Analysis of a Build vs Buy Approach to Software Acquisition For Student Government

Elisha Madison
The College at Brockport, elisha.madison7@gmail.com

Follow this and additional works at: https://digitalcommons.brockport.edu/honors

Part of the Software Engineering Commons

Repository Citation
https://digitalcommons.brockport.edu/honors/203
Analysis of a Build vs Buy Approach to Software Acquisition For Student Government

A Senior Honors Thesis

Submitted in Partial Fulfillment of the Requirements for Graduation in the Honors College

By
Elisha Madison
Computer Science Major

The College at Brockport
May 18, 2018

Thesis Director: Dr. Sandeep Mitra, Department Chair, Computer Science
Analysis of a Build vs Buy Approach to Software Acquisition For Student Government

Elisha Madison
Department of Computing Sciences
The College at Brockport, Brockport, NY

Faculty Advisor
Sandeep Mitra
The College at Brockport, Brockport, NY

Abstract

“Do we build it or do we buy it?” This is a question that is asked time and time again when acquiring tools for businesses. When this question is asked by student government, a unique set of factors come into play. This project is a case study of the various issues that come into play when a student government organization embarks on the acquisition of a petitioning software system, using which students can put up petitions on various topics. The student body at large should be able to sign these petitions online. Student government should then be able to monitor these petitions and act on them accordingly - i.e, they should be able to delete offensive petitions, recognize petitions that have reached a certain threshold of signatures and convey them to the campus administration, etc. At the start of this project, the variety of open source software tools for this very purpose appeared to indicate that this acquisition should be trivial. As we progressed, we realized that the numerous stakeholders that influence our requirements - from college administration, to campus IT services who will support the software frameworks and provide the server space for deployment - create a unique set of functional and non-functional requirements specific to our campus that make the “buy” option rather challenging to adopt. Furthermore, we discovered that even open source tools can have licensing issues that impose financial constraints, especially if advanced features are to be used. We, therefore, explored the “build” option for a customized tool, and considered a process we should adopt (including deciding on software, server and database infrastructure) to stay within our budget. This research investigates the pros and cons of both approaches, and keeps in mind both technical as well as organizational constraints.
Introduction

To build or to buy? A question that has been asked by businesses and a great deal of other organizations since the concept was ever imagined. When there is a task to be accomplished and the tool is not readily available this question has to be asked. This question has only been asked more and more in the software field in a market that pretty much has a program for everything. If a business was to try to find a word processing software, odds are that they would not go trying to build one from scratch in order to tailor it to their business. It would be much easier to buy some software such as Microsoft Word in order to most effectively use time and money. This may not always be the most effective option and there may be things that the company would have built that would make more sense to use in the long run, but in the grand scheme of things it is cheaper and easier to purchase software that is already built for that specific purpose. A scenario where a company may want to take the build approach would be to develop a very specific software or tool for which there is no plausible substitute within budget. An example of this could be a company-wide customer service outline product: This would create a specific need because all companies are a great deal different policies and ways of formatting customer service, so building their own would make the most sense.

While the two examples that I shared have to do with software acquisition, the build or buy approach to situations has been studied quite intensively because of how often this comes about. A study through the Institute of Electrical and Electronics Engineers [3] examines this process and breaks it down into easier to understand steps when making the decision. There are quite a variety of different steps, but at least from the perspective of the software development angle, the reasons are largely not funding related. According to this study the most impactful
reasons for not building it “in house” are risk transfer, maintenance support, and lack of clarity of precise strategic importance of software. The most important deciding factors for either buying or building are time, cost, quality, in-house expertise, and customization dimensions of the development activity.

An easy way to show some of the most accepted methods to proceed with acquiring software are shown below in Figure 1, which is extracted from a chart provided in [3].

![Software Acquisition Processes](image)

**Figure 1. Software Acquisition Processes**

Figure 1 gives a good breakdown of what happens when either decision is made to build or buy. If the decision is to “build” (aka “make”) then the next decision becomes whether or not to complete the project “in-house” where one would have their own developers/engineers complete the project, or have the project outsourced to another company or contractor in order to complete
the project in whatever capacity it may need to get done in. Then if the “buy” approach is taken, there is a little bit more that would go into the process of deciding whether or not the project would be bought all at once or if the project would be purchased in separate components. This is especially important when one is trying to figure out just what is necessary for their organization at the time.

If the software is purchased as a whole package, as shown in the diagram, then it can either be purchased in the form of CoTS (Commercial off The Shelf) or they could be purchased MoTS (Modifiable off The Shelf). There are many cases in which either or would be more beneficial to those looking for a cut and dried solution such as the previous example of a word processor, which would be following the CoTS model. However, if one is looking for some software that absolutely needs to be vetted through the company and needs to have several changes made to it in order to adhere to either company policy or culture, then that is a circumstance that would require the software to acquired using the MoTS model. There are other varied ways to argue for the use of each of these approaches as one moves forward through the software development phase, but the truth is that there is hardly ever one clear or best answer at the start of the project. Usually, as a project proceeds through development, it will need to be repeatedly analyzed before any firm decisions are made moving forward. [3]

Significant considerations that need to be taken into account the project gets closer and closer to the development stage. Also, it is important to make sure that the right decision is being made rapidly. Therefore, one also has to keep in mind that those two factors are constantly vying for your attention at any given moment ([2]). Figuratively speaking, if you’ve got one person in your ear telling you that you have a strict deadline, and you have another person in your ear
saying that it has to be “close to perfection”, then some cognitive dissonance will be at play and there will have to be a decision made to either have the project launch quickly or launch most effectively. While this conflict is pretty standard amongst all sorts of professions, the implications in the software application realm are important - for example, modern social media exposes every bit of someone’s personal lives - consequently, this makes it quite a bit more interesting and necessary to look at the nature and process of acquiring software with a critical and close lens. To expose every bit of someone’s personal lives makes this quite a bit more interesting and necessary to look at with a critical and close lens lately. Now that the general idea of what we are analyzing has been discussed, it is time to start off with the process of how the software acquisition came to be a discussion in Brockport Student Government (BSG).

**Historical Background**

While the BSG administration under my leadership (I held the position of BSG President from May 13, 2017 - May 12, 2018) is the first to have the execution piece happen with an online petitioning system, this particular case has been a topic of conversation for BSG for approximately four years. By the time I was elected into the position as president, almost all of the requirements and permissions of the college administration to complete and implement this project had been taken care of and agreed upon. It came later that BSG needed to make a decision in regards to the build vs buy methodology. What had been agreed upon was that the Brockport Student Government was going to be able to take existing code from an outside source and use that in order to incorporate their own online petitioning system to use across campus. Given the fact that I was a computer science major, and the supporting fact that the RIT student
government administration had told us that the project would take a maximum of thirty hours, I believed that I would be able to do it as my honors thesis. That turned out to evidently be much harder than anticipated, and BSG could not adopt this process, leading to the alternate process that this thesis revolves around.

This project involving the development of Brockport’s online petitioning system is modelled after the student-built system, Pawprints ([4]), which has been used at Rochester Institute of Technology (RIT) for approximately four years. Given the success of this system and the constant student engagement and feedback that followed the implementation of it, it was something that a great deal of surrounding schools have been trying to implement for quite some time. The Pawprints code has been successfully implemented using the “Buy” methodology in both SUNY Geneseo ([6]) and Nazareth ([5]) colleges. Their process was essentially just changing the styling of the Pawprints web page, but keeping all of the same functionality and having their college’s web team transfer over the code of Pawprints from GitHub in order to execute the system.

We thus see that these two institutions followed the MoTS methodology. This would have been the easiest method of moving forward if all that we needed was a functioning petitioning system and if there were no budgetary concerns. However, given the nature of our requirements, our particular goals regarding sustainability of the website and collaboration with the college and our alumni, and other similar conflicts we faced, this was unable to happen which is when we started to explore the option of the “Build” methodology. At first we tried to avoid this methodology due to the increased upfront cost and the flexibility of the finished product. There are also always worries from college administration that the students will take this online
petitioning system to push a ridiculous or hateful agenda, but given the policies and restrictions in place, situations like these can be easily avoided and we will explore that topic a little later into this study.

Our initial impression as student government was that we were going to be able to have our college’s web team execute a process similar to SUNY Geneseo or Nazareth and have them take the code from GitHub and be able to make it work for the Brockport campus. In some of the more initial meetings in regards to bringing this about, Brockport’s web team was under the impression that it would take no longer than a couple of weeks to execute the project given the existing code. These conversations were being had a year and a half ago. After these initial conversations, BSG was then told that in order to have the project be completed by the college’s web team, it would have to be put on the backburner for another two years. This was something that was deemed unacceptable by last year (2016-2017)’s BSG administration and the decision was made that we at BSG would have to handle it ourselves if we wanted this online petitioning system to be realized and sustained.

This proved to be quite the trying time for everyone. As last year’s BSG administration’s term was coming to a close, a great deal of the hard work and focus that was being devoted to the issue of this online petitioning system was unable to be given. This made it quite a bit harder to “hit the ground running” for my administration - i.e., we saw that we had to move into the role of essentially the coordinator for this software development project. Over the summer of 2017, our current student government leadership had consistent meetings with individuals from different parts of the Library and Information Technology Services (LITS) and Website Development departments of the college and we weren’t given a great deal of support at first. That was when
we happened upon the chair of the department of computing sciences, Sandeep Mitra, who decided to give us a great deal of assistance in connecting some of the pieces that we may not have been able to on our own.

BSG was still under the impression that we would be able to move forward with the implementation of the open source code from the RIT-developed Pawprints system and just connect it to a server and be able to execute it in whatever capacity the College at Brockport allowed us to. We were quite incorrect in thinking that it was going to be as simple as that. With the help of our advisor, we were able to connect with a recent Brockport alumna, Ms. Yangyan Dong, from the College at Brockport’s Website Development team who had seemed to be willing to help us complete this project. At this point, there seemed to be a way to have the petitioning system created through one programmer, but we noted that we had to deal with programming languages and techniques that were completely foreign to me and other student software developers on our campus (these were not covered in our curricula). Still, it seemed as though it would be a simple process moving forward by modifying the Pawprints ([4]) code.

**ECHO: The online petitioning system development process**

After an in depth analysis of everything that the project was truly going to require, it became rather evident that whatever we were going to get done would require more than the thirty hours that the RIT student government administration had led us to believe. Part of this issue was that we did not foresee that we had to handle the nuances of the open source code from MIT used in Pawprints. This software, used directly by Pawprints at RIT, allowed them to be able to use a template for petitioning. Consequently, RIT did not have to worry about coding that
part of the project. This seemed like a great tool for us to use at Brockport and we had every intention of moving forward using it. We then ran into the roadblock of the licensing details the MIT software had attached to it. Something that RIT and in general private institutions do not have to worry about as much as student governments at state schools is funding. This MIT software had some aspects of it that were open-source and free to use, however in the case of our petitioning site, if we were to go with the buy approach, we would have to pay a substantial and recurring fee to MIT to legally use certain components of their code.

This fact and the connections that Ms. Yangyan Dong had to people outside of the college led us to a company formed by recent alums of the College at Brockport known as NuIntegrations. This freelance software engineering group became the focal point for our efforts in actual implementation of Brockport’s student online petitioning system, which, at this point, we entitled ECHO. This was quite the milestone for this project because there was finally a group of individuals who were now going to have a focused goal of helping us complete this online petitioning system in a timely fashion.

After accepting in principle that BSG would outsource the development, most likely to NuIntegrations, the next step was the gathering and documentation of the requirements of the ECHO system. At this point, we were still in the mindset that we could provide our requirements document to NuIntegrations, and they would use the “Buy” methodology – in other words, they would use the source code that was readily available in RIT’s GitHub repository to complete our project. Our initial requirements appeared to be consistent with Pawprints’ functionality, and we set up systems in place to move forward with this strategy. All we needed at this point was server
space. This is where some of the more bureaucratic side of the administrative roles came into play.

As the current BSG administration was going on through the course of the summer of 2017, the conversation of how we would access server space came up quite frequently. If we were to use the services provided by the company Rackspace ([7]) in order to host the ECHO site, this would have caused the price of ECHO, on average, to be two thousand dollars or more a year. BSG felt as though there should be a less expensive way to ensure that we are getting the best quality product for the students whose student activity fee money that we are using to fund this project. After some discussion and consultation with our adviser, BSG set up a meeting with some of the individuals in supervisory roles for the College at Brockport’s LITS, with regard to using some of the same server space that the college already uses through ITEC ([8]), located in Buffalo, NY. ITEC is the service provider of choice for web and related services for many institutions within the SUNY system. However, getting to the point where we could contract out to ITEC also proved to be something that became much more of an issue than we had anticipated. Given the fact that BSG is not technically bonded to the College at Brockport, and acts as a 501-C(3) non-profit organization affiliated with the college, this situation caused some individuals to be hesitant when it comes to being collaborative. This fact led to the lead of the meeting that we were in to give us some ultimatums and ultimately not allow us to use the server space or even the same account that the college has in order to use some separate space on ITEC’s server.

This denial of collaboration gave us a new piece of information to use in our decision as to whether or not we were going to connect to Rackspace ([7]) and transfer and edit and
configure code from RIT. Because the College administration as well as the Web Team suggested moving forward using ITEC instead of Rackspace, my administration decided to move forward using ITEC in order to offset the large cost of more than two thousand dollars a year for a relatively clunky and unoriginal coded website. As our advisor, the point person for NuIntegrations, and members of my administration communicated to the individuals at ITEC, we discovered that they would be unable to support the technologies and frameworks used by the Pawprints source code. After a great deal of time between emails being sent back and forth from ITEC and BSG, the fact became clear that using the “Buy” methodology would require the development team to make considerable modifications to the existing code base, and by the time everything that was necessary was fixed, BSG would ultimately end up paying our contractor (NuIntegrations) quite a bit more.

This piece of information was enough to push my administration to request that NuIntegrations develop their own version of an online petitioning system from scratch using the frameworks and databases that would be compatible with the restrictions ITEC set. Given this decision, we had to take a few weeks to make sure that we were analyzing requirements for the system and we had several meetings with Dr. Mitra to ensure that we were doing so. This happened to take quite a bit more time than we had thought because there was a great deal that we had overlooked as potential threats, or incorrect use of code, that would make our online petitioning system less effective than it may be able to be. There were several meetings with individuals from NuIntegrations in order to make sure that all of the requirements were captured before any of the coding actually started. Our advisor said we should also be very careful in formulating our detailed requirements, and constantly made sure that each side was taking into
account every possible angle that the students and college administration would come to the forefront with.

This process of requirement gathering had us, as a student government, thinking critically about how we actually wanted and needed this program to run as well as what challenges we may have to face doing so. Some requirements that took up quite a bit of time were, not surprisingly, on the administrative side of the website. Making sure that the students weren’t able to post personal attacks on individuals and to make sure that the site was one that inspired others to challenge the status quo, we needed to have policies in place to make sure we could effectively facilitate that atmosphere. A policy that was deemed as a requirement in the previous year was that every petition that was going to be submitted would have had to be approved by both the BSG and a member of college administration from the Office of Enrollment Management and Student Affairs. This issue was another point into the “pro” column for the “build” approach, due to the fact that we would otherwise have had to include code for this process into the Pawprints code available to us all while trying to not disturb anything else in the system.

Another difficult requirement was dealing with the anonymity of the students who were going to be using this system. In Pawprints, something that is a common practice in their system is that anyone who posts a petition is required to make their identity known to those who use the site as the author of the petition. While looking at this part of our own system critically, we decided that although the identity of someone could be discovered by BSG and college administration, it would not be made public for the general body of users of the site if the individual did not want their identity to be known. These important differences allowed us to
have much more customized features in mind as we moved forward with the “In-House Build” methodology.

As we moved forward with the “Build” methodology, we started to realize just how much better of an idea it was to have this be the route we take. Had we not taken to the “Build” approach, we would not have been able to establish such a close relationship with the Website Development team of the college and would not have been able to use systems such as the CLAW, which enables Brockport students to securely sign onto applications with their college provided netId’s (email ids) and passwords. Another opportunity that came out of this approach was that we can now truly make this online petitioning system exactly what we wanted it to be. If one goes to the online petitioning system sites of Nazareth or Geneseo, they will see Pawprints with nothing more than superficial stylings changed. But with ours it is going to be our own unique and effective application that, while still consistent with petitioning systems as embodied by RIT’s Pawprints, will run much more “naturally” for our campus. Another great opportunity that came out of this collaborative process was that we were able to connect with the college’s Website Development team in order to come up with a sustainable way to approach the launch and update of our own Brockport Student Government website.

Given the fact that we were going to be moving forward with building this in our own way, we had to be sure that ITEC was going to be able to adapt to all of the frameworks and technologies being used by our developers. Again, this took much longer than anticipated – there were weeks between emails and follow-ups in regards to questions that may have been easily answered over the phone or in person. ITEC’s hesitation to work with us only caused confusion as to how we were going to be moving forward effectively and we were asking ourselves why
they didn’t want to make a deal such as this. I found that through interacting with the individuals from ITEC, we needed to operate on a more specifics first approach with what exactly they were going to need to deploy as far as framework and database compatibility. Despite many more requirements or permissions needed to give to the people at ITEC, the process to move forward turned almost back to square one when the point person I had been talking to cleared our project and sent us over to their sales division.

Once we were decided on how the online petitioning system was going to be designed and implemented, we then had to come up with a contract in order to lay out specifically who would own what portion of the code to the website and what exactly NuIntegrations would be doing and how they would be executing the development of this project. This required partnership from not only the college administration, but from our own Business Manager and Treasurer to closely analyze and examine just what was being asked and what time lines were going to be followed. After about a month of negotiations as well as question asking and answering, we were able to come up with a contract and this contract was signed before the college winter break of 2017-18.

One distinct disadvantage that came out of the “Build” methodology that was made apparently clear at this point was that it was going to take more time than we would have liked. The contract being signed seemed as though we were going to be able to get things done quite rapidly so long as we were able finalize what we needed in regards to acquisition of server space. As alluded to earlier, the issue of this was complicated by all sorts of bureaucratic conflicts. Given the fact that the college was already in a contract with the server space provider, ITEC, there appeared to be hesitation for them to work closely with BSG. After about four weeks of
capturing details regarding deployment between ITEC and BSG, the contact for the sales representative was shared.

After contact with the sales representative, it was suggested that BSG reach out to the Chief Information Officer. When we did so, he responded with the idea that the college would pay for the additional space on the already existing contract with ITEC, and BSG would pay whatever the difference was. As mentioned previously, there was a conversation during the previous summer that ended with some other administrators deciding that it would be impossible to move forward in this manner – but ultimately, this was the approach we adopted.

Another aspect to add to the list of cons to the “Build” methodology was the frustrations with securing an effective means for developers to access the server space that we had just acquired. In an ideal world, our developers would have access to the server space and in turn the development site at all times and from any computer. However, due to the fact that ITEC has boundaries as to what networks can access the development site, there was no way for us to be able to give full access to our developers outside of the Brockport network. While this did not completely halt productivity, it really has and continues to slow down NuIntegrations and their productivity on bug fixes as well as just development itself. This is just another lesson to be learned as to how one can make sure that these sorts of issues are thought of ahead of time so that it can be handled proactively instead of reactively.

When approaching ITEC about the acquisition of a virtual private network (VPN) there was a firm denial and lack of reasoning behind it. Which, while frustrating, is something that will happen in the development of any project, and this is where the requirement of critical thinking and flexibility comes into play. While there are some security risks that accompany VPN access,
there is a great deal that can be done to mitigate that risk and due to the fact that the developers are in fact a legitimate company. It is rather perplexing as to why ITEC would deny such a request outright. Something that we found was that because we weren’t professionals in requirements capturing, we happened to run into roadblocks because of steps we may not have remembered to take or did not know to take. This is something to take note of for others trying to acquire software. While in our case I will defend that we made the correct decision in the build approach, it is not, in fact, the path of “least resistance”.

Something that also needs to be taken into account constantly that was not always at the top of our priority list, especially dealing with a build methodology, is the importance of web security. While we were able to secure the use of the College at Brockport’s CLAW system that allowed us to make sure that students were the only ones permitted to log in, the VPN, SQL injection, and plenty of other potential security issues (learn more about these attacks at https://blog.sucuri.net/2014/11/most-common-attacks-affecting-todays-websites.html ) need to be checked and accounted for from the beginning stages of requirements capture to the end of development. In our case, we did not have any (to the best of our knowledge) security risks exploited for our system. What is to be taken away from this example is that when building or buying software, it never hurts to be too careful about security. Constant checking in with the college’s web security team, or just ensuring that all security risks are mitigated to the best they can be is an action that is extremely important. ([1]).
**Project Status and Process Analysis**

As of May 4th 2018, the preliminary version of the ECHO site has been released and ready for beta testing moving forward with an expected launch of Fall 2018 semester. Now that the build methodology has been executed, we need to be sure that its follow through is as effective as possible. Just because we now have a functioning site, does not mean that requirements are one hundred percent accurate or there aren’t potential bugs to be fixed. With the buy methodology, this risk that not all requirements would have been captured and executed properly is almost always accounted for through the observation of the software before purchasing. Our build methodology leaves a bit of creative wiggle room for developers, which can be both good and bad.

Something that can and should be done before official launch and execution of any build or buy project should be a set of focus group/s that are put together to test functionality and make sure that the site is as effective for those who are experts in it and those who are using it for the first time. Software for student governments especially should ensure that ease of use is something that comes toward the top of the priority list, because every student should be able to effectively use it. These focus groups would ideally consist of individuals that have a wide variety of technical skills and should have representation of a good variety of the Brockport student body. While this cannot always be achieved, it is an important goal to strive for because these potential bugs that come with a build methodology project are generally going to be found on accident by individuals just using the system.

On a much less technical note some of the final ideas to keep in mind looking forward are buy-in from students and use from students. While I do not have an accurate representation of
what that will look like for this project, if one is to use a build or buy methodology, there are some pros and cons and some best practices that should be used in marketing such software. On each end there are different aspects to play off of in order to garner support for the use of the system.

In the event that the software has used the build methodology, one can play off of the idea that it was “homegrown” and would be something that was built with the help of Brockport students for the benefit of Brockport students. This, partnered with our specific case of having recent alumni work on the project allows us the more added effect that Brockport was incorporated at all sorts of different levels from administrative to current students to future students. This can be a powerful marketing tool for those who would likely more closely identify and want to use a product that was produced by individuals that were within the Brockport community. Aside from this, the build aspect can be used to show that the requirements and the whole design of the project were built around concerns and, while modeled after something else, was tailored specifically for students at the College at Brockport.

In the event that the software has used the buy methodology, there are some other ways that individuals can play on the feelings of constituents to establish buy-in. Something that the buy methodology generally has is the idea that there was some other site or some other company that has used the software before and has had success with it. In order to gain support for the software, specific examples can be used of success stories from other places that allow for students to see the software as legitimate. This can be seen as a downside for the build methodology because although there is almost always great care taken in development, it also generally has not been vetted through other avenues other than the student government project
managers or administrative equivalents. Individuals can create marketing plans around what other institutions have found successful with the same software, while the build methodology relies on continually new and original ideas to push the project forward.

**Conclusion**

If this thesis can help guide student government administrators in the future try to develop plans for software acquisition, then I would consider this thesis a success. This specific collection of pros and cons is specific to Brockport Student Government and is not by any means comprehensive. There are areas in which the build methodology is more effective and there are areas in which the buy methodology is more effective. This critical analysis and collection of general tips and scenarios are given as a guide to make open-minded decisions and prepare for what is to come given the situations that Brockport Student Government had to deal with. The advice and lessons learned through this process are not to push individuals to one methodology or the other, but rather to take them and more effectively navigate software acquisition for a student government.
References


4. Pawprints: (https://pawprints.rit.edu/)

5. Naz Petitions: (https://petitions.naz.edu/)

6. GeneseoSpeaks: (https://speaks.geneseo.edu/)


8. ITEC: (http://www.itec.suny.edu/)