Week 2 - Experience/Case Study in Projects

* Describe your experience with project and project management?
* What types of projects have you worked on?
* What methodology did the project use?
* What were a few lessons learned from your experience?

Describe your experience with projects and project management?

What types of projects have you worked on?

What methodology did the project use?

What were a few lessons learned from your experience?

I worked on one major project for 11 years during my tenure at a large Fortune 500 company. I was hired as a Reporting Analyst and my job was compiling 3rd Party Advertising Reports for Rich Media. The process at the time when I started was very manual. I had to log in to 3rd Party Ad Serving web analytic websites to get ad reporting data that was recorded on these external companies’ websites for the ad impressions that they served to my company’s websites. For example, a user browsing our .com might see advertisement for a new Nintendo game and the code that displayed the advertisement would be housed on a 3rd Party Server like a Google Ads Server.

Well, the 3rd Party Server would record the metrics for each time it was served to a user, and our internal systems would also record the same ad serving metrics. The two systems never counted the same exact counts exactly the same, and so the reports I compiled relied on a variance calculation that needed to be monitored to stay within a certain acceptable percentage range. I compiled the reports manually by copying and pasting data from the 3rd Parties into an Excel template for each advertising campaign.

After a few months of manually building the reports, a coworker and I found out how to record Macros in Excel and how to subsequently view and edit the code that was generated by recording the Excel macros. Leveraging our new found skills, we started to work together to automate many of the repetitive tasks of building the reports and eventually, over the course of many years, went from being Reporting Analysts responsible for manually compiling reports to de facto Developers responsible for a web application that we designed, built, and maintained that replaced the old Excel reports with a web-based reporting UI.

Being on the business side of the organization, and being that we were both self­-taught, we remained a small self-supported team lacking support from traditional engineering teams that worked on other floors. In a way, we naturally worked in an Agile like manner, we self-organized and self-directed. And for about the first 5 years this worked very well for us and the product we built. However, as we learned new tools and technologies, and as the business needs constantly changed, our web application grew without proper guidance and eventually this led to the application metastasizing into a wholly unorganized mess of multiple copy-paste functions, ETL processes, and many tightly-coupled chains of data transitions.

At many points I tried introduce ways adding more organization and process improvements to our homegrown methods, but unfortunately as the work load grew, the maintenance increased, and the bugs piled up, we became diametrically opposed in our approaches to how we should manage the application. My coworker was obsessed with pleasing our internal business segment customers with extremely quick turnarounds, yet had no concept whatsoever of the old idiom “if you fail to plan, you plan to fail”, something that had been deeply ingrained into me by learning the meaning of that idiom long before I first heard about it.

Because at some point over the course of time, my work drifted towards DevOps as I took on the responsibility of managing each migration of servers as we went from our web app being hosted on PC under a desk, to a used rack server in the IT closet, to cloud based IaaS server, as well as managing our transitions from MS Access to SQL Server, and multiple UI redesigns. By managing so many changes without formal training of any kind or experience with any PM methodologies, the mistakes of our process of constant change without foresight I grew to deeply understand how important it is to have at least a minimum amount change/process management an that it’s extremely important to slow down and set expectations so that there is time to consider the effects of making changes or adding features.

Near the end of my time there, there was a concerted effort by management to corral our process by introducing us to the Agile Method, but by this time my coworker and I had grown adverse to each other’s working styles. And to add salt to the problem, I had developed anxiety due to the fact that I had to manage more and more of the issues related to standardizing and refactoring code to allow for easier changes to the code. The business rules of our customers constantly changed as they added new vendors (each with their own complex agreements to adhere to)... yet my suggestions at even the slightest process improvements were met with hostile objection and sometimes ridicule.

When we did end up trying the Agile Method, but it was really disorganized. We started using a tool called Jira, and there we created our backlog. However, my coworker was ever more reclusive, excluding me from any discussions he had with our customers and siloing himself to develop quick wins on his own, essentially leaving me to be siloed fixing bugs and doing extensive refactoring poorly planned out changes. We used a Kanban board in Jira to manage our work, but we ended up largely working in two separate waterfall silos.

My manager was really smart, but he was promoted to Director and a manager was never hired to fill his role. Even before that, he was really focused on bigger things and largely left us to fight amongst ourselves. At one point, not sure if it had anything to do with my constant suggesting that we needed more structure around our processes, but we finally had a real Project Manager that was hired to help manage projects for the entire department. She was a godsend! There was finally someone who could understand the pain that our application and its development and management was in. Progress was beginning to be made in the direction of cleaning up our processes.

Then disaster struck. I nearly blacked out while driving to work, I pulled over and tried to get help because something was becoming very wrong, yet I couldn’t speak. I was less coherent than President Biden in his worst of gaffs... I had a full-on anxiety attack. It turns out that constantly being wrong when you know you are right is not healthy. So, I decided to take a break from work and took a one month Leave of Absence.

When I returned, I found out my hostile coworker had quit, the PM was fired, and a Technical PM (who was a belligerent micromanager) was brought in manage the mess. To make matters worse, the Database Engineer who ended up writing the complex SQL Stored Procedures and SSIS Packages at my ex-coworkers’ behest, had put in his 30-day notice, also while I was away. The Database was another area I was routinely left out when I came to planning, yet expected to help improve and maintain.

The Technical PM turned out to be a real mistake, one of the reasons the DB Engineer decided to leave for Amazon. I won’t go into that, but remember I described him as a belligerent micromanager. I was then left with an application to manage, that had multiple copy-paste-tweaked versions of everything. We had three business segments that we managed, and each segment had two UI sections, each UI section had multiple pages, and then there was three ETL processes (with its own sub-processes), and each with a massive collection of SQL Stored Procedures. And, so you really understand how bad this was, each business segment’s artifacts were created under the direction of my ex-coworker without my input. Each with its own disorganized haphazardly cobbled together collection of bugs in a waterfall fashion in a silo... then for the other two business segments, everything (UI, Backend, SQL, ETLs, etc.) was copied and pasted, bugs and all... then tweaked until it worked... until something, somewhere, didn’t work.

In the end, I inherited the whole project to manage and develop on my own. Needless to say, it was too much to handle and I was long tired of not having my voice heard. But I was loyal to the quality of my own efforts, and really loved the company, so I stayed long enough to convince leadership that the Project Managers, Technical or otherwise, would be beneficial and wise to have, but would be pointless to have with only one self-taught developer trying to manage everything. At this point in the game, no amount of project management would be able to keep the sinking product afloat, no matter how good they might be. And so finally, an I convinced management to hire experienced Software Engineer. The expectation was for me to have the new developer job shadow me for three whole months, while I fixed bugs, and tried to make changes to business logic to reflect new advertising agreements.

It was clear to me that managements’ plan on three-month job shadowing was not going to work. So, I continued to make my case that was not the right course of action, while simultaneously explaining the ins & outs and all the ugliness of the whole thing, and also while gauging the expertise and experience of the temporary developer. After about a month I determined the new developer was sufficiently experienced and nuanced enough in the overall big picture problems we faced, so I directed him to rebuild one ETL process that would feed some new SQL tables and ultimately route all the necessary data to one of the APIs. The data handling, I had him rebuild from the source to the API that feed the existing UI page that had the major problems was fixed within about two-weeks.

Fixing this one issue alone, if trying to modify the existing backend rather than building a smaller fully functioning piece like we did could have taken two weeks just to analyze which “Jing Brick” we would need to pull out and fix to keep the tightly-coupled system from crashing down. This ended up being the best way to go. And after we did this change, specifically the ETL and new SQL tables, we had the foundation that would be able to eliminate possibly 60% of our data issues, and I’m giving a modest guess. This is because we decided we could have one ETL and one set of SQL tables that would be loosely-coupled, and that the one ETL would handle ALL the necessary data transitions before being loaded into the database. The old backend system still relied on a massive amount of copy-pasted-tweaked SQL Stored Procedures to run business logic to transform data before feeding the APIs... and even still, some sections of the UI also had embedded business logic.

After finally selling my leadership that my understanding was right, and the rebuild of the entire system, piece by working piece, starting with the backend was the best option, I put in my notice to quit. It was perfect too, as North Seattle College has so many required classes that are only offered right in the middle of the work day for the AAS-T program. I feel this experience has taught me a lot about being a good developer, communicator, and even a little about Project Management. I hope to learn a lot more about Project Management throughout this quarter.