

The Value of Prototypes

A Whitepaper on Computeration's Implementation Process

Table of Contents

- Introduction 3
- The Extract-Clean-Load Process..... 4
 - Extraction 4
 - Cleaning..... 4
 - Loading..... 4
- Typical Legacy Data Loaded 4
 - General Ledger Chart of Accounts 5
 - General Ledger Historical Transactions 5
 - Customer, Vendor, and Item Master Records 5
 - General Ledger Open Transactions..... 5
- Proof of Concept 6
- Go-Live Date 6
- Exceptions to This Prototype Method 6
- Saving the Best for Last—Payroll 7
- Summary of Benefits from This Style of Prototyping 8
- About Computeration 8
- About Gloria Braunschweig 9

Introduction

There is a lot of information available on [Choosing the Right Software](#). The clincher, the decision maker, should be the prototype. Before 100% commitment to the software purchase, before all personnel are trained, and before processes and customized components are re-designed, we advocate the prototype.

The prototype eliminates one of the leading causes of project compromise and failure: over budget and not on time. The prototype allows you to test the entire process so you can develop realistic budgets, schedules, and training plans. The prototype points out why you should budget your implementation in steps: (1) prototype, (2) create the budget and time schedule, (3) implement live..

Implementation of [vertical market software](#) into a company that represents the software's "sweet spot," still involves new client personnel, working with consultants they've likely not known for long, with new data the consultants aren't familiar with, and frequently a new service pack level of the software. Each variable is associated with risk even in a vertical market situation, enough so to mandate a prototype. When the installation of software is completed by software generalists, the prototype is even more important. So how do you balance the value of the prototype with its cost?

We advocate using an entire historical year as the subset of data for prototyping, rather than extracting a small subset of data. You'll recognize the following benefits while lowering the cost of the overall implementation:

- A prototype using at least one year of historical data enables you to re-use the prototype data for the foundation of the live implementation, thus work isn't duplicated between the prototype and the live implementation.
- The prototype enables testing of the extraction, cleaning, and loading time of historical data—one of the most variable factors in an implementation.
- During the prototype you can identify the risks and workarounds frequently encountered with extracting data from legacy software.
- Your staff can better preview and train with familiar data, decreasing their learning time.
- Financial and management report formats developed prior to the crunch of the live implementation takes pressure and risk off the Go-live period of time, resulting in a better outcome.
- A prototype eliminates the need to operate parallel systems. You work the kinks out between the old and new system prior to the crunch of the Go-live period of time.
- The estimates of time you and your consultant develop during the prototype are based on better information; you develop a realistic timeframe and cost in advance.

Even sensitive payroll data can be prototyped in this manner without parallel operation of systems. Forms and reports are designed during the prototype for use in the live implementation. The balanced, historical data in the prototype is re-used as the base for the live implementation without duplication of effort. Once the data is transformed in the prototype system, everyone can see what the new system will look like and how it will act with your data. You can use the prototype for development of management and financial reports and for training purposes—familiar data presented in a meaningful format.

The Extract-Clean-Load Process

Extraction, cleaning, loading (ECL) of historical data is one of the most variable factors in an implementation. Rather than using a few selected transactions, manually entered into a “conference-room demo,” selecting an entire year of data gives you a reliable subset of data.

Extraction

The extraction frequently involves unsupported software or a lack of cooperation from your legacy system consultants. The developer of the legacy software may legally object to someone other than their certified consultants analyzing their design structure. Whatever the reason, the first risk to alleviate is associated with gaining access to usable legacy data.

Cleaning

Within the prototype process, your experienced employees execute the cleaning of master records—customers, vendors, employees, and inventory items. The favorite tool to use is Microsoft Excel if the data can be broken into manageable subsets. Microsoft Access or SQL are also utilized with extremely large datasets. The end goal is displaying large quantities of data for fast and easy editing.

Loading

Loading of data is accomplished using a variety of integration tools; most commonly the Integration Manager in Dynamics GP or eConnect with Web Services. Integration Manager and eConnect use the business rules within Dynamics GP to create master records and work transactions—the system checks data integrity just like when data is manually entered.

We can also push data directly into SQL tables using SQL Server Integration Services because we have tools to explain the data format requirements. See the information below for a description of the data typically loaded in the new system.

Typical Legacy Data Loaded

To develop a further understanding of the concept, let’s first discuss a typical example of legacy data to be imported:

- Master records
 - General Ledger accounts, although frequently the format is changed in the new system
 - Customer master records
 - Vendor master records
 - Inventory item records
 - Employee address, tax, pay code, deduction, and benefit records
- Transactions
 - Historical general ledger transactions
 - Open receivables transactions
 - Open payables transactions
 - Open sales quotes and orders
 - Open purchase orders

Note that in the typical situation, historical, fully-paid payables and receivables aren’t imported. The complexity of matching multiple documents against invoices—payments, credit memos, write offs, and returns—makes the import of these historical transactions complex and time consuming, i.e., expensive. While there are exceptions, most companies choose to simply leave the legacy system operative for a period of time on the old hardware.

General Ledger Chart of Accounts

The chart of accounts is usually expanded in the new system to include departments, locations, cost centers, etc. The format—the pattern of the segments—is usually changed. We've found it easier to import the historical general ledger into a chart of accounts identical to the legacy system. That cuts proofing time and when we're ready to change the format of the general ledger chart of accounts, we're on familiar ground using Dynamics GP tools. The old to new format can be translated during the loading of the historical data using a translation table within Integration Manager; however, we've found the validation of the account balances afterwards to be difficult with small companies near impossible with large companies.

General Ledger Historical Transactions

Most implementations involve several years of historical general ledger data so a company can produce comparative financial reports. Risk factors cloud our ability to accurately evaluate the data to be extracted. Frequently in legacy systems we encounter data damage, mixed date formats, and closing transactions handled in obscure methods. The only way we have to judge how well and how long it will take to extract, clean, and load legacy data is to complete a sizable portion of it in the prototype. Usually the largest data file is the balanced general ledger transaction file.

As an accountant, "balanced" means balanced to the penny. I have seen systems that are a few dollars off, or "close enough." Then found they were hundreds of thousands of dollars off in both directions with unrelated transactions coincidentally coming close to offsetting each other. We do a cross-check with debits and credits, number of transactions, and testing of major account balances.

Companies frequently require 6 years of historical data. To use the prototype to test our processes, we'll import all the historical detailed transactions. We write the routine then walk away; we don't care if it takes 6 minutes, 6 hours, or 6 days to load. Once we have the historical years loaded, we can duplicate historical trial balances quickly and easily.

Once we have our proof of concept, the four oldest years can be consolidated into month-end balances. That's easier than developing one set of load routines for month-end summary balances and another set of load routines for detailed transactions.

Customer, Vendor, and Item Master Records

The next set of data typically requires the most cleaning: customer, vendor, and inventory master records. The most commonly used tool for cleaning is Microsoft Excel with its familiarity, powerful searching and sorting, and pivot tables.

We also pick between two methods to track additions/changes/deletions, based on the size of the data, so we don't have to re-extract, re-clean, and re-load the historical data.

1. Client personnel maintain a manual log of changes, additions, and deletions
2. We perform an electronic comparison of the files from the prototype against the same files extracted anew from the legacy system, producing a report of the changes. The client then cleans this smaller set of data.

General Ledger Open Transactions

Once we've completed our prototype, it's not uncommon for us to complete a second wave of integrations with general ledger open transactions. We initiate the live database then start working towards a very rapid "Go-live" date. We "draw a line" with the client. For example, if we're implementing in August, the client agrees not to post anything older than June 30th. Performing this second wave of imports allows us to prepare for a very quick last wave—those transactions created just before our "Go-live" date.

This second wave has both risk and reward. We may have held the historical GL transactions in the legacy format. We may wait until our Go-live date to convert to the new format. Practicing that conversion at least once under pressure assures us we won't delay the Go-live date. If we decided to transform the GL transactions to the new format, we get a practice session before the Go-live date utilizing this second wave of ECL.

Proof of Concept

This is the point where the prototype ECL is completed. Now it's ready for writing financial statements, testing open transaction entry, developing customizations, and modifying management reports. We use all of these components together for testing and training.

A redundant restorable backup of the system is retained at the appropriate points so we can use this dataset as the foundation for the live implementation.

Go-Live Date

I always tell my clients if they've done a prototype, the rest is easy. They'll be ready for their Go-live date and wonder what the worry was all about on that day.

Usually Go-live is as easy as several hours of training on a module, then we start the client entering the open balances. We implement a module-a-day in small companies. Large companies will have a sales team, payables team, inventory team, etc. so we implement several modules in a day.

Exceptions to This Prototype Method

A common exception to this method is when we're implementing multiple companies for a client, as many as 30 to 60 companies, for example. In this instance, we typically implement an entire company as the prototype. It's important to choose the right size of company when this is the template method. Too large of a company slows down the prototype and users lose training skills while they wait. Too small of a prototype company and we miss important details.

One small manufacturing company utilized their prototype to re-design everything from the ground up. They tried several general ledger account formats and dependent financial statement designs. They re-organized their customer and vendor classes within the prototype. They completely re-numbered their inventory items, resulting in new bills of material, also testing their concepts of units of measure several different ways. This prototype was prepared with a very small subset of data to make the prototype nimble and easy to change.

We've had some implementations with extremely tight time frames and the added complication of incomplete historical data. In some cases, the companies have been as many as four years behind producing general ledger trial balances, financial statements, tax returns, and bank reconciliations. In these cases (more than one so it's not uncommon), we may initially skip the historical transaction import and simply use open transactions. The prototype is very skinny on data; just enough for us to demonstrate proof of concept.

We import open transactions, train, and go live within days. Then we revert to work on the historical data; in all cases this was just the general ledger historical transactions. We import the oldest legacy year's transactions into work transactions, prove we balance to the legacy data, post, and close the historical year. We continue to march forward through the historical years. Because of Dynamics GP's ability to hold open unlimited years, this method works in a pinch. Its success depends on the GL control accounts truly balancing to the subsidiary receivables and payables ledgers. I've seen companies take more than a year to come into balance if this isn't handled correctly.

Saving the Best for Last—Payroll

One implementation I haven't discussed is Payroll. I've been chased down in a parking lot because an employee's new weekly check was \$0.02 different from last week's check. Employees are sensitive about their payroll!

Companies always want to run parallel with their payroll application. We have successfully deployed relying on a prototype instead, eliminating months of parallel processing. In my opinion, parallel processing is an outdated process. Client personnel can barely manage learning and using the new system, let alone running two systems at once.

First of all, recognize that payroll reporting is date dependent from the day of the pay run to the week, to bi-weekly, semi-monthly, monthly, quarterly, and yearly.

Secondly, the setup of all the employee pay, benefit, deduction, and tax codes is absolutely critical to payroll. If you don't mark a benefit as tax-exempt, tax calculates incorrectly and you have a red flag instantly. You may look at the tax calculation and think the problem is caused by exemptions, but actually the tax-exempt status of deductions and benefits may be at fault.

Thirdly, vital to this process is a re-storable backup before any payroll transaction is processed.

An example explains it best. Let's choose Monday, August 17, 2009 as the "Go Live" date, with weekly payroll paid on Friday.

We initiate a payroll prototype on the balance of each employee's codes as of the end of the first calendar quarter ending March 31st. We create one big transaction, a "manual check" that is the total amount of their gross and net pay, benefit, deduction, and tax codes. This one big transaction is exactly equal to the legacy system.

Then we do the same thing with the second calendar quarter ending June 30th.

And then we create another "manual check" for the month of July. We have matched our legacy system exactly to the penny in all aspects at this point to first quarter, second quarter, and the seventh month.

Now we insure we have a redundant restorable backup. Remember that we're at a point in time more than two weeks after the first weekly August payroll. We duplicate that first August pay run and troubleshoot all differences. We coordinate this with less pressure than if we're trying to do both pay runs on Friday, August 7th.

Then after we've discovered all differences, again with coordination so we're not under pressure, we duplicate the August 14th payroll. We can test, train, and operate with this prototype for as long as necessary (up to year end).

Sometimes, we re-do these historical pay runs for as much as a month because some deductions aren't subtracted every pay run. This prototype process continues until we're confident. We aren't running parallel; we're running the new software after the existing system is completed.

But when we're finally confident, we go live on the new system, changing it to first position. The old system is immediately retired.

With each of these pay runs, we match the new system payroll to the legacy system. If we don't match (and we usually don't), we delve into the details to find out why. Dozens of reasons can cause the difference:

- A pay, benefit, or deduction code wasn't correctly setup exempt or non-exempt from tax

- Garnishments weren't prioritized correctly
- Hours were input incorrectly
- Dynamics GP uses a slightly different tax calculation method than the legacy system—there are 8 acceptable methods

I have found that usually the legacy system is at fault. We usually make mistakes the first time. So we usually find the errors were in the old system.

With a payroll prototype prepared in this fashion you get the benefit of the prototype, the proof of concept of duplicating (or finding the errors) in the legacy system, and it's less pressure with a better outcome than running completely parallel. You've simply re-created past payrolls and found your differences before you go live.

At each step, you've taken a restorable backup. SQL Server backups allow do-overs!

This simple concept of handling a payroll implementation with a prototype takes the pressure off and saves months of running parallel.

Summary of Benefits from This Style of Prototyping

- The ECL of data isn't done twice
- The live implementation moves more quickly because the bulk of the ECL has already been completed
- Balancing and proofing against the legacy system is faster and easier
- The prototype provides familiar data for training purposes
- The prototype provides familiar data for management and financial report development
- Conversion of master record numbers and ID's is completed within in Dynamics GP where we're confident of tool use
- The client gains invaluable experience with the Dynamics GP tools and application
- The data is portable
- Prototyping with historical data eliminates the need to operate parallel systems

The prototype doesn't add to the expense of an implementation. With processes developed for small and mid-size businesses, the work is re-usable in the live implementation and much of the risk has been eliminated.

About Computeration

Specializing in multi-location implementations of Microsoft Dynamics™ GP in the Pacific Northwest, with clients around the world, Computeration makes your implementation your implementation successful by offering experienced project management, data integration, training, and consulting services.

We've been serving companies since 1990 and have a reputation for staying on time and on budget with the utmost integrity. Our clients are:

- Manufacturing and wholesale distribution companies that manage inventory and may also provide service for their products.
- Holding and family group companies with asset management needs, and
- Not-for-profit, municipal, or tribal agencies requiring asset, property, and powerful financial reporting.

Computeration relies upon proven prototype procedures designed to lower your risk by testing your implementation prior to live deployment. We provide exceptional service, training, customization, and support as a Microsoft Gold Certified Partner.

About Gloria Braunschweig

Gloria has over 30 years of experience as an internal and external project manager implementing and operating systems on local area networks, wide area networks, hosted co-locations, minicomputers, and mainframes. She has held positions as a CFO or controller, or consulted for companies in software development, discrete and process control manufacturing, wholesale and retail distribution, service, high-tech, law, not-for-profit, education, entertainment, and hospitality.

Initially as a purchaser of systems, Gloria learned the value of good support and project planning. Holding to her values as a consumer, for over 18 years she has established long-term relationships with clients and managed consultants involved with the implementation of business information systems in mid-sized organizations. As a business owner, Gloria has experienced the full spectrum of business operations and management. This facilitates her relationship with her clients and increases her understanding of business challenges.

Gloria helps her clients economically justify their system cost and leverage information into knowledge. She has been a Certified Management Accountant for over 25 years and holds numerous Microsoft Dynamics certifications along with certifications on a number of products which tightly integrate to Dynamics.

Computeration, Inc.
15350 SW Sequoia Parkway
Ste 198
Portland, OR 97224
P: 503 968 7964, 866 808 7964
F: 503 968 7076
www.computeration.net