.

It is critical to provide extensive documentation. While one of life’s more onerous tasks, without any documentation, it becomes impossible for a digital service to be maintained long-term. It is important to know how the service was installed, what files are required for its functioning, how to interact and use the service, and how to troubleshoot common errors. This documentation need not be solely written. We found it helpful (and far quicker) to provide video and audio documentation, especially when describing installation and configuration processes. If it is not possible to create documentation as the project is being built, consider setting up a blog to allow for the opportunity to document along the way.

Any files that are required for the project should, if possible, be readable in plain text. Written description should accompany any non-text, or binary, or proprietary file. If a specific directory structure is important, then this should be diagrammed as best as possible. Consider creating documentation (both written and audio-visual) that takes a new user through the process of reassembling the service from its components on a new machine. This will provide nearly all of the critical information needed to maintain a project long-term.

If the project is coded, realize that, at some point, it may be obsolete. Are any standards put in place at the institution informing the creation of maintainable code? If not, create some general guidelines, and document these for future use.
Any non programming language specific notes or diagrams, if preserved, will provide future maintainers with a way to conceptualize the original vision for the project, without necessarily having to understand all of the technical details.

Finally, it is important to identify individuals who are responsible for specific maintenance tasks. Simply stating that one individual is responsible for maintenance is insufficient. Distinct and identifiable tasks should be enumerated, and individuals should be assigned to each of them. An example of a vague task is “updates software.” A better crafted, more specific task can be, “monitors security warnings and updates and applies necessary patches.” Specific statements make it easier to identify who is responsible for all of the various maintenance tasks needed to keep the service up and running.

**Conclusion**

It is important that digital projects persist, to add value to the services that libraries provide in order to connect users with information. No matter how complete the original planning was, some projects may end up in an abandoned state and will require revival months or years later. It is our hope that this document provides help and guidance for those tasked with bringing an aging digital service back to life. Additionally, our insights may be able to help those currently building new digital projects determine how best to provide all of the necessary information to the future guardians of these projects.

**Notes**


**About the Authors**

Dianne Dietrich (dd388@cornell.edu) is the Research Data & Metadata Librarian at Cornell University.

Jennifer Doty (jendoty@unc.edu) is the City and Regional Planning Librarian at the University of North Carolina at Chapel Hill.

Jen Green (greenjen@umich.edu) is the Spatial and Numeric Data Services Librarian at the University of Michigan.

Nicole Scholtz (nscholtz@umich.edu) is the Social Sciences and Spatial Data Librarian at the University of Michigan.

Subscribe to comments: For this article | For all articles

Leave a Reply