Source: Chalifour, J. (TBD). Evaluating Digital Tools. In P. Mitchem, D. Rice, & A. Gambill (Eds.), *Developing Digital Scholarship Services at Regional Comprehensive Universities*. McFarland & Company, Inc.

Evaluating Digital Tools

Joshua Chalifour

Abstract

Digital scholarship librarians must continually assess digital tools for the needs of researchers and other stakeholders. Selecting these tools without a well-defined, evaluation process risks their lack of use, wasted funds, and dissatisfied stakeholders. In this case study, the Concordia University Library had to decide whether to upgrade its preferred reference management tool or migrate to a new one. In order to find a solution that would best satisfy the users, the Library undertook a comprehensive evaluation process. A digital scholarship librarian applied a model, similar to what is frequently used in the private sector, to the academic situation and fit it for the constraints of the particular situation. The case study serves as an example of a process that DS librarians can use repeatedly to identify the strengths and weaknesses of a group of tools with respect to user requirements. It promotes communication with stakeholders thus reinforcing relationships among librarians and their research community.

Introduction

Fostering and managing digital scholarship (DS) initiatives entails regularly evaluating new software or technologies. These tools, essential to digital scholarship, can enable techniques and avenues in research that would not otherwise have been possible. However, the effort to wisely identify and select them is significant and comes with risks. An ideal evaluation of these tools can involve many stakeholders and phases but must contend with boundaries set by institutional circumstances or resources. How can we consistently practice evaluation processes that achieve stakeholder buy-in for the right tools?

Digital scholarship librarians sometimes operate in consultative roles, sometimes in collaborative roles[1]. Both roles encounter many new technologies that need to be evaluated for their fit, capacity to further research, and benefit to stakeholders. If an evaluation is undertaken with good understanding of the users' needs along with the broader implications of employing the tools, it stands a better chance of user adoption. Alternatively, acquiring these tools without a well-planned evaluation process risks stakeholders being unsatisfied, not using the tools, and funds being wasted.

I will describe a model for evaluating software and provide an example of its use at the Concordia University Library (a comprehensive university in Montréal). The model is based on my practice within the private-sector at an enterprise software analyst firm (prior to becoming a DS librarian). Such firms advise organizations on evaluating and selecting large-scale systems that often involve significant changes to business processes. Although I describe the evaluation to change reference management

[1] Cox, "Communicating New Library Roles to Enable Digital Scholarship."

tools in this case study, the model may be adapted and applied to other digital tools and circumstances within academic milieux.

Literature Review

Literature on a generic model for evaluating software in academic situations is relatively uncommon. More commonly, the literature addresses specific types of software (inside and outside of academia). Eastham et al. describe a process for evaluating a product lifecycle management (enterprise software) system. They describe steps that include the following: 1) identifying the systems available on the market, 2) screening which vendors would be appropriate, 3) developing a hierarchical decision model, in which they asked their expert stakeholders to weight the importance of high-level categories of criteria, and 4) contacting the vendors that scored highest to obtain additional information.[2] They emphasize eliciting feedback and gaining buy-in of the application.

We've all probably encountered a situation in which people unhappily use a tool, implemented from a non-consultative decision, and which does a poor job satisfying their requirements. Thus, we should consider what that implies for our roles in the evaluation process. Jim Connelly, reflecting on selecting a records management tool, states the following.

> "...each software package has its own strengths and weaknesses. Consequently there is no 'one' perfect piece of software that provides the perfect solution for everyone's problems. This is precisely why some companies employ consultants to review their needs objectively and select the best possible software."[3]

As DS librarians, we may function similarly to the objective consultant[4] and we face some risks that we want to mitigate as much as possible. Susan Sherer focuses on managing the risk of failure from deploying purchased-software. She argues that a failure risk increases when acquiring software that doesn't meet its users' functional needs.[5] Rosendahl and Vullinghs recommend identifying risks in the software acquisition process early, when there is more possibility to manoeuvre toward a positive outcome.[6] Their well-considered taxonomy of software acquisition risks includes the human elements of technical and people skills, the "volatility of project participants," and uncertainty around requirements among a variety of other risks.[7] Although they do not suggest solutions, it's prudent to anticipate these risks from the outset when undertaking an evaluation. Identifying the right stakeholders and determining the right sorts of communications is important to obtaining well-understood requirements.[8] We can consider our stakeholders based on the ISO framework applying to software quality and user interaction, which addresses the following types of user needs.

> "1. Primary user: person who interacts with the system to achieve the primary goals.
>
> 2. Secondary users who provide support...

---

[2] Eastham et al., “PLM Software Selection Model for Project Management Using Hierarchical Decision Modeling With Criteria From PMBOK® Knowledge Areas.”
[3] Connelly, “The Sargasso Sea of Records Management Software,” 21.
[4] Lippincott and Goldenberg-Hart, “Digital Scholarship Centers: Trends & Good Practice,” 5.
[5] Sherer, “Purchasing Software Systems: Managing the Risk,” 258.
[6] Rosendahl and Vullinghs, “Performing Initial Risk Assessments in Software Acquisition Projects,” 147.
[7] Rosendahl and Vullinghs, 149.
[8] Coughlan, Lycett, and Macredie, “Communication Issues in Requirements Elicitation: A Content Analysis of Stakeholder Experiences,” 526–28.

3. Indirect user: person who receives output, but does not interact with the system."[9]

Because determining what will satisfy the needs of our users reduces the risk of a failed evaluation, the work identifying stakeholders is of considerable importance.[10] Burnay et al. state that "...one possible way of selecting stakeholders is to account for their respective *level of commitment to the RE [requirements engineering] project*, i.e., how the stakeholders are intellectually or emotionally bounded to the RE project."[11]

In some evaluation settings, teams can be formed with stakeholders from different departments. Though not apt for all evaluation projects, discerning what sort of engagement stakeholders would have with the tool helps identify who to involve. Evaluating digital library systems, Goh et al. considered users' work processes. Referring to librarian and patron stakeholders, they state "...a thorough understanding of the users of libraries and the system itself should be obtained."[12] Indeed, we need to understand the goals in our stakeholders' workflows. Greta Kelly, evaluating educational technologies, argues against starting a project by comparing software features or costs in favour of finding out how well the software supports learning goals.[13] While that position applies to learning, the broad concept remains that the work people undertake should dictate requirements for the software.

To improve our stakeholders' ability to achieve their goals, we need to analyze their business processes and elicit their requirements. Literature on requirements tends to reference software engineering practices in which requirements elicitation impacts design, development, and other issues.[14] Because requirements elicitation helps to understand problems that users experience in their work processes, it applies nicely to projects evaluating existing software. Şen et al. explain that they assemble a team of stakeholders to represent requirements "...in a list of process and business related functions, scenarios or use-cases..."[15] Von Scheel et al. state "A business process is a collection of tasks and activities (business operations and actions) consisting of employees, materials, machines, systems, and methods..."[16] They break down processes into subprocesses and tasks. As Pazak and Beshansky discuss the value of evaluating business processes, they point out how involving employees from different parts of the organization in the project—particularly when it changes their processes—benefits the likelihood that people will want to adopt the new system.[17] They also state that "Requirements can be ordered in terms of priority, and then mapped to the functionality of prospective software choices to help determine both selection and shape of an effective deployment."[18] Thus, eliciting requirements based on stakeholder business processes helps us determine on what we should compare tool alternatives as well as mitigate the risk of selecting something that does not satisfy our users' needs.

A taxonomy developed by Achimugu et al. shows that there are many ways to prioritize requirements; quite a few of which have a basis in Saaty's analytic hierarchy process (AHP).[19] Saaty directs us to

---

[9] International Organization for Standardization and International Electrotechnical Commission, *Systems and Software Engineering: Systems and Software Quality Requirements and Evaluation (SQuaRE)*, 5–6.
[10] Glinz and Wieringa, "Stakeholders in Requirements Engineering," 18.
[11] Burnay, Jureta, and Faulkner, "How Stakeholders' Commitment May Affect the Success of Requirements Elicitation," 1.
[12] Goh et al., "A Checklist for Evaluating Open Source Digital Library Software," 360.
[13] Kelly, "A Collaborative Process for Evaluating New Educational Technologies," 106.
[14] Franco, "Requirements Elicitation Approaches."
[15] Şen et al., "An Integrated Decision Support System Dealing with Qualitative and Quantitative Objectives for Enterprise Software Selection," 5274.
[16] von Scheel et al., "Phase 1: Process Concept Evolution," 1.
[17] Pazak and Beshansky, "Six Steps To Successful CRM-CTI Deployment," 48.
[18] Pazak and Beshansky, 49.
[19] Achimugu et al., "A Systematic Literature Review of Software Requirements Prioritization Research," 579.

"decompose" a decision into components that define the problem we have or information we need, create a hierarchy of criteria, then undertake a pairwise comparison process to determine priorities for the decision.[20] While my object here is not to delve into the benefits or drawbacks of a particular prioritization or decision-making technique, the model I will describe and its case study incorporate some of these ideas (though not intended as examples of AHP). Şen et al. argue that "...a systematic and repeatable selection methodology is a crucial need for minimizing the uncertainty and risk."[21] In the following, I hope to emphasize stakeholder input, while offering a model that can be repeatedly used.

The Evaluation Process

The decision to evaluate some software could initiate from stakeholder sources or from our own oversight. In a case initiating from stakeholders, there could be a problem that one or more people have, which a digital tool would help solve. We'd want to apply some precision, probing that problem. This is somewhat analogous to the work that librarians do during a reference interview, except here we want to identify what the person needs to accomplish, what is their goal and what processes to they undertake to achieve it.

For example, a stakeholder (professor) raised a question about our library providing access to one particular LaTeX tool. Upon further inquiry it turned out that he needed to collaborate on LaTeX documents with colleagues. Exploratory research led to the discovery of at least seven alternative tools to the one the professor had requested. This sort of query could result in a project initiated by user needs. However, before rushing to implement the answer that had been suggested with the query, we'd want the due-diligence of an evaluation. That way we gain an understanding of other stakeholders' needs and what would best satisfy them.

In the case of our own oversight, we could initiate an evaluation due to knowledge we acquire via our overarching concern for, or practice of digital scholarship. We stay current with emerging digital techniques and tools. Thus, we learn of things that our community would value. From this perspective we're proposing things that address the needs of, or open new possibilities for our community.

Regardless of the source that initiates a software evaluation, it's essential that we understand the users' perspectives. First, so that we can identify and analyze requirements for the software. Second, because we ought to communicate with stakeholders in such a way that they feel certain their needs are heard, understood, and accounted for. As mentioned, a risk in selection comes from the human factors. To mitigate this risk we want to infuse our evaluation process with communication, involving stakeholders.

The Model

This model comprises eight key parts that tend to recur within software selection methodologies.[22] I will provide detail about each part while explaining how I applied them in the case study example of selecting a reference management tool. Steps A – H are a practical sequence but depending on the situation, some parts could be undertaken simultaneously or in an alternate sequence.

A) Identify the problem

[20] Saaty, "Decision Making with the Analytic Hierarchy Process," 85.
[21] Şen et al., "An Integrated Decision Support System Dealing with Qualitative and Quantitative Objectives for Enterprise Software Selection," 5272.
[22] Mohamed, Ruhe, and Eberlein, "COTS Selection: Past, Present, and Future," 1.

B) Identify the stakeholders
C) Research the ecosystem in which relevant tools are produced and used; gather background information about the technologies
D) Identify user requirements
E) Define evaluation criteria and express them in a hierarchy for comparison and decision-making
F) Determine rating scales and rate the alternatives
G) Prioritize the hierarchy of criteria
H) Analyze all factors related to the technology and interpret the results

Case Study: Applying the Model to Evaluate a Reference Management Tool

A) Identify the problem

For well over a decade, the Concordia University Library had been using and supporting the RefWorks application as its preferred reference management tool. As RefWorks' parent company, ProQuest, prepared to launch a new version of the tool, we faced the imminent challenges of upgrading and migrating our community to the new version.

Concordia librarians had incorporated RefWorks into subject-specific instruction sessions. We offered regular workshops on using RefWorks in aid of research and writing. In addition to the library's instruction and support for RefWorks, some departments in the university used it heavily for research projects or as a form of online catalogue for their own collections.

The impending migration would disrupt regular usage, requiring outreach, librarian support (new workshops, documentation, etc.), and training for a significantly different interface. This made it an opportune time to consider whether to continue using RefWorks or to put the same sort of effort into supporting an alternate tool.

RefWorks served many of our community's needs but we knew the tool had shortcomings, some of which were recurring complaints from users. As part of our ongoing awareness of the ecosystem for scholarly tools, we were tracking what other tools could do. We also knew that as our community experiments with new types of scholarship techniques, it would be important to support innovative tools that enabled those techniques. Thus, we initiated an evaluation of the new version's features against what other similar applications could provide.

B) Identify the stakeholders

Changing reference management tools would have a potentially significant impact on our community's research workflows but probably not a critical one. In fact, we understood that options existed for individuals to potentially continue using the former tool, even if we opted to change what the library officially supported. Ideally, we'd have identified a range of stakeholders to interview about the processes in their research workflows pertaining to reference management. Some institutional constraints prevented wide-scale interviews or surveys for that purpose at that time. We devised and pursued alternate techniques to identify our stakeholders, elicit requirements or understand priorities, and later to ensure ongoing communication.

We acquired descriptive information about our user community (roughly 30,000 accounts) through the high-level statistics tracked by the RefWorks application. While most of the accounts were registered to undergraduates, only about twenty percent of those had been used within the last couple years. Over thirty percent of graduate accounts had been used within that time. This could indicate that since graduate students are likely to be more heavily involved in submitting research-heavy assignments or other research work, they rely more on tools that support these processes. In addition to faculty (nearly 20 percent of their accounts had been used within two years), this helped to build context for the processes that people would likely be engaged in while using the tool.

Identifying how many references were stored by the different types of users and how much storage space they used for saving documents was an indication as to how heavily each group of users relied on the tool, over two-thirds of our users stored fewer than 65 references in their accounts—minimal usage. Likewise just 546 accounts used more than 2 MB of storage and only 16 used more than 100 MB. This information helped us to picture the level of commitment that different types of users had in their RefWorks accounts. Essentially we understood that groups with higher quantities of references in their accounts tended to be graduate students and faculty researchers, reinforcing the notion that they'd be key stakeholders.

C) Research the ecosystem and gather background information

In this step of the model, we want to discover what environmental forces continue to make a tool useful or threaten it with obsolescence. This can influence the way we understand a tool's capacity to address user requirements as well as the set of criteria on which we will compare the tools.

According Adomavicius et al. a technology ecosystem includes both similar technologies and related (competing or supporting) technologies.[23] They recommend extending that scope to consider social and governmental forces, technical forces, and economic forces on the technology.[24] Although they do not include it in their ecosystem model, they mention the "firms or agents behind the technologies"[25] and I believe that those are necessary parts of the ecosystem to research for our process.

Making sense of the ecosystem that produced the technology products and services provides insight about what motivates developers and their organizations. Gaining an awareness of the tools and practices of other institutions helps us to consider the part we play influencing what is developed. These things help us to analyze the direction that developers are likely to move toward so that we can make our best attempt at intelligently anticipating and preparing for the tools, trends, and practices on the horizon.

I began developing a long list of tools that could be used for reference management. The list covered a very broad definition of reference management. For example, simple online citation formatting tools were included, as were some tools that went as far as to also provide mindmapping functionality.

This initial long list was based on general search engine results and tools identified from other public comparisons, articles, or relevant literature about citation and reference management issues. I reviewed articles that explored not just functionality but practices within other academic libraries because understanding different practices could lead to considering alternate applicable tools.

[23] Adomavicius et al., “Technology Roles and Paths of Influence in an Ecosystem Model of Technology Evolution,” 189.
[24] Adomavicius et al., 199.
[25] Adomavicius et al., 200.

In addition to those sources of information, I reviewed the websites of 46 academic libraries to determine which reference management tools they support and to what degree they offered support. Four tools were commonly mentioned while a few others occasionally made an appearance. The degree to which each of these tools were supported overall helped to paint a picture of the direction in which our academic environment is moving. Ultimately, I reduced the long list to eight alternatives. These tools recurred in the environmental scan and had concrete evidence of ongoing development, factors which supported their selection as candidates for in-depth comparison.

Information about the core technologies and their providers was useful toward shaping the final decision.

Core Technology

Obtaining details about all the possible features was not needed at this point but identifying core functions that delimited the tools was useful. This maintained a track toward comparing similar applications. For example, one tool might be a fully fledged reference management tool, with advanced functionality for handling bibliographic information whereas another might be a very simple website that generates a citation to copy and paste. These compete in some ways but do not serve equivalent uses for comparison.

Providers

Understanding the operating models and background of the tools' providers was important for validating the sorts of support we could expect. It also enabled considering factors like the sustainability of their operations and hence the product's longevity. For example, I verified whether the tool was developed by a commercial entity or other sort of organization. That background information provides context for the way the software is developed. It allowed me to describe and evaluate what could be expected from each organization.

The following criteria supporting these contextual concerns were answered primarily from information available on the providers' websites.

i. Date of the tool's most recent release

> A tool that had not had a new version released or even security and bug patches within a couple years, was a warning that the organization was not actively improving the tool, its priorities had shifted in another direction, or perhaps the tool had been abandoned/discontinued. Regardless of the reason, choosing a tool that was not being actively worked on would be a poor choice with respect to its sustainability.

ii. Availability of:

- service (paid-service or other options?)
- online community (active user forum, open bug reporting, feedback options?)
- documentation
- training (live or online classes, or videos)
- implementation services or other consulting support

iii. Location of:

- Corporate headquarters and regional office (as applicable)
- Location of hosting servers (if applicable)

Knowing the location was helpful to engage in discussions with the provider and for considering support availability (were there people within our time-zone to help?). It was also useful to consider legal implications. For example, different laws around privacy and data security may apply, particularly if using cloud storage outside of the local jurisdiction.

iv. Funding entity or backing enterprise

Determining whether the application had the backing of one or more commercial entities impacted some of the other criteria. Knowing about this in advance meant that the evaluation could be structured to take into account aspects of the commercial offerings, which I'll expand more in the final analysis.

It also raised the necessity of discussing with the vendor, what exactly the company intended to provide. For example, would they charge additional fees for support services? There are commercial entities that focus on free and open source software. FOSS commercial business models frequently focus on support since, due to the obligations of free and open source software (FOSS) licences, it's often not relevant to sell software by licence instance. Understanding the difference in these entities' business models compared to proprietary vendors, clarified what was being offered. In this case, the commercial aspect was essentially for a cloud hosting service to extend users' storage options. Also, having some form of commercial backing could have proven important, especially around the consideration of support.

v. Licence

Often licence information could be determined from the tool's website. Proprietary tools had information about licensing restrictions. Free or open source applications provide the application's source code—areas which typically identify the licence. Recording the licence type with the other information about the application being tracked made it easier to evaluate the various strategies for acquiring and implementing the tools. It dictated to some degree, the sorts of involvement that would be needed from other members of our institution (e.g. stakeholders in financial services or IT). It also gave greater insight on what to consider as we anticipated future trends likely to unfold within the software's ecosystem.

D) Identify user requirements

This evaluation involved assessing and understanding user requirements early in the process. In a comprehensive, ideal evaluation we could have had a project team that included stakeholders from our community and learned about the steps they took to accomplish their research work. We'd record and analyze those steps to understand their processes, mapping those to the types of functionality that would support their goals. Indeed, for a more complex software system with a greater impact on our community, this procedure would be paramount to reduce risk. Due to constraints we had during the early part of this project, we came up with the following alternate techniques to obtain an understanding of users' processes and requirements.

We regularly offered library workshops for learning RefWorks, which were well-attended. I designed an additional workshop on the topic of evaluating and selecting a reference management tool. Both of

these workshops, while helping the students that attended them, were opportunities to learn about their needs. At the start, students responded to a questionnaire about some of their practices researching, including citing in their writing and their knowledge of the tools. We collected 42 graduate student (Masters and PhD students) responses and 7 undergraduate responses to the questionnaires so along with noting the topics of their questions during the workshops, we discerned what some of their major requirements were.

The information gathered from the workshops contributed toward understanding key issues in their research workflows and behaviours toward managing citations. It included information about the degree to which they:

- needed help working with citation styles, formatting their references, and creating bibliographies
- thought functionality from a broad list of categories were important to their work
- sought help from librarians on citation and bibliography issues
- had difficulty keeping track of bibliographic information
- needed to collaborate on documents or otherwise share research reference information
- used library web resources to get help managing reference types

They identified the degree to which they were familiar with alternate reference management tools, which spanned from having never heard of a particular tool to using it regularly. That information enabled discussion on features from different tools, which further revealed their requirements.

Ultimately, combining the information from the survey responses, student commentary during workshops, and feedback collected from stakeholders and colleagues over time, led to a richer understanding of user needs.

E) Define evaluation criteria and express them in a hierarchy for comparison and decision-making

Defining a clear, comprehensive set of criteria on which to evaluate the tools is a detail-oriented project, consuming more time than some of the other steps. It entails gathering information from disparate tools and making sense of that information in a way that can be applied across the tools. For large-scale evaluation projects one would want to issue a request for information (RFI) to vendors, soliciting information about how their products support the requirements.

Organizing criteria into a hierarchical structure ensured a couple things. First, it helped to clearly identify overall aspects of each tool that would suit the needs of our users. Applying the same set of criteria to all tools, forced the tools to conform to our users' needs. This was in contrast to fitting the users' needs to what the tools provided, which would have resulted in a difficult comparison, perverted to each tool’s idiosyncratic design. Second, we could drill down to sections of the hierarchy and pinpoint how well each tool satisfied the exact same requirements in comparison to its alternatives. This was important because it enabled us to retain control over an evaluation that vendors otherwise try to skew, emphasizing criteria most favourable to their products.

An RFI would not have been appropriate for the scope of this evaluation. Instead, taking the following steps led to discovering the individual features and functions[26] of the tools.

1) Reviewed the product's web page, it usually promoted key features.

2) Searched for product white papers (sometimes marketing documentation), which normally explain the technology but these did not always exist.

3) Tried a demo version of the product. Sometimes proprietary applications offered versions of the tool on a trial or limited use basis, which were useful ways to determine much of a tool's functionality. When the tool was FOSS, I also installed it. For software that was hosted as a service, there were always free or trial accounts available.

After gaining access to the tools, I systematically went through their menus and other user interface elements to discover and document the features. The downside to this approach is that depending on the complexity of the tool, it occasionally required more effort to discern its functionality.

4) User manuals and training documentation provided detailed information about the features of the software.

5) Online fora provided details on functionality. Bug tracking systems usually identify certain features of a tool in service of the problem being worked out, however these tended to be too granular. User fora are full of questions and answers from people already using the tool. The questions that surfaced from these fora were useful not just for identifying functionality but also for understanding the way people use it.

6) Software comparison websites provided insight on some functionality but these were usually not granular enough to be of much use. Additionally, other university library websites occasionally provided copies of their own comparisons (helping students make decisions). Those comparisons tended to be high-level, providing sparse detail about functionality.

7) It's possible to buy reports from software analyst or consulting firms, which detail the functionality of various like-tools. This option wasn't pursued because reference management tools are not the sort of technology that those firms typically analyze.

After capturing the features and functions of the tools for comparison, I organized them into parent/child relationships. Top levels were broad functional categories, which branched to sub-categories, and eventually at the bottom level, 106 leaf criteria represented individual features.

This hierarchy would be used for the dual purpose of rating the products and assigning users' priorities. It essentially, models the decision to be made as a result of the evaluation.

F) Determine rating scales and rate the alternatives

To consistently compare like-criteria across the tools and to get a definitive result, I applied a numeric rating scale across the criteria. Features for each tool were rated based on how completely they supported each criterion. Different scales may be more appropriate for other tools but the following scale was sufficient for evaluating reference management tools.

---

26 Nicholas Joint's insightful article "Evaluating Library Software and Its Fitness for Purpose" distinguishes objective function from subjective. I use "functionality" here in the objective sense, referring to the properties contained within a technology.

Rating	Points
Supported	3
Support via add-on	2
Future support	1
Not supported	0

If a tool supported something, it would receive full points. If some add-on component or third-party plugin for example, existed and could be installed with the tool enabling it to provide that functionality it would get a point less because that opened greater possibility that there would be integration problems or loss-of-support in the future: the add-on introduces more uncertainty and complexity to manage. If something was not currently supported but was planned on the vendor's roadmap for imminent release, then this rating provided a point to account for the trajectory of the product in favour of one that would not offer the feature at all.

These ratings applied to a relatively easy-to-distinguish situation, which could be objectively verified against whether or not a given tool had the specified functionality. Considering a criterion about something like user interfaces (UI) is more complex. To some degree, UI criteria could be addressed through objective design practices and assigned points. For example, if tips were available next to different interface functions of the tool, those would be considered support for “Contextual help.”

Evaluating which tool had a user interface that our community favoured, would have required making users’ subjective experiences explicit. Ideally, we could have organized demonstrations and asked users to provide feedback about their preferences on a scale such as: better than average = 2 points, average = 1 point, and below average = 0 points. This would have helped ensure that we decided on a tool they liked using. Due to the constraints mentioned earlier and the scale of this evaluation, we focused on rating objective criteria.

G) Prioritize the hierarchy of criteria

The best tool is not the one with the most features, it is the one with the features that best satisfy users' needs. The point values assigned in the previous step merely reflected how each tool satisfied the criteria of the comparison. The other aspect of evaluating the tools, was to prioritize how important the criteria were. Some features we cannot live without while others are somewhat useful or simply unnecessary for the users.

It was important to return to our community's requirements and use them to adjust how much the functionality would influence a decision.

Assigning priorities to the criteria caused some to have a greater or lesser impact on the evaluation when calculating the results. The following table of priorities used a higher-granularity technique[27] to express how important each criterion or group of criteria was.

[27] Benestad and Hannay, “Does the Prioritization Technique Affect Stakeholders’ Selection of Essential Software Product Features?,” 262.

Priority	**Value**
Necessary	5
Very important	4
Important	3
Nice-to-have	2
Not important	1
Unnecessary	0

If something was necessary, it had the greatest impact on the evaluation because people using the tool could not do without it. Thus, tools that do not support this criterion in a satisfactory way were penalized to a greater degree than if they lacked support for a criterion that people find merely nice-to-have (but could conceivably do without).

After assigning priorities to each group of criteria or individual criterion in the hierarchy, their values could be multiplied by the points assigned to each tool for the degree of support it provided to that criterion. For example, a tool that **Supported** a **Very important** feature would gain 12 points (3 × 4) whereas a tool that satisfied that feature via an add-on would gain only 8 (2 × 4) points.

Tool	Priority	Tool A: Points for Criterion	Tool B: Points for Criterion
Criterion 1	Very important (4)	Supported (3)	Future support (1)
Criterion 2	Not important (1)	Future support (1)	Add-on (2)
Criterion 3	Nice-to-have (2)	Not supported (0)	Add-on (2)
Score without priorities		4	5
Score with priorities		13 *(3 × 4) + (1 × 1) + (0 × 2)*	10 *(1 × 4) + (2 × 1) + (2 × 2)*

Table 1: Example table showing the impact of prioritizing criteria according to user needs.

If we didn’t consider the priorities in the preceding table, Tool B would score higher for its functionality. However, some of those points were for criteria with less importance, which only becomes apparent upon considering the users' priorities. Calculating the total score with respect to the priorities shows that Tool A earned more points from the criteria that matter most, making it the better choice.

I used a simplistic prioritization method in which I set a maximum total priority value (1000) for the entire hierarchy. I distributed a portion of that value to each group of functionality based on how much that group should influence the evaluation relative to the others. That influence was interpreted so that it reflected the survey results and other feedback collected from our users. I distributed each group's

priorities as they mapped on to the criteria within, still aiming to reflect users' needs. This task could be achieved for the 106 functional criteria used to compare the tools. Evaluating a complex system with hundreds or thousands of criteria requires a suitable prioritization technique, which would be more involved than this example.[28]

H) Analyze all factors related to the technology and interpret the results

While sophisticated decision analysis tools exist[29], in this case a spreadsheet worked well to capture, organize, and analyze the information discussed. After constructing the hierarchy, rating the alternatives, and assigning priorities for user needs, it was possible to analyze the following areas: features (number and quality), infrastructure, trials or demonstrations, user/researcher preferences, provider background and sustainability, support, training, and cost.

The hierarchy was organized as follows. Other evaluations could include more than the three main categories listed (e.g. demonstration feedback and reference check information).

1 Background (18 criteria including provider information, licence, support, etc.)

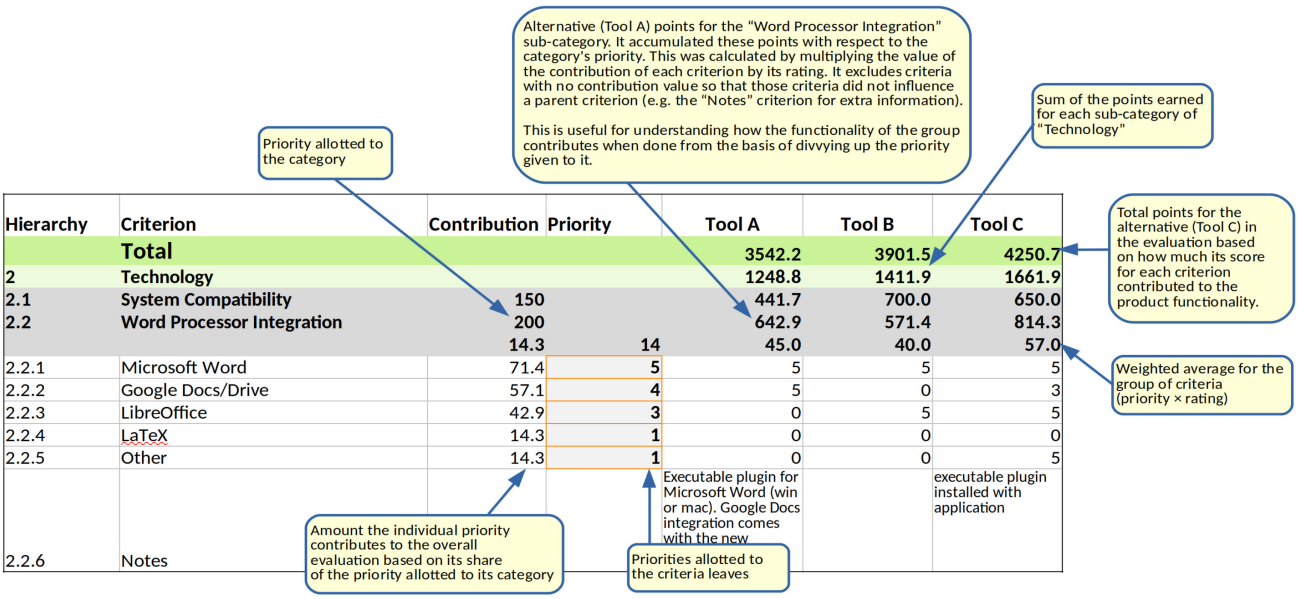
2 Technology

- 2.1 System Compatibility (12 criteria)
- 2.2 Word Processor Integration (5 criteria)
- 2.3 User Interface (8 criteria)

3 Functionality

- 3.1 Import methods and Academic Database Integration (24 criteria)
- 3.2 Import formats (8 criteria)
- 3.3 Export (8 criteria)
- 3.4 Citation styles (9 criteria)
- 3.5 Simple bibliography creation (6 criteria)
- 3.6 Collaboration (10 criteria)
- 3.7 Reference and File Management (16 criteria)

[28] Fernandes, Rodrigues, and Costa, “Comparing AHP and ELECTRE I for Prioritizing Software Requirements,” 7.
[29] French and Xu, “Comparison Study of Multi-Attribute Decision Analytic Software,” 77.



Hierarchy	Criterion	Contribution	Priority	Tool A	Tool B	Tool C
	Total			**3542.2**	**3901.5**	**4250.7**
2	**Technology**			**1248.8**	**1411.9**	**1661.9**
2.1	**System Compatibility**	**150**		**441.7**	**700.0**	**650.0**
2.2	**Word Processor Integration**	**200**		**642.9**	**571.4**	**814.3**
		14.3	**14**	**45.0**	**40.0**	**57.0**
2.2.1	Microsoft Word	71.4	**5**	5	5	5
2.2.2	Google Docs/Drive	57.1	**4**	5	0	3
2.2.3	LibreOffice	42.9	**3**	0	5	5
2.2.4	LaTeX	14.3	**1**	0	0	0
2.2.5	Other	14.3	**1**	0	0	5
2.2.6	Notes					

*Figure 1: Example of hierarchy, priorities, alternatives and their ratings (**Word Processor Integration** group expanded to show leaf-level criteria)*

Of the eight tools, four best satisfied the user priorities and were shortlisted. The remainder fell too far from the priorities to consider further.

Next, by including the cost information, it was possible to produce a cost/benefit analysis. The following table shows an example high-level summary of four alternatives based on their functionality and cost.

Tool	Prioritized rank (best set of functionality)	Best value	Price from vendor	Cost per point	Recommended price based on value
Tool A	3	3	$26,000	$7.34	$7,506.11
Tool B	2	2	$21,000	$5.38	$8,267.49
Tool C	1	1	$9,000	$2.12	$9,000.00
Tool D	4	4	$25,000	$8.65	$6,122.18

The **Prioritized rank** column, reveals which tool received the most points for the functionality it offered with respect to what was most important to users. Tool C offered the best set of functionality, so if the only thing we cared about in making our decision was the tool with the best functionality for our users, we'd select Tool C.

The **Cost per point** column shows how much each point of functionality that the tool garnered is worth in dollars. This is calculated by dividing the total cost of the tool by its total points.

The **Recommended price based on value** column shows that if a tool were to be competitive against the lowest priced alternative (in this case, Tool C), then the tool should be re-priced at the amount

designated. It was calculated by multiplying the tool's total points by the lowest priced alternative's *cost per point*.

The **Best value** column ranks which product would be the best to acquire based on the value it provides for the price the vendor is charging. It was determined by ordering the products in a sequence where the best (#1) product has the smallest gap between its actual cost (the price from vendor) and its recommended cost.

Seeing the evaluation results side-by-side like this makes it easier to justify a decision. When we consider the best value, it becomes clear that Tool C provided the most applicable set of functionality at the lowest cost to the Library. Suppose we had only compared tools A and D. Overall Tool D costs less than tool A so if a decision was based merely on overall price from the vendor, the Library would choose Tool D. Looking at the **Cost per point** column shows us that in fact, per each point of functionality, Tool A costs less than Tool D and it would better satisfy our users' needs.

We could use the **Recommended price based on value** column to negotiate a better price from the vendor. If we had reason to select a tool that was more expensive than we'd like, we could tell the vendor that based on the functionality it provided, it was overcharging for what the product delivers. If it wished to compete with the other vendor then it would have to lower its price to be equitable.

This process enabled me to report on our users' requirements, our current situation supporting them, the options that we could take to upgrade or else migrate to an alternate tool, and the trade-offs for doing so. The report addressed the results of the criteria ratings and priorities and it facilitated creating charts based on the values in the spreadsheet. The ultimate result was a clear and justifiable recommendation based on our user needs.

Post-evaluation steps

Having put together this information, we were keenly aware that not all stakeholders had been fully represented. It was important to secure broad support toward migrating to a new tool. To begin that process the next phase was to present the evaluation results to the librarians and information services personnel since they were involved with reference management teaching or support.

Upon presenting the evaluation process and its results, the meeting became an opportunity to get more input. People raised issues that they knew would be concerns for the faculty they worked with or that they thought might need to be addressed if we were to migrate to a different tool. Seeing the evaluation results, people favoured migrating to a new tool, Zotero. The library assembled a team to develop a project plan for migrating to the new tool within a year (the time until the former tool's agreement would expire).

1. Outreach

In order to be successful, we wanted to reach out early in the migration and with appropriate frequency to all stakeholders, including other primary users, secondary (support) users, and indirect users, which were explicitly stated in the project plan. This enabled communication with stakeholders that we had not been able to consult. It also opened the possibility of changing course if those stakeholders brought to light needs that the new software could not satisfy. As mentioned in the literature review, this is a risk that results from not being able to fully consult with all representative stakeholders in advance. Under the constraints and best-efforts taken for a comprehensive evaluation, it was a manageable risk.

Outreach included e-mail campaigns informing users of the plan and soliciting further feedback over the course of the year.

2. Development of a support program

Developing support meant creating and acquiring documentation as well as teaching materials. We also planned for a guide on how to migrate to Zotero. We coordinated with library technicians for help ensuring that all the library's computers had the new application. Furthermore, we re-considered our internal support processes by forming a small team of people to manage the migration project and collaborate for ongoing future support.

3. Workshops and training

In addition to our regular library workshops, we partnered with other departments in the university to offer workshops on the new tool, spreading the news. We also decided to prepare our colleagues with training.

4. Migration assistance

We initiated a migration assistance campaign to all account holders, whether or not they had used their accounts recently, to ensure that everyone was aware and had the opportunity to migrate. This included migration workshops and individually contacting the heaviest users to meet with us. Because we had already identified that this would be a relatively small group, we were able to budget time to spend with them. Lastly, we began documenting the process for migrating from one tool to the other.

Although post-evaluation, each of these four steps were designed to remain in communication with our stakeholders, encouraging their buy-in.

Conclusion

The model of a process to systematically evaluate digital tools, as detailed in this study, supported and justified a recommendation for a new reference management application. I've applied that process, which was based on past experience in the private sector, to digital scholarship initiatives.

A technique explored in this case study involved identifying stakeholders and eliciting their requirements. As the DS librarian, working in an intermediary role I applied an understanding of those requirements to express the evaluation's priorities. In other, more complex evaluations one would formalize stakeholder interviews, methodically documenting and decomposing business processes (e.g. a student's research workflow) into component tasks. These could then be used for mapping functionality and determining priorities.

Undertaking a well-defined evaluation process enables us to pinpoint crucial areas for engaging stakeholders. As discussed, early stakeholder engagement in the evaluation process reduces the risks of making an unsatisfactory decision. Ultimately, as we continue to explore and experiment with software and technologies in digital scholarship, we want to develop trusting, fruitful arrangements with our research colleagues and other stakeholders.

Bibliography

Achimugu, Philip, Ali Selamat, Roliana Ibrahim, and Mohd Naz'ri Mahrin. "A Systematic Literature Review of Software Requirements Prioritization Research." *Information and Software Technology* 56, no. 6 (June 1, 2014): 568–85. https://doi.org/10.1016/j.infsof.2014.02.001.

Adomavicius, Gediminas, Jesse C. Bockstedt, Alok Gupta, and Robert J. Kauffman. "Technology Roles and Paths of Influence in an Ecosystem Model of Technology Evolution." *Information Technology and Management* 8, no. 2 (June 1, 2007): 185–202. https://doi.org/10.1007/s10799-007-0012-z.

Benestad, H. C., and J. E. Hannay. "Does the Prioritization Technique Affect Stakeholders' Selection of Essential Software Product Features?" In *Proceedings of the 2012 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, 261–70, 2012. https://doi.org/10.1145/2372251.2372300.

Burnay, Corentin, Ivan Jureta, and Stéphane Faulkner. "How Stakeholders' Commitment May Affect the Success of Requirements Elicitation," 336–41, 2015. https://doi.org/10.18293/SEKE2015-105.

Connelly, Jim. "The Sargasso Sea of Records Management Software." *Records Management Quarterly* 30, no. 2 (April 1996): 21.

Coughlan, Jane, Mark Lycett, and Robert D. Macredie. "Communication Issues in Requirements Elicitation: A Content Analysis of Stakeholder Experiences." *Information and Software Technology* 45, no. 8 (June 1, 2003): 525–37. https://doi.org/10.1016/S0950-5849(03)00032-6.

Cox, John. "Communicating New Library Roles to Enable Digital Scholarship: A Review Article." *New Review of Academic Librarianship* 22, no. 2–3 (January 1, 2016): 132–47. https://doi.org/10.1080/13614533.2016.1181665.

Eastham, James, David James Tucker, Sumir Varma, and Scott Matthew Sutton. "PLM Software Selection Model for Project Management Using Hierarchical Decision Modeling With Criteria From PMBOK® Knowledge Areas." *Engineering Management Journal: EMJ; Huntsville* 26, no. 3 (September 2014): 13–24. http://search.proquest.com/business/docview/1561957630/abstract/3168835D00774CDFPQ/1.

Fernandes, J. M., S. P. Rodrigues, and L. A. Costa. "Comparing AHP and ELECTRE I for Prioritizing Software Requirements." In *2015 IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, 1–8, 2015. https://doi.org/10.1109/SNPD.2015.7176282.

Franco, A. Jaramillo. "Requirements Elicitation Approaches: A Systematic Review." In *2015 IEEE 9th International Conference on Research Challenges in Information Science (RCIS)*, 520–21, 2015. https://doi.org/10.1109/RCIS.2015.7128917.

French, Simon, and Dong-Ling Xu. "Comparison Study of Multi-Attribute Decision Analytic Software." *Journal of Multi-Criteria Decision Analysis* 13, no. 2–3 (2005): 65–80. https://doi.org/10.1002/mcda.372.

Glinz, M., and R. J. Wieringa. "Stakeholders in Requirements Engineering." *IEEE Software* 24, no. 2 (March 2007): 18–20. https://doi.org/10.1109/MS.2007.42.

Goh, Dion Hoe-Lian, Alton Chua, Davina Anqi Khoo, Emily Boon-Hui Khoo, Eric Bok-Tong Mak, and Maple Wen-Min Ng. "A Checklist for Evaluating Open Source Digital Library Software."

Online Information Review; Bradford 30, no. 4 (2006): 360–79. http://dx.doi.org/10.1108/14684520610686283.

International Organization for Standardization, and International Electrotechnical Commission, eds. *Systems and Software Engineering: Systems and Software Quality Requirements and Evaluation (SQuaRE)*. First edition, 2011-03-01. International Standard, ISO/IEC 25010. Geneva, Switzerland: ISO, 2011.

Joint, Nicholas. "Evaluating Library Software and Its Fitness for Purpose." *Library Review; Glasgow* 55, no. 7 (2006): 393–402. http://dx.doi.org/10.1108/00242530610682119.

Kelly, Greta. "A Collaborative Process for Evaluating New Educational Technologies." *Campus - Wide Information Systems; Bradford* 25, no. 2 (2008): 105–13. http://dx.doi.org/10.1108/10650740810866594.

Lippincott, Joan K., and Diane Goldenberg-Hart. "Digital Scholarship Centers: Trends & Good Practice." CNI: Coalition for Networked Information, November 20, 2014. https://www.cni.org/events/cni-workshops/digital-scholarship-centers-cni-workshop.

Mohamed, A., G. Ruhe, and A. Eberlein. "COTS Selection: Past, Present, and Future." In *14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'07)*, 103–14, 2007. https://doi.org/10.1109/ECBS.2007.28.

Pazak, Mike, and Judie Beshansky. "Six Steps To Successful CRM-CTI Deployment." *Customer Solutions; Norwalk* 23, no. 11 (May 2005): 46–49. http://search.proquest.com/docview/208163922/abstract/E1D68598657B4F63PQ/1.

Rosendahl, Esa, and Ton Vullinghs. "Performing Initial Risk Assessments in Software Acquisition Projects." In *Software Quality — ECSQ 2002*, edited by Jyrki Kontio and Reidar Conradi, 146–55. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2002.

Saaty, Thomas L. "Decision Making with the Analytic Hierarchy Process." *International Journal of Services Sciences* 1, no. 1 (2008): 83. https://doi.org/10.1504/IJSSCI.2008.017590.

Scheel, Henrik von, Mark von Rosing, Marianne Fonseca, Maria Hove, and Ulrik Foldager. "Phase 1: Process Concept Evolution." In *The Complete Business Process Handbook*, edited by Mark von Rosing, August-Wilhelm Scheer, and Henrik von Scheel, 1–9. Boston: Morgan Kaufmann, 2015. https://doi.org/10.1016/B978-0-12-799959-3.00001-X.

Şen, Ceyda Güngör, Hayri Baraçlı, Selçuk Şen, and Hüseyin Başlıgil. "An Integrated Decision Support System Dealing with Qualitative and Quantitative Objectives for Enterprise Software Selection." *Expert Systems with Applications* 36, no. 3, Part 1 (April 1, 2009): 5272–83. https://doi.org/10.1016/j.eswa.2008.06.070.

Sherer, Susan A. "Purchasing Software Systems: Managing the Risk." *Information & Management* 24, no. 5 (January 1, 1993): 257–66. https://doi.org/10.1016/0378-7206(93)90003-C.