Exodus Project - Pigs Really Do Fly!

A detailed account of the Ohio Department of Public Safety's journey to decommission and replace its mainframe technology.

Ohio Department of Public Safety; Information Technology Office

August 31, 2012

Table of Contents

Introduction 2

Exodus Overview 2

Project Drivers 3

The Great Debate 4

Proof of Concept 5

Project Approach 6

Project Staffing 9

Chicken or the Egg 10

Are We There Yet? 11

The Valley of Despair 12

Pigs Really Do Fly 12

Lessons Learned 14

Conclusion 17

Introduction

The Ohio Department of Public Safety (Public Safety) provides protection of the public through education, prevention, technology and enforcement activities. Public Safety is an umbrella organization with eight divisions supported by approximately 4,000 state employees and 1,500 contract employees. In the early 1970's, Public Safety commissioned its first mainframe to support operations across the state. Over the years, the mainframe hardware was upgraded several times, with the latest model being a Unisys Clearpath Dorado.

In 2007, Public Safety began the journey to retire its mainframe technology and replace it with a Windows-based server environment. The project was initiated because of the high cost of running the mainframe, our desire to support the State's effort to save public funds, and the diminished availability of resources to support the code base running on the mainframe. The project took five years and an estimated 84,500 person-hours to complete.

This document provides a detailed account of Public Safety's journey and while it does address some high level technical issues encountered, it is not meant to be a technical white paper.

Exodus Overview

"Project Exodus" is the name of the project that Public Safety chose to decommission the agency's Unisys Clearpath Dorado mainframe and migrate the code base to a Windows-based server environment.

The mainframe technology has been a staple at Public Safety since the early 1970's when the first Unisys mainframe was installed. Over the past 40 years, this technology became the backbone for the processing and maintenance of driver license, state identification card and vehicle registration information for the citizens of Ohio. The Public Safety mainframe was also vital to public safety as it provided information to law enforcement, traffic bureaus and courts throughout Ohio in a real-time or near real-time basis.

In total, there were more than 2,000 programs running on the mainframe representing more than two million lines of code. Approximately 75% of the code base was written in Pacbase, an IBM CASE tool. Another 20% of the code base was written in a Unisys proprietary language called Enterprise Application Environment (EAE). Both of these tools generate COBOL, which is compiled and executed on the mainframe. Pacbase and EAE-generated programs are accessed by the end users through either traditional mainframe "green screens" or web-based screens. The remaining 5% of the code base was traditional COBOL, with some programs dating back to the mid-1970's.

Data on the mainframe was accessed through a variety of online screens by the 1,500+ Bureau of Motor Vehicle (BMV) and Deputy Registrar end users. These users manually queried the data, inserted new information and modified existing information tens-of-thousands of times per day. An additional 20,000+ law enforcement officers, hundreds of insurance companies, Ohio courts, and the credit bureaus also accessed this data on daily basis.

Data was also entered into the mainframe either real-time (on demand) or through a batch process from over 2,500 contributing entities (e.g., courts, insurance companies, other state agencies). This data was then exported on a nightly basis to approximately 2,500 external stakeholders and customers. There were also hundreds of reports that were generated real-time or delivered the following day through batch processes.

Additionally, on an annual basis, the Ohio BMV generates and mails over 30 million letters for items such as vehicle registration renewal notices, suspension notifications, dealer letters, etc. The logic, data and letter templates were processed by the mainframe.

Finally, there were approximately 400 jobs that made up the mainframe batch schedule and executed from 6pm to 6am every day of the year.

The goal of Project Exodus was to migrate all these items from the mainframe to a Windows-based server environment and decommission the mainframe technology. This led to the identification of the following critical project success factors:

1. The Unisys OS/2200 mainframe being completely powered down and removed from the data center floor.

2. The functionality of the mainframe being available in the new Windows- based server environment with appropriate system response times.

Project Drivers

The Public Safety Unisys mainframe had proven itself to be dependable, highly available, secure, and relatively efficient from a performance perspective. Plus, with more than 2,000 programs running on it, hundreds of integration points, over 30 million letters being successfully generated annually, and a massive batch process, there was no compelling operational deficiency or problem to force a move from the mainframe.

The reality of the situation is that budgets are extremely tight for both government agencies and private-sector companies with Public Safety being no exception. Public Safety's mainframe licensing and maintenance costs from January 2007 – March 2012 were approximately $8.5 million dollars. The majority of this cost was to license the processing power, known as MIPS (millions of instructions per second), to operate the mainframe. The more transactions processed on the mainframe, the more MIPS used. In theory, MIPS were an endless commodity with little tangible expense to the provider, but with a high cost to the customer.

Additionally, the agency's Unisys mainframe was at the end of life and had to be replaced by the end of 2011 which represented the conclusion of the existing five-year contract. (Public Safety was able to negotiate one contract extension with Unisys through March 2012). The cost to replace the mainframe with a newer model was estimated to be $2,000,000 - $5,000,000.

Thus, the total cost to Ohio's tax payers was estimated to be $10,500,000 - $13,500,000 to continue to run the mainframe for Public Safety from 2012 - 2016. This was a large expense and to support the state's effort to save public funds, the agency believed that the same service could be delivered to the public at a much lower cost.

Another project driver was the increasing age of the workforce that was familiar with the mainframe and Pacbase code. The majority of Public Safety employees who have been working with these technologies have been doing so for more than 20 years and are nearing retirement.

Furthermore, Public Safety recognized that the majority of incoming employees have never heard of Pacbase and had no desire to learn it because the technology is old, antiquated, and has a very limited customer base. Pacbase is not being taught in colleges or technical learning centers, and training for the technology is not available in the open market.

The end result is a dwindling workforce with no pipeline to replace the employees as they retire. In five to ten years, Public Safety could be left with more than two million of lines of code running on a mission critical platform with very limited options to support it. There is always the option to bring in consultants, but there are few people in the market that know Pacbase and the administrators that work with Unisys mainframes are very expensive.

Unless Public Safety acted in 2007 to address these issues, five years before the mainframe contract was to expire, the existing mole-hill of issues would certainly turn into a mountain. By migrating off of the mainframe and onto a Windows-based server environment, the way has been paved for the Pacbase code to be rewritten into a modern technology such as .NET.

The Great Debate

In 2007, Public Safety made the decision to decommission and replace the mainframe technology before the end of 2011. A project team was then formed to answer the question: How does an organization retire a technology that has been in place for about 40 years and has become the backbone of processing for the agency?

The first step was to not reinvent the wheel as Public Safety was surely not the first organization, either public or private, to execute a project to retire their Unisys mainframe technology. Thus, Public Safety looked externally for case studies and examples of organizations that traveled down this road previously. What we found was unexpected and caused many to become alarmed.

The fact is that very little information is available about organizations of similar size to Public Safety who migrated and decommissioned their mainframes. It seemed that most organizations that were able to successfully decommission their mainframe were much smaller than Public Safety or paid a consultant organization tens of millions of dollars for assistance. And, of those that used consultants, most of their stories were ones of failure and regret. Thus, the "great debate" began…so now what?

After much discussion and debate, the team proposed the following three options:

1. Make no changes to the code base and instead generate and compile it to execute in a Window-based server environment. There were known compilers available; but, would they work and what architecture would be required to run over 2,000 programs written in Pacbase and EAE in a Windows environment?

2. Reengineer and rewrite all programs in .NET. While this may take much longer, as it requires the team to document, code and test the hundreds of thousands of business rules while also reengineering business processes, ultimately, Public Safety knew that this approach would work.

3. Stay on the existing platform and find a way to fund it. The mainframe was the backbone of the business and whatever it costs to run it was an expense that must be paid. The costs of migrating off of the mainframe was overwhelming and based on the case studies available in the industry, the chances of success were small. Additionally, there was no way the project could be completed in less than five years. One team member equated this mission to a car traveling at 55 MPH while trying to change four tires all at the same time. Another team member believed the task to be so overwhelming and impossible that 'pigs would fly' before this project was completed.

The team spent several months debating the merit of each option, meeting with the Public Safety divisions and the Fiscal Department, contacting external entities such as Gartner and other government agencies, and researching additional options. Ultimately, Public Safety opted for option 1. The existing Pacbase code would be generated as COBOL (no change) and compiled with a Windows COBOL compiler for execution in a Windows-based server environment. It also required the EAE code to be migrated to Unisys' new Agile Business Suite (ABS) which generates C# for execution in a Windows environment.

The team determined that rewriting the code and reengineering the business processes (option 2), would cost too much, would take too long and carried too much risk.

Option 3 was not feasible as Public Safety could no longer afford the high price to lease MIPS to run the mainframe. Additionally, option 3 would require Public Safety to purchase a new mainframe.

Proof of Concept

Before any work could begin to migrate the code base off of the mainframe, a full inventory of all of assets had to be collected and the basic premise of moving the code base off the mainframe had to be validated.

While a full inventory of the Pacbase and EAE code was available in the code repository, there was no inventory of the remaining 5% of the code which was mostly written in COBOL. The only way to find this code was to either ask several of the more senior programmers or stumble across it while performing other functions (which was usually the case and continually created more work for the team). Furthermore, of all the programs identified, there were about 50 which were actively running in production, but no one knew what they did.

Additionally, while a direction was set as to how the code would be migrated off of the mainframe, there was considerable doubt as to whether the compiled code would run in a Windows-based environment and if the architecture could be developed to match the efficiency and reliability of the mainframe. After all, if it was so easy, why are there so few success stories? Thus, we initiated the following proof of concept:

1. Determine requirements for generating Pacbase code in a Windows-based server environment;

2. Determine the requirements for migrating EAE code to Unisys' Agile Business Suite product;

3. Capture volume and nature (batch vs. online, inquiry vs. update) of each type of transaction;

4. Determine the database sizing counts (table and rows) and logging requirements;

5. Determine all other metrics needed to size the new environment; and

6. Execute the code to determine if the approach would work.

Once the proof of concept was completed and the necessary questions had been addressed, the team was ready to begin the work of converting the Pacbase programs to code that would run on Windows-based servers, migrate the EAE code to the Agile Business Suite and rewrite the COBOL code into .NET.

Project Approach

Given the diverse types and large number of programs running on the mainframe, moving all functionality into a new production environment at the same time would not make sense from a risk perspective and would be unfeasible in terms of managing the development effort. The ultimate goal was to migrate all functionality off of the mainframe as soon as possible with minimal risk and as little disruption to Public Safety operations and the citizens of Ohio as possible.

Based on our research, we found many documented instances where other businesses took more than a week to complete their migrations and then experienced significant issues after

going live. We understood that the Public Safety divisions and the citizens of Ohio would have no appetite for a long shut down of services followed by major issues after implementation. In fact, based on the length of the shut down and the magnitude of issues that followed, even if the team was successful migrating off of the mainframe, the project could be considered a failure based on the impact to the customers.

To mitigate much of this risk and minimize the system down time during go live, the team divided the work into the following four distinct parts:

1. Compile and generate the Pacbase code responsible for the Ohio BMV Driver License system;
2. Compile and generate the Pacbase code responsible for the Ohio BMV Vehicle Registration system;
3. Migrate each application written in EAE to Unisys' Agile Business Suite; and
4. Rewrite the COBOL-based applications using Microsoft's .NET technology.

Each component was then analyzed individually to determine how it could be broken into smaller pieces to minimize risk and if possible, migrated into production in smaller units to reduce system downtime. For example, to migrate the Ohio BMV Driver License and Vehicle Registration system, the team developed the following three step plan:

1. Migrate the Vehicle Registration database from hierarchical to relational;
2. Web enable the front-end "green screens;" and
3. Convert the Pacbase code to run on Windows-based servers.

Each component was executed separately and moved to production individually. This helped to simplify the project, reduce risk and minimize system downtime during the moves to production.

The plan for the EAE systems and COBOL programs was to convert or rewrite each system individually as most were stand-alone applications. The team reviewed the inventory of applications and developed a plan to migrate, re-platform or re-code the applications in a sequence that would minimize risk. This sequence was based on a number of factors such as the importance of the application to the customer, the number of customers impacted, the level of integration with other applications, the impact to the customer if the application was unavailable and the complexity of the application.

This approach proved to be on target for Public Safety as it provided the opportunity to first move low volume and low impact non-Pacbase applications into production and correct issues on a small scale that would have been catastrophic if found on a widely used, high volume application. This was especially valuable for the migration of EAE to ABS as the team was able to find and resolve a many of the issues that were unique to our infrastructure. The

applications written in COBOL were all rewritten or the functionality was absorbed by existing systems.

The only downside to this approach was some migrated ABS applications had to be addressed twice if they integrated with existing Pacbase applications. The first time they were migrated they had to be set up to integrate with the existing Pacbase code running on the mainframe. Later, when the Pacbase code was migrated to Windows-based servers, another change had to be made to allow for proper integration in this new environment.

Project Staffing

The project was mostly staffed with Public Safety employees from IT development, infrastructure, database, networking and project management. When the project was initiated, approximately six to eight team members would meet weekly to discuss various technical issues and the high level architecture. Initially, each team member spent approximately four to eight hours a week working on the project. We also hired a full-time contractor to assist in designing the technical architecture of the solution.

Once the technical direction was set, these same six to eight team members worked on the proof of concept and also began to put together the designs for the infrastructure.

After we validated the proof of concept and finalized the project approach, we assigned additional resources to complete each sub-project. However, because of maintenance responsibilities and an internal initiative to train the existing Pacbase resources in .NET, Public Safety team members could not be allocated full-time to the project. As a result, three full-time Pacbase contractors who served in staff augmentation roles were secured to help convert the massive amount of programs and reports.

The chart below identifies the staffing plan throughout the entirety of the project:

Activity	Time Frame	# of Team Members	Total Hours
Initiation; Determine technical direction	June 2007 - March 2008	6 – 8 Public Safety employees at 4 – 8 hours per week	1,500
Proof of concept; Architecture Design	April 2008 – December 2008	6 – 8 Public Safety employees at 4 – 8 hours per week; 1 full-time contractor	3,000
Pacbase: Driver License System	January 2009 – December 2011	10 Public Safety employees at 4 - 28 hours per week; 2.5 full-time contractors	30,000
Pacbase: Vehicle Registration System	January 2009 – February 2012	10 Public Safety employees at 4 - 28 hours per week; 1 full - time contractor	30,000
EAE to ABS migration	January 2009 – February 2012	8 Public Safety employees at 4 - 36 hours per week (includes building EAE functionality in other Public Safety systems and the complete rewrite of others in C#); .5 full-time contractors	18,000
Replacement of COBOL code	January 2009 – December 2011	2 Public Safety employees at 4 – 20 hours per week	2,000
TOTAL	**June 2007 – March 2012**	**PROJECT EXODUS**	**84,500**

Chicken or the Egg

Several advantages of running a mainframe are the processing power, efficiency and overall availability of the environment. The challenge for Public Safety was to create a Windows-based environment that met or exceeded the standards that were established by the mainframe. This was a tall order given that mainframes were designed exactly for the type of environment running at Public Safety: high volume, high demand, and large batch processing.

The root question was "how many Window-based servers does it take to replace a mainframe?" Unfortunately, while there was some research available on this topic, much of it only applied to IBM mainframes and given the uniqueness of each organization's processing needs, it was a bit like comparing apples to oranges. However, despite these challenges, the team was able to develop a good estimate of the number of physical servers that needed to be purchased to create a Production, Quality Assurance (QA), Test and Development environment for both Pacbase and ABS. In total, 18 Unisys ES7600's were purchased.

When it was time to build the environments, the team struggled with whether virtual machines or physical hardware should be used. Ideally, the team would have liked to have used virtual machines where possible, but they also had to create an environment that was highly available and could withstand the assault of tens of thousands of transactions per minute coming from both internal and external sources. There were also several unique services that had to be emulated in the new environment such as a name search that is very specialized to the Bureau of Motor Vehicles (BMV) processing, real-time access to highly sensitive information by law enforcement and connectivity between systems running in the new Windows environments.

As the project proceeded, the topic of technical infrastructure and environment availability became a bit of a sore spot. The developers were requesting a properly architected environment where they could work out the migration and performance issues. However, the infrastructure team wanted to understand how the converted programs would react in a Windows based environment before spending weeks or months building the environments. Both teams needed something from each other before they could proceed…so, what came first, the chicken or the egg?

As it turned out, neither came first. Instead, it was a flying pig. The team realized the impossibility of the situation and the schedule for providing each environment was adjusted. The original goal of creating a solid QA environment several years prior to going live was adjusted to allow for significantly more time in Test. This provided an opportunity for the development team to determine the areas where performance would suffer under higher volume as well as to discover the technical challenges of migrating from a mainframe to a Windows-based server environment. The change also allowed the infrastructure team to work through these technical challenges and make adjustments that were later reflected in the QA and Production environments.

In the end, Public Safety created six Windows-based server environments that met or exceeded the processing power, availability and responsiveness of the mainframe at a cost significantly less than buying a new mainframe. This challenge was originally viewed as an impossible hurdle when the project began. This was an important step as the team began to see the pig take flight.

Are We There Yet?

Project management books and methodology are clear when it comes to developing a project timeline, especially when the common Solution Delivery Lifecycle (SDLC) or Waterfall methodology is used. After the project is scoped and the work breakdown structure is developed, each segment of work is divided into activities. Those activities are sequenced and then durations are assigned based on a number of different estimating techniques. From there, a project schedule or timeline is generated. Ideally, the schedule that is generated fits within the external constraints of the project.

In this case, while the list of known work could be determined based on application inventories and the previously defined steps to move the code off of the mainframe, the team "did not know what they did not know." Experience told the team that there would be plenty of "undetermined" work to be completed. At question was what amount of contingency time should be reflected in the project schedule.

Furthermore, it was nearly impossible to estimate times for each task, as no one knew how long it would take to convert the programs and what testing issues the team would encounter. While Public Safety had some baseline estimates from other sources, they did not provide much assistance.

After the estimation process, Public Safety developed a schedule that seemed to be appropriate. In the end, it was not even be in the ballpark. Furthermore, this new schedule failed to take into account that the number of leased MIPS available to Public Safety was finite.

At the rate ODPS was burning through the leased MIPS, they would be expended before the project was completed. This was eventually mitigated by purchasing more MIPS, albeit at a premium price. However, one external constraint that could not be mitigated was the contract end date with Unisys. Per contract, Public Safety knew that the mainframe had to be retired by March 31, 2012, as Unisys would no longer permit the mainframe to be operated, regardless of the number of MIPS that were purchased or maintenance contract extensions that were signed.

So, after several years of measuring actual work versus the estimated work, determining that our estimates were too low and then re-baselining, Public Safety took a different approach. Public Safety looked at the end date and worked backwards. Essentially "drop-dead" dates were established with clearly defined success criteria. Management and the development staff worked together to measure the progress of assigned work and prioritize the remaining work and issues to ensure the established dates were met. This often meant that the most optimal

method to resolve an issue was not selected in lieu of a short-to-mid-term solution that would suffice until Phase 2 of the project could be executed (Phase 2 is a future rewrite of the Pacbase code into a modern technology such as .NET).

The Valley of Despair

It is not uncommon for large and complex projects with a timeline that is measured in multiple years to fall into a "valley of despair" and the Exodus Project was no exception. The team spent so much time working on the project while making limited progress on a what seemed to be an endless amount of work, that there did not seem to be an end in sight.

Further complicating the effort was the fact that the existing programs running on the mainframe required a significant amount of maintenance and support to handle business unit ad-hoc reporting requests, data fixes, user support, requests from external entities, system issues, etc. The Ohio Legislature also continued to pass legislation that required significant changes to the existing code base. As a result, team members were continually being pulled off of the project to handle these items as the business needed to continue to operate (thus the analogy of trying to change four tires of a car while moving at 55 mph).

In addition, the length of time it took to determine a direction, execute a proof of concept, train the Pacbase resources in .NET and determine the architecture for the new environment also affected the project. Once these issues were resolved, other critical issues would take their place such as code that was successfully tested six months prior would no longer work because an architecture change was made or the networking to support the new solution was not adequate. There were so many unknowns that the team struggled to complete assigned tasks. Plus, with a timeline that was originally estimated to be three years, there was no sense of urgency.

As a result of these factors, especially in the early years of the project, it seemed that completing the project was not possible and Public Safety was making the same mistakes as many other organizations that failed to complete their migrations. In fact, it was during this time, that the Exodus mascot of the "pig with wings" was born and began to appear throughout the ranks of the development staff.

Pigs Really Do Fly

Despite all the issues they encountered, the team was on track to complete the project before the mainframe contract expired. The decision to divide the work into four distinct groups and assign separate resources to each initiative, thus creating four sub-projects, and then executing each sub-project in small individual phases, proved to be a major key to success. This approach brought the team out of the "valley of despair" and provided opportunities for wins and for the team to see and experience success.

Another key factor to the team's success was simply gaining experience in working with the migration tools and learning the idiosyncrasies of making the existing programs run in the new Windows environment. Issues that seemed insurmountable and originally led the team to

believe the solution would not work or could not be completed by the deadline date were being quickly resolved and the experience was being shared throughout the team. One team member commented it was a bit like learning a new programming language with your eyes closed.

In total, the migration strategy included approximately 25 individual moves to production spread over a two-year time frame with the largest two, most critical migrations, occurring in the last three months of the project. This is where the proverbial "rubber hit the road" and success or failure of the project would ultimately be determined. In short, would the pig take flight?

These two critical moves were the migration of the Ohio Driver License System and the Ohio Vehicle Registration System. These two systems were the backbone of the mainframe and represented over 80% of the mainframe processing.

For each migration, the Ohio BMV was shut down at 6 p.m. on Friday and remained closed through the following Monday at 8 a.m. (Ohio BMV field locations, privately owned entities known as Deputy Registrars, were permitted to continue processing on the Friday of the migration until all customers had been served. They were also required to be closed on the Saturday of the migration despite it being a normal business day). A go live plan was created for each migration with hundreds of tasks and related activities. The go live activities spanned across the entire weekend, 24-hours a day with participation from both IT and the business units.

In the end, while a few functional, performance and internal server communication issues were encountered immediately after go live, which required a "swat team" to solve, all stakeholders involved considered the project to be a complete success. In fact, one key stakeholder commented that the migration was a bit melodramatic as business continued as normal on the following Monday at 8 a.m. In the six months following the migration, the project team has made several modifications to increase system responsiveness, increase system stability, decrease system down time and resolve issues that have arisen.

Lessons Learned

In mid-2007, when Public Safety made the decision to decommission the mainframe and use Windows-based servers, the team searched for other organizations that had completed such a project in the hopes of learning from their experiences. Unfortunately, the information available was extremely limited. This, in itself, was troublesome for Public Safety. Public Safety's goal, by releasing this article, is to enable other organizations to learn from the Exodus project.

What worked?

1. The decision to separate the Exodus project into four smaller sub-projects and then further decompose the work into manageable components. This approach provided the team opportunities for "wins" throughout the project.

2. The selection of the Fujistu compiler to compile the Pacbase-generated COBOL code to run on Windows-based servers. The compiler worked as advertised and was the key technology that allowed the Pacbase code to ultimately execute in a Windows-based environment with minimal change to the code in the Pacbase repository.

3. The limited use of contractors. The majority of the work was done by Public Safety employees which permitted the knowledge to remain in-house and also significantly reduced the cost of the project. The cost of this decision, however, was that other agency projects had to be cancelled or significantly delayed.

4. Communications, in general, were successful throughout the project and especially during the two critical go live weekends. Additionally, an online notification system was used during the migrations to keep stakeholders informed. The Communication Plan was thorough and several meetings were conducted with the interested individuals prior to rollout to keep all stakeholders informed.

5. The expectations of the Public Safety Director's Office and divisions were properly set. This enabled the end users and IT to remain confident when some issues were encountered.

6. When issues were encountered, the team was able to properly prioritize the issues and allocate resources to the highest priority items. This was done through an ad-hoc triage process.

7. The team had executive sponsorship throughout the entire project including during the delays and challenges. Management fully understood the consequences of failure, and constant, open communication to senior management helped keep them focused on the effort. They directly supported the effort by helping others understand the priority and criticality of the project.

8. Prior to going live, significant effort was put into working in a "pre-production mode." This effort enabled several large issues to be discovered and resolved prior to going live. It also allowed the team to increase their general confidence in the process.

9. User acceptance testing was successful because the users were engaged and worked diligently to test the system. The users were more confident with the change because the re-platforming approach meant the same code was executing, and that meant very few errors occurred.

What Could Have Been Improved?

1. The original project estimates were very inaccurate. The team struggled with trying to estimate the work where there was no expert judgment available or models to follow. This resulted in the project needing to be re-baselined several times and the perception of missed dates.

2. The initial phases of the project where design and proof of concept was executed could have been completed quicker had the decision to re-platform not been repeatedly debated.

3. Additional contractors could have been employed at the beginning of the project and assigned the maintenance work of keeping the existing mainframe programs operational. Those resources could have also been responsible for responding to customer requests and implementing new functionality as mandated by legislation. That would have freed up Public Safety development resources to concentrate on the Exodus project.

4. A greater emphasis could have been placed on designing the infrastructure and getting it built far enough in advance to allow for more system and load testing. One contributing factor was that the team did not know what to build as it was unclear how the mainframe code and associated data volume would behave in a Windows-based environment. Nevertheless, Public Safety should have erred on the side of caution and built the environments. If a change had been required or an entire environment needed to be rebuilt, that should have been deemed an acceptable risk.

5. The decision to train the Pacbase resources in .NET during the middle of the project was both distracting and premature as it pulled the team members temporarily off the project and had them focus on a new technology. While Public Safety thought the employees were being prepared for Phase 2, it was later realized that the start of Phase 2 was only an estimate and instead the resources should have been focused on completing Phase 1.

6. Several bad testing assumptions were made. For example, any time there was a call to another application, regardless of how that call was made, it should have been stress tested.

7. Public Safety should have communicated with external IT service providers (e.g., the Ohio Office of Information Technology who provide ODPS email, State Printing, State Network Operation Group, credit card vendor, etc.) well in advance of the go live schedule to make it clear that any potential service interruption during that time frame was not acceptable. One service provider had planned to do maintenance on a go-live weekend that could have impacted our ability to complete the migration.

8. The project team should have had a method to determine the difference between converted code and rewritten code. This would have helped in the troubleshooting process.

9. The lack of full team (e.g., developers, server management, database analysts, etc.) co-location proved to be a hindrance. While the team was able to overcome it, some planning and troubleshooting would have been more efficient if the resources from the different teams were sitting in close proximity. While it may seem trivial at first glance, experience proved that even the distance across a large office can create a communication barrier.

10. Plans should have been made in advance for the business unit testers to check in with IT at a pre-specified location before proceeding to their work areas to begin testing during migration weekends. Instead, when the business unit arrived to test, signs had to be put on the doors to request the testers stop by IT first as they needed to be briefed on a few issues before testing could begin.

11. During the go live weekend, there was an instance when a server needed to be restarted. This restart impacted an actively running production job which caused the entire job to fail. In retrospect, all necessary parties should have been physically contacted to ensure a restart is acceptable (in this case, we would have had to restart anyway, but at least the developer who was running the job would have known in advance).

12. Time should have been allocated to execute a side-by-side parallel test between the legacy applications running on the mainframe and new Windows-based applications. This would have exposed several issues that were not discovered during user acceptance testing.

Conclusion

The retirement of the Public Safety mainframe, which was part of our solution architecture since the early 1970's and was responsible for much of the processing at Public Safety, is truly a remarkable accomplishment. Public Safety completed the project with only the assistance of a few contractors who served in staff augmentation roles. The project team persevered through several "valleys of despair" which, at times, seemed too overwhelming to overcome. In the end, despite the countless unknowns and technical issues, the Public Safety mainframe was successfully retired.

In total, the project took approximately five years and Public Safety spent approximately 84,500 total person-hours (or 42 person-years) working on the project with the majority of the work being completed in the final three years. In the end, more than 2,000 programs representing two million lines of code along with the generation of 30 million letters annually, 400 batch jobs and integration with 2,500 contributing entities were all successfully migrated. The total cost savings to Public Safety and the citizens of Ohio is approximately $7,000,000 - $10,000,000 during the next five years.