Building a University Information Technology Knowledge Base

Using TiddlyWiki

A Master's Thesis Project

Presented to the

Department of Communications and Information Design

In Partial Fulfillment

of the Requirements for the Master of Science Degree

State University of New York Polytechnic Institute

By Sam Fischer

May 2021
SUNY POLYTECHNIC INSTITUTE
DEPARTMENT OF INFORMATION DESIGN AND TECHNOLOGY
CERTIFICATE OF APPROVAL

Approved and recommended for acceptance as a thesis in partial fulfillment of the requirements for the degree of Master of Science in Information Design and Technology.

June 11, 2021
Date

Kathryn Stam

Dr. Kathryn Stam - Thesis Advisor

06/18/2021
Date

Dr. Ryan Lizardi - Second Reader
ABSTRACT

Keeping a knowledge base up to date can be a difficult and endless task for an Information Technology Services (ITS) department at a university. As technology emerges and is embraced by students and faculty, ITS can become responsible for supporting end users with all sorts of questions and problems. A knowledge base is a website or application used to store helpful information in the form of articles that can be easily shared with end users, giving them a way to support their technology problems without contacting a live person. When managing a large-scale knowledge base, information can be inadvertently replicated in many different areas and contradict other articles, especially when more than one manager can add or edit information. Information modules, or chunks of verified information that can be referenced and inserted into an article, can help eliminate contradictory information and ensure consistency with information that has to be in multiple places through the use of hypertextuality and transclusion. Hypertextuality is a way of organizing information and documents by creating individual bits of information that are assembled together to create a larger document, and transclusion is a way of assembling hypertext documents by simply referencing one into the other. TiddlyWiki is an application that is built around the idea of hypertextuality and transclusion. This project aims to explore the potential for a knowledge base to be built with the TiddlyWiki application.
# Table of Contents

Abstract 3

Figures guide 5

Introduction 6

Literature Review 7

Methodology 14

Project 15

Modules 16

Articles 17

Transclusion 18

Layout 20

Tags 23

Project Link 24

Conclusions and Future Outlook 24

References 26
FIGURES GUIDE

Figure 1: Screenshot of Banner Module 16
Figure 2: Screenshot of instructional article 17
Figure 3: Screenshot of transclusion in an article 19
Figure 4: Screenshot of images displayed in an article 20
Figure 5: A screenshot of the Home page 21
Figure 6: A screenshot of the Accounts article 22
INTRODUCTION

Students and faculty at a university use many pieces of technology throughout their college careers, both college-owned and personal. They might wonder how to check their campus emails, how to reset a forgotten password, or how to create a personal campus website. For these kinds of questions, students and staff contact their Information Technology Services (ITS) department for answers. The ITS department, confronting the mass of questions that students and faculty bring to them, can create an online portal of answers called a knowledge base. The knowledge base is a repository of information that can be shared with end users when they bring up questions that have already been asked.

Normally, when an end user brings up a new question or issue, the ITS team springs to action and works to answer the question or resolve the problem. For commonly asked questions and commonly submitted issues with simple steps to resolution, ITS can create knowledge base articles and share them with new end users as a way to alleviate the direct attention they need to give each question or issue. While some end users may still rely on a live support person, many end users may be able to help themselves simply by being given the right information in a format that they can easily follow.

A knowledge base can quickly fill with articles. If the ITS department isn't careful, articles may quickly become out of date as new technology emerges or as college policies and procedures change. Keeping the knowledge base up to date can become a full-time task. One way to help keep the knowledge base accurate is to use small bits of information called modules across many articles. Modules can then be combined to create a full article. Editing an outdated piece of information on an article involves simply editing the module that contains that
information. If a module is used across many different articles, editing the module will propagate throughout the entire application, saving a lot of time and energy.

The main goal of this project is to demonstrate the utility of information modules when writing knowledge base articles and explore how to organize modules into articles and articles into a knowledge base. This project aims to answer the following questions:

1. How can information modules be used to build consistent knowledge base articles?
2. How can information be organized and displayed on a knowledge base?
3. What are the benefits of using TiddlyWiki as the foundation of the knowledge base?

**Literature Review**

It seems apparent that the more that technology is used in everyday life, the more often people will need to support that technology when it breaks or does not work as expected. The proliferation of consumer and professional technology, and the increasingly complicated nature of that technology, means that people will be more likely to seek professional help when working on issues with their technology. When faced with issues, end users will require help from professional technical support staff who may support an end user's technology either in-person, or through the use of a repository of helpful technical information called a knowledge base.

The main goal of a knowledge base is to create a self-service support center that allows users to find answers to commonly asked questions without utilizing in-person resources. "A knowledge base doesn't simply have to be a reservoir of help center articles. It can also include functions like a frequently-asked questions (FAQ) section, a user forum, articles, white papers, how-to articles, video tutorials, case studies, and dictionaries or glossaries" (Birkett, n.d.). To effectively manage a knowledge base, there are several important elements that must be
considered: optimizing digital interactions, modularity, and tagging. This review will take a closer look at these topics and how they relate to maximizing the benefits of a knowledge base.

There are many times when people want in-person assistance for their technology problems. According to HubSpot Research, "62% of customers want to communicate with companies via email for customer service, 48% want to use the phone, 42% prefer live chat, and 36% like "Contact Us" forms" (Birkett, n.d.). For end users who want in-person assistance, there should be a robust system in place to capture their requests. Conversely, for users who want to find solutions to their problems without a live support system, either by reading knowledge base articles or watching a how-to video, there should also be a simple to use and comprehensive repository for that information. Unfortunately, according to a global survey conducted by Amdocs, "75% of surveyed consumers said they would prefer to use online support if it were reliable, but only 37% currently even try to use self-service options, which are often perceived as inaccurate or incomplete. An overwhelming 91% say they would use a single, online knowledge base if it were available and tailored to their needs" (Amdocs 2012). When knowledge bases are not effectively written, communicated, and maintained, end users will be forced to contact live support, which will consume support resources.

A knowledge base should function as the interactive alternative of a live, in-person support session, but not necessarily as a replacement for one. It should give the end user clear and concise information that is easy to find. Ideally, solving a problem via a knowledge base should be just as conclusive and rewarding as a live interaction. At the same time, it should save considerable time for support staff who are repeatedly answering the same questions from end users. To save the most amount of time in managing the accuracy of knowledge base articles, it is useful to break the articles apart into manageable chunks, or modules, that can be more easily
updated and verified. Articles would no longer be purely static information pages, but more dynamic and hypertextualized.

When trying to manage a knowledge base that contains repetitive information across the entire system, the ability to update a single module which propagates to all articles can save considerable time and resources. For example, if many articles in a knowledge base all contain a reference to a manufacturer's website, and the manufacturer changes their URL, that change would need to be manually updated across the entire knowledge base, article by article. If the manufacturer's website URL was stored as a module and referenced throughout the knowledge base as a form of hypertext, updating a manufacturer's website URL would simply involve updating the one module.

When creating knowledge modules, these modules must be organized and indexed so that they can be easily searched for and used to generate knowledge base articles. The non-linearity of the references to the individual modules allow for dynamic article generation, similar to a wiki. As described by Will Lakeman (2008):

Unlike many other devices for the organization of knowledge, the flexibility of these nonlinear associations establishes the indexing and organizational systems of a wiki as an inseparable part of their content; where devices such as the content pages of a paper encyclopedia are transformed into active projects such as the "navigation links" area of Wikipedia, a catalog of catalogs devoted to a metadiscussion of the ever evolving schema by which users organize its content. (p. 151)
For users who like to search the knowledge base by theme, by utilizing the index of modules as an artifact of the knowledge base to search by instead of a static list of articles like a table of contents, searching the knowledge base will always yield relevant results.

A major benefit of modularizing knowledge is that work can be done on the building and management of the knowledge in a non-sequential process. The technology with the greatest need for new knowledge base articles can be prioritized over less impactful ones. "Tasks are sequenced so that technology and component development tasks with the greatest need for new knowledge and with the greatest impact on other component design and development tasks are undertaken first" (Sanchez & Mahoney, 1996, p. 68). In environments with limited resources, prioritization of tasks is incredibly important, so focusing on the most impactful tasks is important for not over-utilizing resources. "Noncritical functions that are part of the less-important 80% should be minimized or removed altogether from the design. When time and resources are limited, resist efforts to correct and optimize designs beyond the critical 20%, as such efforts yield diminishing returns" (Lidwell et al., 2010, p. 14).

To categorize information that will be included in a knowledge base, it is important to understand what information is and whether it has a hierarchy. According to Alavi and Leidner (2001), at the base of the knowledge hierarchy is data, the raw data and metrics that are used to determine patterns and turned into meaning information. Information, the second tier of the hierarchy, places data in a context in order to have a framework of interpretation, which becomes knowledge. While the transition from information to knowledge is not always clear, the key to distinguishing them is not in the content of the information of knowledge. "Rather, knowledge is information possessed in the mind of individuals: it is personalized information related to facts, procedures, concepts, interpretations, ideas, observations, and judgments" (p. 109). According to
this view, knowledge only exists when it allows someone to increase their ability to solve their own problems. Everyone has a different level of familiarity with technology, a different capacity for learning and at different speeds, and different needs. The goal of the knowledge base is to help end users translate information into knowledge as effectively as possible. Additionally, it should also allow subject matter experts to translate their personal experience and wisdom into digestible information to users at all levels. "Information is converted to knowledge once it is processed in the mind of individuals and knowledge becomes information once it is articulated and presented in the form of text, graphics, words, or other symbolic forms" (p. 109).

Another way of viewing a knowledge base of modularized chunks, each tagged and indexed, is as a hypertext learning environment (HLE). "HLEs are often used for inquiry-based learning, discovery learning, web quests, and so forth; that is in environments where the instruction does not necessarily follow a set order or structure" (Walhout et al., 2015). There are many benefits to organizing data in this way. It allows for users to browse for content thematically and to discover content through the tags. As opposed to traditional printed articles, where the information is fixed in a linear, consecutive order, tags allow for dynamic searching and sorting.

When organizing the knowledge base articles, tags allow for an index and dynamic searching and sorting, the assumption is that the article still has to actually 'live' somewhere like a folder or subfolder.

One essential difference underlies and inspires tool building efforts in support of folders vs. tags. Computer-supported folders follow a "put this there" model that agrees well with the way we organize objects in our physical world: Put the keys on the table; put the tax return in the "tax" folder. This model is simple and familiar, but also limiting.
Information can be organized in only one way. An item of information can be in only one place at any time, even though we may have several ways that we'd like to organize our information. (Hartzler et al., 2009)

Tagging appears to have numerous advantages over folder-based organization. Even though the goal of information modularization is to make information chunks as small as possible, a situation could still and likely will exist where one module of information may have numerous tags implicit within it. Without folders, all information modules would exist in one space and would require the heavy use of searching. Searching for tags and keywords has certainly become a popular phenomenon, like the Google search function, however the categorization of articles and tags is not always straightforward and does not always produce meaningful results right away in the search results. "Although some participants liked having all articles in "one place", enabling a complete scan, they were forced to take explicit steps to achieve control over workflow and clutter comparable to that of folders" (Hartzler et al., 2009).

According to a study by Balcytiene (1999), students who were given a hypertextualized paper to read instead of a conventional paper benefited more if their prior knowledge of the subject was low. "This means that when the hypertext is not too complex, low prior knowledge students do not get lost. In the hypertext system the restricted access to the information provided the low-prior-knowledge students with a global view of the whole complexity of the learning domain, without overloading them with too many details that they could not handle" (Balcytiene, 1999, p. 235). For end users with a low technical background or slow to learn technology, a hypertext-based knowledge base with limited hypertextual features would allow the user to explore the content without getting overloaded by too much content or too many avenues to more content. Users who use little technology would probably not get much out of a content-rich
environment with too many ways to get lost, so for these users, keeping a conventional, linear-based format of knowledge base article with basic hypertext features, for looking up basic terms, may be most advantageous.

Information layout is important for the functionality of a knowledge base, not just in the structure of the knowledge base but in the types of information being presented to users. "Multidimensionality is commonly regarded as a way of enriching traditional static representations and of enhancing knowledge acquisition by providing different points of view in the same conceptual network." (Reimerink et al., 2016, p. 448). Having multiple types of information in the knowledge base - text, image, videos, audio, graphs, etc. - will increase the ability for the article to transfer knowledge effectively to each type of learner. Additionally, providing articles aimed at various levels of learners could be helpful. Different articles could be written for beginner or advanced users to provide each level of learner with appropriate amounts of information and prevent information overload.

According to Ridi (2018), there are multiple characteristics of hypertextuality, including granularity, integrability, and multimediality (p. 394). A document built with hypertext is assembled by connecting individual nodes together. These nodes should be granular bits of information that can be integrated easily into other documents in any order and combination. "Indeed, only if a document can be decomposed into many nodes, it will be possible to connect them in many different ways" (p. 394). The nodes themselves can be anything from text, images, video, audio, or any combination of media. Traversing a hypertext document should allow the user to explore a never-ending document, "without ever arriving to any definitive end (or beginning)" (p. 394). Multimediality refers to the structure of links between nodes and how the overall layout of the structure is visually presented to the user.
These can make the whole structure of a hypertext map-oriented rather than index-oriented, by favouring spatial organization over temporal organization. While the latter is more typical of linear texts based on an ordered sequence, like in lists, when facing an image or a map, the readers can freely choose to pay attention to any of its parts, all simultaneously available to their look. (p. 394)

Ridi (2018) also describes how transclusion is incorporated into the overall functionality of hypertextuality: "In the WWW, there are numerous duplications of the same pages and of their informational contents, which make it difficult to distinguish the versions and to identify their chronology and relationships... [T]hanks to the "transclusion" method, information units are never duplicated, but they are displayed or included wherever they are useful but without compromising the uniqueness and priority of the original source. (p. 405) With transclusion, information can be seen in multiple places through references to the original source; changing the original source will ripple through the entire system wherever it is referenced.

A knowledge base is full of articles of a particular kind of knowledge. The author of any knowledge base article likely has some proximity to the internal systems mentioned in the article. The author of an article has to convert their internal knowledge of the systems mentioned into external knowledge base articles. According to Nonaka et al. (2009), these are two different kinds of knowledge, tacit and explicit:

Knowledge that is uttered, formulated in sentences, and captured in drawings and writing is explicit. Explicit knowledge has a universal character, supporting the capacity to act across contexts. Explicit knowledge is accessible through consciousness. Knowledge tied to the senses, tactile experiences, movement skills, intuition, unarticulated mental models, or implicit rules of thumb is "tacit." Tacit knowledge is rooted in action,
procedures, routines, commitment, ideals, values, and emotions. Tacit knowledge can be accessible through consciousness if it leans towards the explicit side of the continuum. (p. 636)

Nonaka et al. (2009) describe that tacit and explicit knowledge fall along a continuum. Tacit knowledge, on one end of the continuum, is knowledge that cannot be easily communicated as it encompasses, "knowledge for inherent physical functioning to the insights or inspiration needed for an act of creativity" (p. 640). Explicit knowledge is at the other end of the continuum and is knowledge that can be articulated, written down, and shared with others. For a knowledge base to be functional, people with tacit knowledge of issue resolution and internal systems have to convert their tacit, personal knowledge into explicit knowledge base articles in order for knowledge to be shared and for others to make use of that knowledge.

For an ITS department, the knowledge that is used to build the knowledge base is held by technology professionals with what Nugroho et al. (2018) calls professional knowledge. "Professional knowledge is based on faith or trust that knowledge producers have made fair and honest use of the primary evidence and knowledge in their policy advice, that they have a good understanding of the context in which a decision has to be taken, and that they have not manipulated the evidence primarily for their own gain or purpose" (p. 34). The professional knowledge of ITS professionals serves as a bridge between the campus community and the technology that they use every day. ITS has to both generate knowledge through the work that they do and must also translate that work into articles that can be easily understood and used by the community. Some technology can be learned and understood by end users directly, but some technology can only be understood through ITS due to its complexity or industry practices.
Building and organizing a knowledge base, as well as keeping a knowledge base up to date is a difficult and endless task for ITS departments. The knowledge should not be used to replace in-person interactions but allow for end users to find answers to problems that are repeatedly being answered, often in exactly the same way. Tags present the opportunity of alternative methods of organization of information within the knowledge base. This research will explore knowledge base organization and usability.

**Methodology**

Building the knowledge base involves finding or creating the information that will populate the system. As a student at a university, it seemed relevant to use documents and websites of a university ITS department. The SUNY Polytechnic Institute ITS department website was settled on as it already contained a plethora of web content, almost exclusively text based. The organization of the SUNY Polytechnic Institute ITS department website is a simple tree structure on the left side of the screen, with roughly a dozen top-level categories and many subpages in each category. While the navigational tree is visible from most of the subpages, it often gets replaced by a smaller navigational tree when exploring certain categories.

In order to build this project, a platform to build and organize the content has to be chosen. After some previous research, TiddlyWiki was settled on as it seemed best fit for the tasks for this project. TiddlyWiki is, "a unique non-linear notebook for capturing, organising and sharing complex information" (TiddlyWiki, 2020). This project will use the TiddlyWiki application to build a website to use as the knowledge base. TiddlyWiki is built around the concept of hypertextuality and transclusion and has many features that make creating complex and dynamic chunks of information very easy to make and manage. In TiddlyWiki, every chunk
of information is called a tiddler. A tiddler can be anything from a URL, a sentence of text, or an image. A tiddler can also be a collection of other tiddlers. The text of one tiddler can be displayed through another through transclusion. Throughout the project, I will use terms like "modules" and "articles". Modules are small chunks of information that are assembled together to create an article. In terms of TiddlyWiki, both are tiddlers. The articles that I create to build the knowledge base are themselves tiddlers made up of statements to bring other tiddlers of information modules together through transclusion.

This project will begin by choosing four of the most useful top-level categories. The categories chosen are About ITS, Accounts, Email, and Staff, Faculty, and Student (SFS) Websites. These four top categories will provide me with enough ancillary material to build the many information modules eventually become the knowledge base articles. A clear distinction can be made between the questions "what is x?" and "how do I do x?", so the knowledge base will be designed to primarily show end users answers to questions on how to do something like change a password, but definitions and terms should be easily discovered in a glossary. This project will be built using research on hypertextuality from Riccardo Ridi in Hypertext due to the similarity between TiddlyWiki and the concepts of modularity and transclusion that it was built around. Ridi (2018) describes characteristics and components of hypertexts that will be used in building this project.

Project

Modules
The first step in building the knowledge base was to build the individual modules that were eventually combined with many others to create the articles. Lidwell et al. (2003) explain that modular systems, "should be designed to hide their internal complexity and interact with other modules through simple interfaces" (p. 160). Ridi (2018) explains that, "The more a certain amount of information is decomposed into small (as long as they are also self-explanatory) and numerous nodes, the greater will be its granularity, which will allow the reaggregation into a more articulate hypertext" (p. 396). When reviewing the SUNY Polytechnic Institute ITS department website, there were various concepts and software names that appeared throughout many of the instructional how-to articles. These concepts and software definition modules became the first tiddlers in the project. For the sake of scope of this project, thirteen of such modules were created, such as Figure 1 which shows the Banner module.

![Banner Module Screenshot](image)

**Figure 1: Screenshot of Banner Module**

Various other ancillary modules were created such as URLs to external systems and the article footer as seen at the bottom of Figure 1. In Figure 1, the blue Banner button in the description is a separate module called Banner URL including only a link that brings the user to the Banner system in a new browser tab. If the university moves the Banner system to a new
server and the URL changes, only the *Banner URL* module will need to be updated for every article that includes the *Banner URL* module to remain accurate.

**Articles**

Once the concept and software modules were created, the actual instructional help articles could be written. For the purpose of this project, articles were written that had a clear step-by-step process for their resolution. This allowed for the types of articles to be consistent, along with their style and layout. There were many already written on the SUNY Polytechnic Institute ITS department website that were used to create this project.

---

**Figure 2: Screenshot of instructional article**

Figure 2 illustrates a typical article containing the title of the article, an option to show images, the step-by-step guide, and a footer containing a link to the home page. This structure became the template of virtually all of the instructional articles, leveraging aesthetic and internal
consistency. When building the articles, particular attention was given to "use aesthetic consistency to establish unique identities that can be easily recognized," and to "ensure that systems are always internally consistent" (Lidwell et al., 2003, 56).

Transclusion

Transclusion is a key component of TiddlyWiki. In the editor of a tiddler, transclusion is invoked using curly brackets with the title of the tiddler to be transcluded in between the brackets, for example the footer in Figure 2 is invoked as {{Article Footer}}. This mechanism allows for simple editing of an article if the name of a tiddler has changed. This also means that a finished article may be merely a series of transclusion statements. TiddlyWiki handles transclusion especially well, keeping the inner workings of TiddlyWiki a secret to the end user when displaying only one article.

Users are not likely interested in traversing the knowledge base randomly but are looking for direct answers to their questions. Once they have found the appropriate article, they may come across an unknown term or concept due to the technical nature of the content. This is problematic since the user now has two issues; they have been answered a question with another question. This situation is where modularity and transclusion are extremely useful. Figure 3 shows the Banner module of Figure 2 displaying as the text block with the orange background, inline with the rest of the article. By clicking the Banner link in step one of the article, the user can toggle the transclusion on and off. This allows for the end user to answer secondary questions immediately, without having to navigate away from their original question. In the case of needing to change one's password, as in Figure 3, if the end user doesn't know anything about
Banner, simply asking them to navigate to Banner to change their password will not help them. The *Banner* module contains not only a description of Banner but also a link directly to the service, making navigation easy for the end user. This inclusion of modules through transclusion is a part of every article, and often many times throughout.

![Figure 3: Screenshot of transclusion in an article](image)

In addition to the transclusion of modules, articles also contain a feature to display images that accompany each step in the set of instructions. These images augment the transmission of knowledge contained within the article. "Pictures are generally more easily recognized and recalled than words, although memory for pictures and words together is superior to memory for words alone or pictures alone" (Lidwell et al., 2003, p. 184). By combining the images inline with the steps of the instructions, the images are proximally closest to the text and most likely to be used and remembered by end users. The text of the button to show the images
toggles with the display of the images. By default, the images are hidden so the button reads *Show Images*, and changes to read *Text Only* when pressed and the images are displayed. *Figure 4* illustrates the images displayed on the article *Change My Password*. Some users may prefer the concise and visually compact version of pure text, so the button simply gives users the option to display images without forcing it.

![Screenshot of images displayed in an article](image)

**Figure 4:** Screenshot of images displayed in an article

**Layout**

Ridi (2018) explains that, "the main problem of large and complex hypertexts is the extreme ease with which the readers can lose orientation during the navigation, failing both to find what they are looking for and to understand what their position is in relation to the overall structure they are exploring" (p. 400). To help overcome this, the overall layout of the knowledge base in this project is based on the idea of hierarchy. Because of the scope of this project, and the
relatively small number of top-level categories, a tree structure was used to organize the knowledge base. The tree consists of two levels: the top-level organization that includes the main subject categories, and the articles which fall under the subject categories. There are direct parent-child relationships between these levels. "Tree structures illustrate hierarchical relationships by locating child elements below or to the right of parent elements. Tree structures are effective for representing hierarchies of moderate complexity, but can become cumbersome for large or complex hierarchies" (Lidwell et al., 2003, p. 122). The knowledge base is displayed using a certain kind of layout in TiddlyWiki using the Table-of-Contents macro called toc-tabbed-internal-nav. The toc-tabbed-internal-nav is a layout that features two windows: an expandable tree on the left side of the screen, and a space for dynamic content on the right side of the screen. When a link is clicked on the tree on the left or within the content on the right, the content of the target link replaces the content on the right. The main benefit of this layout is that the navigational tree on the left never moves or changes, besides expanding or collapsing the main categories, so the end user never feels like they navigated away from where they started. This layout is visible in Figure 5. The end user can navigate to anywhere on the navigation tree and it will never change or move.

Ridi (2018) describes the types of typologies of hypertexts which includes sequences, grids, trees (hierarchy), and webs. "Sequence, hierarchy and grid can be considered simpler and more predictable types of hypertexts than an irregular and unpredictable structure such as the World Wide Web, that can however include inside also sites or their sections organized, just like sequences, hierarchies or grids. All the possible ways of connection among information units can be placed in some position within the conceptual field of hypertextuality" (p.398). The nature of
the tree layout of the toc-tabbed-internal-nav allows for structure in the knowledge base through the use of tags and presents a simple and predictable style that is simple for users to navigate.

Figure 5: A screenshot of the Home page

There are four main categories in the knowledge base - About ITS, Accounts, Email, and Staff, Faculty, and Student (SFS) Websites. Each of these categories have their own article that gives a brief description, links to the how-to articles, and links to terms that are associated with that category. Figure 6 shows the Accounts article that acts as the landing page of that category.

A user could jump straight into a how-to article via the navigation tree on the left, by using the Home page, or by using the landing page of the four main categories. Providing multiple ways to access the list of available articles gives the end user more chances to find the right information. The Home page and the navigate tree on the left, if fully expanded, provide a full glimpse of the how-to articles in the entire knowledge base, while the landing pages of the
categories provide a filtered list of how-to articles related to that category along with links and relevant terms.

In each how-to article is a footer that contains a link back to the Home page. This was included to ensure that an end user never hit a dead end if they navigate the knowledge base exclusively through the content window on the right. A similar feature was added to the concept and software definition modules; at the bottom each of these modules is a link to the Glossary of Terms. Both can be seen in Figure 3. Since navigation of the knowledge base takes place all in one tiddler, and the user is never actually navigated to another URL, the back and forward buttons on a browser do not function in TiddlyWiki the same as with normal websites, so these footers help prevent dead ends. "When orientation is completely lost or when a new search is desired, it is very useful to have, in each node, an anchor that brings us home. However, the concept of home is not unique, so this anchor could activate a direct link to the main page of the hypertext that we are visiting" (Ridi, 2018, p. 401).

![Figure 6: A screenshot of the Accounts article](image)
Tags

The layout of the knowledge base was based on a Table-of-Contents Macro structure that relies on tags, so each article was given a tag so that the tree structure emerged dynamically. The four main categories in the knowledge base - *About ITS, Accounts, Email*, and *Staff, Faculty, and Student (SFS) Websites*, as well as *Home and Glossary of Terms* - were given the tag *TableOfContents*. The toc-tabbed-internal-nav that makes up the entire knowledge base layout uses the *TableOfContents* tag to form the main structure. Each article was then given a tag of the name of the parent category it was supposed to belong. For example, the article *Change My Password* was given the tag *Accounts*. This places the *Change My Password* article underneath the *Accounts* branch of the navigation tree. This was replicated across the knowledge base to create the entire layout.

To build the landing pages of each of the four main categories, the articles that make up that category have to contain a tag used to distinguish them from articles in a different category. In addition to a tag of the name of the parent category, each article is given an additional tag. The concept and software definition articles also have tags. Each has a *Glossary* tag so that they each appear in the navigation tree on the left. This category is collapsed by default as the information contained in these articles is available through transclusion in the how-to articles but can be displayed if an end user wants to explore.

**Project Link**

The website can be accessed at the following link: [http://people.sunypoly.edu/-fischesa/](http://people.sunypoly.edu/-fischesa/)
CONCLUSIONS AND FUTURE OUTLOOK

TiddlyWiki proved to be a very powerful and customizable platform to build the knowledge base with. This project aimed to create a knowledge base using TiddlyWiki, and the platform performed well throughout this project. To prove the concept in terms of usability, layout, and functionality, focus groups would be beneficial. End users could be given a series of tasks, and monitors could review how well the tasks are accomplished, taking note of any particular benefits and shortcomings.

The decision to organize the knowledge base using the toc-tabbed-internal-nav structure had various limitations. By default, TiddlyWiki displays tiddlers in a story river. Clicking a link to another tiddler opens and displays it underneath the original one. Tiddlers do not automatically close, so clicking on multiple links can create a vertical 'river' of tiddlers. There is an option within TiddlyWiki to have the browser URL update as a new tiddler is opened so each tiddler can be easily shared. For this project, the entire knowledge base is traversed through one single tiddler that uses the Table-of-Contents macro to display the navigation tree on the left and the dynamic window on the right. Since the user navigates through only one tiddler, the browser URL never updates, making sharing non-functional in this way. A future addition to this project could include a mechanism to send a compact version of the how-to article to an email address via a button directly in the article.

TiddlyWiki comes with a built-in search bar, but it is in a place on the site that is hidden as other functions are also available there that the end user should not alter. The search function also provides search results of tiddlers that the end user should not see independently, such as images or the article footer. Additionally, the search function opens new tiddlers in the story river as independent tiddlers. A future addition to this project could be to include a custom search bar...
in the table of contents that can be filtered to only show tiddlers containing a certain tag. For this sake of this project, the search function was hidden, but the ability to text search is a huge component of online research and would be a major improvement to this project.

There were many obstacles to overcome in this project, least of all the technical ones associated with TiddlyWiki. TiddlyWiki is very powerful out of the box, but some of the features used in this project were built by the community. While community development of TiddlyWiki is extremely useful and beneficial to others in the community, they are not all consistent in the way that they handle variables. This can be confusing for developers of a complex system as the developer has to be careful that the formatting is correct for each statement that references a different add-on. The success of additional features will require someone with deep knowledge of the TiddlyWiki application, as well as intimate knowledge of the various third-party plugins that make up the knowledge base.

Video can be a helpful addition to how-to articles. This project provides a text and image-based set of instructions, but the addition of videos may prove helpful for end users. Videos take much more time to produce than images, and can become out of date just as quickly, so priority should be made regarding which videos to make, how often to make updates, and how much information to pack into each video.

While TiddlyWiki is aesthetically functional, it could be made more aesthetically pleasing and consistent with the branding of SUNY Polytechnic Institute. The SUNY Polytechnic Institute ITS department website is a child page of the main college website, so a future project could involve inserting the TiddlyWiki knowledge base into the campus website using an iframe in place of the existing ITS website.
Aside from these various possible improvements, TiddlyWiki has proved to work as a platform with which to build a functioning knowledge base. This project proves that there is a strong potential for further development to include more features and contain many more articles.

REFERENCES


https://doi.org/10.2498/cit.2006.02.07


https://doi.org/10.5771/0943-7444-2018-5-393
