WHITEPAPER

The Essential Guide to IBM i Database Modernization

IBM i Enterprise Modernization

Leveraging DB2 SQL and;
Optimized for IBM PureSystems.

inFORM Decisions
Leaders in Document Automation

and

AO
Adsero Optima™
Reclaim Your Heritage

Those within the IBM i community know the platform for its reliability, scalability and for delivering tremendous business value with unusually low operational costs. So why is the system perceived as being “legacy?” Many generalize their thoughts on the green-screen user interface or a lack of effort to publicize the value proposition of the IBM i, or both. Nonetheless, based on extensive research by IBM and others, we strongly believe that the lack of application agility is the number one factor limiting the growth and acceptance of this remarkable platform. In fact, the biggest constraint of application agility is neither the user interface nor the RPG-written applications, it’s the fact that we’re not using the SQL capabilities in the DB2 for IBM i database.

This whitepaper will explain:

1. Why companies should consider modernizing their application’s database;
2. The database modernization challenges companies have faced in the past;
3. The benefits of migrating legacy or heritage database objects;
4. The new technology advancements that can be leveraged after database modernization has taken place.

First and foremost, it is important to ask the following:

What is the cumulative value of the intellectual capital invested in your applications?

What competitive advantage do your current application systems provide?

Without a doubt, there is significant value in your “legacy” or “heritage” applications, and this value can be leveraged relatively easily for dramatic results and benefit to your company. It is entirely feasible (rather simple, actually) to extend the life of these applications while remaining competitive. Our goal is to build a case for database modernization that supports your business goals by defining the value in your heritage application system.

Modernize Your IBM i

IT modernization is the process of extending the value of aging platforms and systems into the future by adopting more modern and efficient technologies. This can be accomplished whilst removing the operational constraints that inhibit growth and agile response to changes in the business environment. However, what should be thought of as an ongoing process of IT system improvement has become known to be a large and seemingly daunting project, because incremental improvements to monolithic code often exhibit “gotchas” along the way. As a result, the importance of system improvement is often forgotten as IT personnel are consistently challenged to ‘do more with less.’

Interestingly, modernization has continued to be a top priority for CIOs. The results of a survey conducted by Gartner of over 2,300 CIOs indicates "Legacy Modernization" as being one of the top-10 CIO business and technology priorities for 2012.[1]

So, why is the IBM i perceived as legacy?

1. **User Interface/User Experience (UI/UX)** – The user interface is the most visible part of the application that younger CIOs and CFOs see as being the default interface of the IBM i and as such are often quick to pigeon-hole the entire platform as legacy. In addition, younger programmers and users cannot relate to the 6 – 8 character field names and the 6 – 10 character file names of DB2 for i, which older programmers have historically employed to name database constructs and elements.

2. **Old style monolithic code** – In previous years, developers created huge “structured” monolithic programs, as that was the programming model of the time. As a rule-of-thumb”, 80% of the lines of code in these monoliths dealt with database relationships and

database validations. Additionally, there was little-to-no separation of function, which made maintenance increasingly problematic. The way in which this presented itself, was massive maintenance backlogs and frustrated users.

We seldom or, more accurately, never improved our applications by introducing newer coding techniques, leveraging new enhancements to the language, operating system and database, not to mention improving the maintainability of the applications, because the system just kept on running. Now, IBM shops strive to remain competitive but are facing several challenges: skyrocketing costs are constraining IT budgets; the many years of intellectual property invested in heritage systems need to be retained and leveraged and major advances in analytics, cloud, mobile, and database engine technologies must be utilized. Fortunately, this problem is solved through database modernization.

Understanding Database Modernization

Companies must think strategically to correct the fundamental challenge brought on by lack of application agility, which is retained by old code and an inflexible database structure.
Often, companies start modernizing the UI because it's the most visible part of the application. The most important piece in modernizing the IBM i should be in modernizing the database and eliminating monolithic code so that discrete functions of core applications can be leveraged for a variety of benefits.

In addition, our old application architectural constructs and monolithic code cause unmanageable maintenance burden, which delays change and enhancement requests, which in turn frustrates users and business managers.

In order to create application agility, applications that were designed on old standards have to be fundamentally re-constructed in order to unlock their massive value, retain the competitive advantage facilitated by those applications and recover the business rules in re-usable components.

All modernization has to start at the database engine layer, as most of the advances we have seen introduced to the platform since 2000, pre-supposes the use of an SQL database engine. In fact, in our experience, if the SQL engine is leveraged, your code base can be reduced by approximately 80%, by simply moving database functions we historically coded in RPG (or other HLL), down into the database engine. It is possible to implement a solution with minimum disruption using a low-risk, gradual, iterative process to ensure no loss of business continuity and to support parallel databases during the process. This is an automatic, gradual and non-disruptive process which does not require any manual recompilation of programs during the initial phase of the migration. Subsequent refactoring or re-engineering of the code at both a code and database level will require recompilation of the objects.

Signs that you need to modernize your database

Operational constraints – Modernizing your database allows you to implement many technological advancements, so that you can achieve greater functionality from your current system.

Application maintenance is an excessive burden – The maintenance burden is often the cause of people moving off of the IBM i system. Most surprising though is that few of the installations have made the connection between the maintenance burden and the cause of this problem: their monolithic code.

Maintenance backlogs and frustrated users – IT developers can simply no longer cope with the maintenance burden of their system, as they not only have to effect the change, but they then have to test every single piece of code that was changed. What usually happens is that the end-users of these systems become frustrated, as changes simply take too long to implement in the very demanding business environment of today.

Cannot find quality resources to support it – Not everyone can maintain your IBM i system, understand and have the programming skills to hand write code. These rare resources are scarce and can be costly.

No executive buy-in – Most people are afraid of what they don't understand. With a modernized system, new executives are more receptive to IBM i technology and all of the benefits that you know so well.

Data integrity issues – New employees may be affecting the integrity of your database. In fact, duplication may have occurred over time at a metadata and structural metadata level which has compromised the integrity of your **database**.

Lack of agility – Your business may be constrained, held back by the lack of flexibility within your system. If you can no longer respond in an agile fashion to changes in the business environment, that's a huge problem. If companies cannot respond rapidly to changes in the business environment, companies are often doomed. As a result, executives and users start looking for alternative application solutions that facilitate rapid response to changes. Lack of agility also includes being held back from technology advancements, by the requirement for a modern graphical user interface, for 24x7 access via mobile devices, smart phones and web browsers.

Workload consolidation – Clients running multiple servers want to increase the utilization of server hardware by allowing one physical server to host multiple virtual machines. This allows the server to operate its original workload as a virtual machine and host additional virtual workloads simultaneously. At the same time, clients want to consolidate legacy databases and applications. A move to next generation expert integrated systems that have been optimized for enterprise modernization allows clients to run more workloads on less hardware using the powerful DB2 SQL database without having to replace any of their applications. Furthermore, power and cooling demands and maintenance costs are lowered translating to lower operating costs and lower capital-intensive projects.

IBM i Strategy and Roadmap

IBM has introduced a myriad of major database technology enhancements since 2000 and over 95% of announcements on the topics of IBM programming language, operating system, database engine and application development since 2000 presupposes the use of the DB2 SQL (SQE) database engine personality on the platform. Despite IBM's indication that SQL is the strategic database interface for IBM i, the bulk of all customers still use record level (RLA) or native IO access as the primary database access method.. This means that although the installed base has the most advanced implementation of the DB2 database engine at their disposal, they are using the DB2engine severely "shackled" or constrained, because it kept on processing transactions the "old" way.

Legacy database engine access (CQE and RLA)

Prior to the advent of the latest generation of relational database engines and modern programming techniques, application programmers were forced to physically code all the relational logic (i.e. the relationship between customers, all valid customer delivery addresses, all orders for customers, all products per order, etc.) as well as all the database validations manually. This is in addition to the business-unique application logic (for instance how orders are fulfilled, from which warehouse products are shipped, how picking is optimized, etc.), which usually translated into the competitive advantage for that particular user of the application.

As a rule of thumb, about 80% of the lines of code of any "legacy" commercial application, is focused on entity (database or "object") relationships (such as customer to customer locations, customers to open orders, order lines to orders headers, products to orders, etc.) and validation rules (product number to a valid product, values chosen within a valid range, postal code - a valid postal code, ID number - a valid ID number, etc.). As these relationships and validations have to be maintained in every single application that updates these files, it can become a very tedious and laborious process to update these efficiently. This is where a lack of application agility can bog a system down as every instance of a validation or relationship has to be identified in the entire application in order to effect a change.

Why SQL?

Regardless of IBM recommending SQL since 2000, one question arises: Why move your application's database objects from the old DDS definitions to native DB2 DDL definitions (SQL)? IBM said it themselves in the whitepaper 'DDS and SQL – A Winning Combination for DB2 for i':

"Those IBM i clients who have failed to keep track of this SQL metamorphosis are missing out on a plethora of new SQL-based functions that can make it easier for their IT teams to meet the ever-changing list of business requirements."

IBM recommends data-centric programming in which you allow DB2 to perform the data processing using SQL. The benefits of using SQL include:

1. Improved agility - Flexibility and openness through SQL allows companies to be agile and regain their competitive advantage.
2. Utilizing the latest technology advancements from IBM - The DB2 SQL engine has been the foundation of all developments and enhancements to IBM i (and predecessors) since 2000.
3. Ability to present a modern database, improve data integrity and enhanced analytics.
4. A plethora of documentation, database tools and resources are available due to the widespread knowledge of SQL.
5. Significant performance enhancements. Using SQL can significantly reduce the size of code, which improves performance.

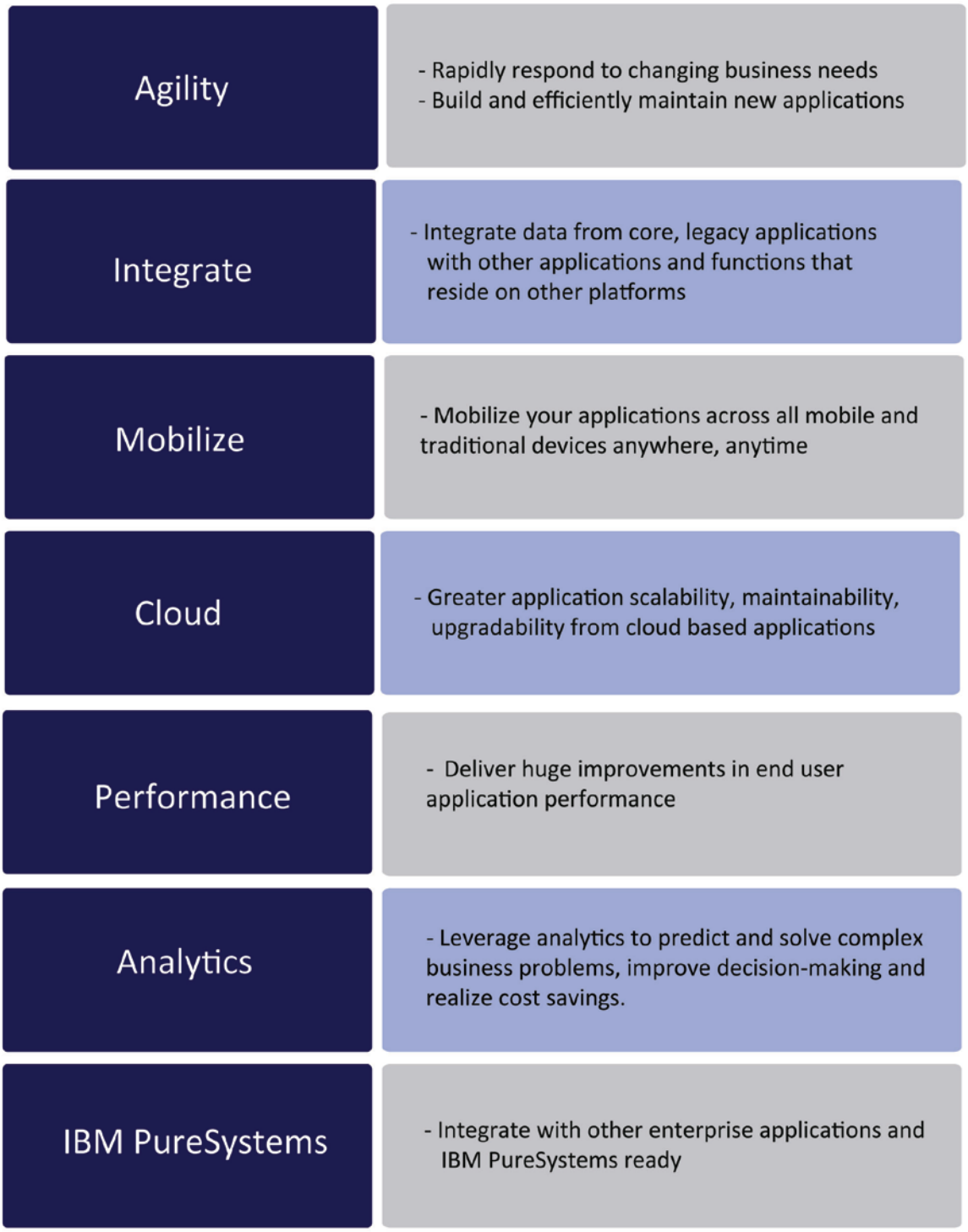
Table 1 displays the comparison of the major DB2 enhancements made to the SQL and non-SQL interfaces since Version 5 Release 1 (V5R1).

Enhancements to non-SQL interfaces	SQL enhancements
Unicode support - UTF-16 and UTF-8	Unicode support - UTF-16 and UTF-8
Binary character data type	Binary character data type
CRTLF PAGESIZE parameter	CREATE INDEX PAGESIZE keyword
Larger decimal support	Larger decimal support
SSD enablement for physical and logical files	SSD enablement for tables and Indexes
	XML, National Character, and ROWID data types
	Identity column attribute
	Hidden and Automatic Timestamp column attributes
	Field Procedure column attribute
	Sequence object
	Column-level and Instead-Of triggers
	Merge statement
	OLAP and Super Group expressions
	Create Table from Select and Insert from Select
	SQL functional indexes
	XML Publishing and Decomposition functions
	IBM OmniFind Text Search Server
	SQL Query Engine (SQE)
	SQE Result Set Caching
	SQE Autonomic Indexes
	SQE Self-Learning Query Optimization and Adaptive Query Processing
	SQE Encoded Vector Index fast path for aggregate processing
	SQE In-memory Database Enablement
	IBM i Navigator Plan Cache Tool

Table 1: *Comparison of DB2 enhancements since Version 5 Release 1 (V5R1)*

Benefits of database modernization

Modernizing your database allows you to preserve your investment in your IBM i system and retain access to all of your heritage applications and data. That alone provides significant savings, but there are many other benefits of database modernization including:



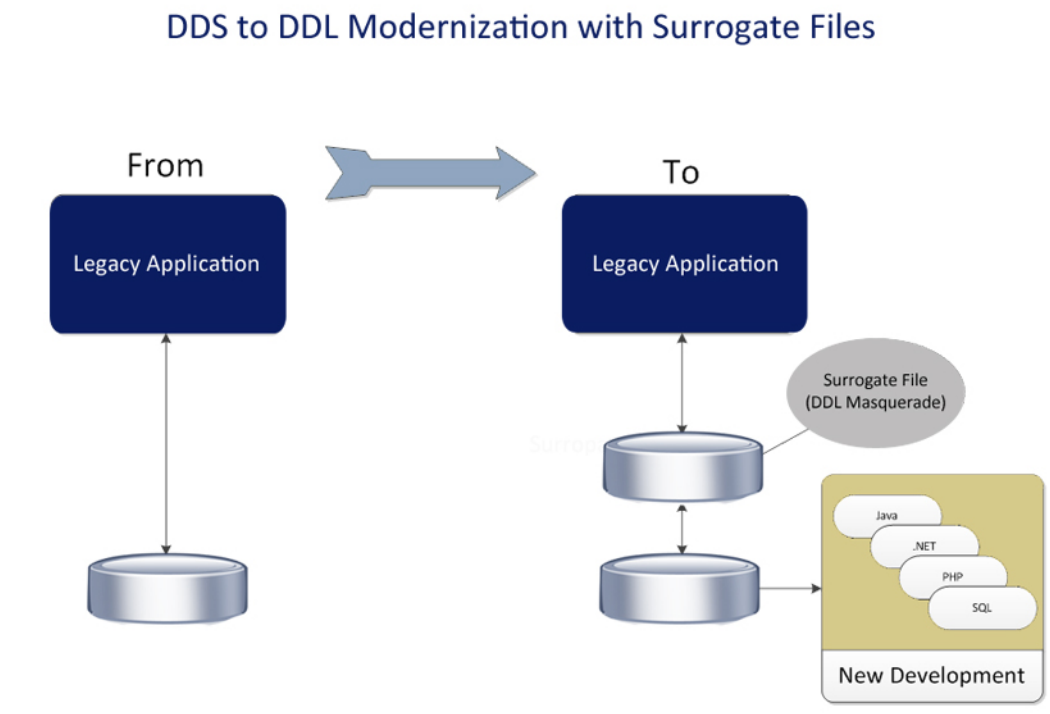
Methods for Implementing Database Modernization

Different methods exist for modernizing the DB2 database from old style database definitions (DDS) to SQL (DDL). It is important to understand these different technology options and know the pros and cons of each. Fortunately, new technology has been introduced that changes the process, making it a low risk, gradual, non-disruptive and relatively simple process. Before we look at this new technology, let's look at the other options which have been known to introduce challenges and inefficiencies.

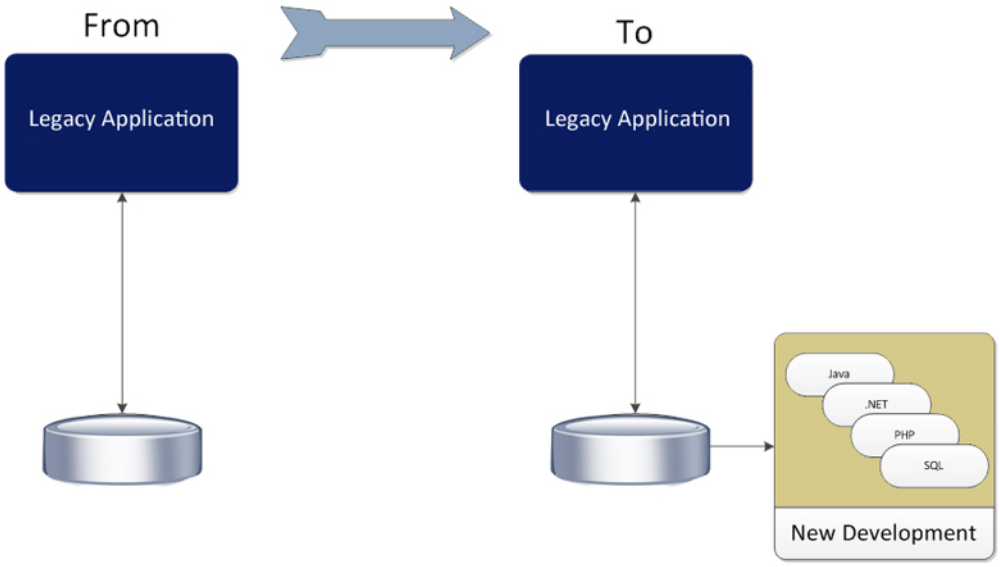
Use of Surrogate files

The use of surrogate files has been recommended in the past by various software vendors as an alternative to the manual conversion of the underlying database infrastructure. "Surrogate logical files" masquerade as physical files to heritage systems thereby allowing the heritage systems to access the "new" database files without the need for recompilation.

While the use of surrogates are beneficial when new applications are being implemented and heritage applications remain unchanged, bear in mind that approximately 80% of the lines of code (all lines of code implementing validations and enforcing data relationships) currently in your heritage application, will eventually and over time end up in the database engine. Then consider where you want to go with your heritage application versus implementing a native migration for a natural long-term approach. Decide if you want the surrogate approach which limits heritage applications or if you want the full might of the DB2 athlete at its disposal.



DDS to DDL Modernization without Surrogate Files



Other Options and Challenges

Access to the source code – Some vendors require access to your source code for modernization implementations. It's important to remember that changes to your source code may force you to be locked into a particular vendor.

Disruption of users – In the past, with a database modernization implementation, the disruption to users and the business were significant. In addition to running two systems in parallel, companies were forced to pay double maintenance with any new or interim solution.

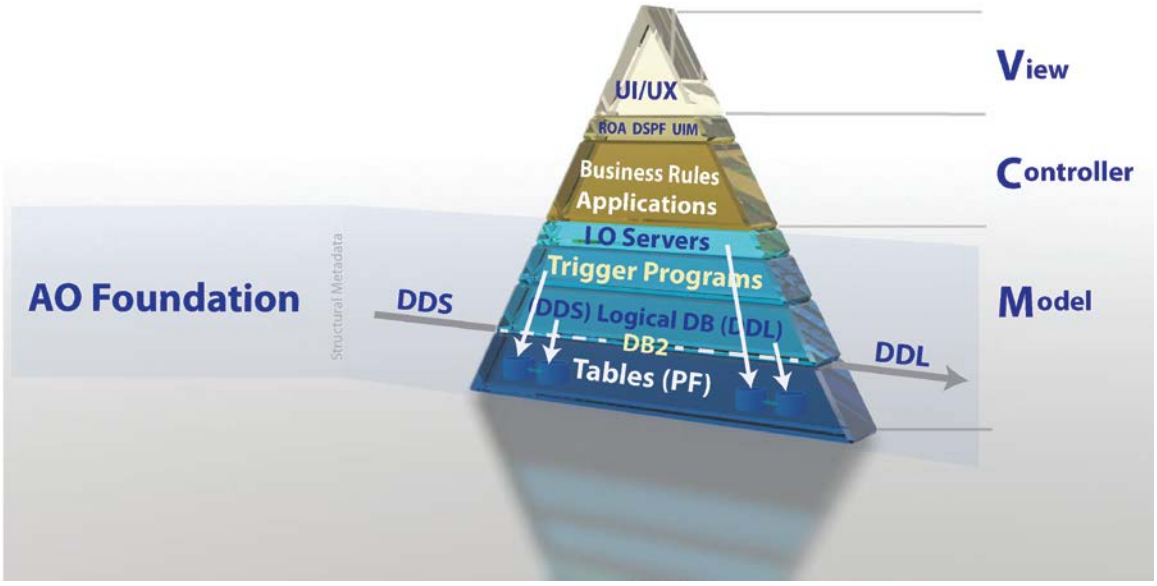
Old programming tools and anomalies built into the system over time – Due to the sheer age of the applications and regardless of the management practices, a LOT of anomalies have been introduced into our systems over time. Some of this was caused by "emergency" maintenance, but a LOT of it is simply caused by neglect – inconsistent validation and relationship rules in our code.

Native Modernization Implementation

It is our belief that the only lasting modernization strategy has to start at the fundamental database definition level. Any other adjustments or maneuvers are tactical moves at best. They will not in the long term remove the fundamental barriers to a permanent solution. The challenge is how to achieve this with the minimum disruption to your users. Native modernization provides the lowest risk for a solid, long term foundation, from which you can then start to extract maximum benefit out of your heritage application. It is imperative that you transfer your database definitions into DDL, with the bulk of your relationships and validations moved out of your application logic into the DB2 database. This will provide you with the foundation to start leveraging the incredible capabilities of SQL and the DB2 engine. DB2 should be allowed to do all the "dirty" work (or "heavy" lifting) for you, so you are free to focus on delivering innovative business solutions and logic.

Multi-Tier architecture - Multi-tier architecture (or MVC in technical parlance) in modernized applications allow companies to achieve separation of code in the database (M), user interface (V) and business logic (C) that result in more agile applications and improved maintainability.

This illustration shows what our optimum application architecture should look like and which elements are non-negotiable going forward. By implementing this architecture, we can reclaim our rich heritage, whilst removing those building blocks that have caused a LOT of pain for IBM i developers. It is significant that if modernized properly, you may end up maintaining as little as 10% of the lines of code you had to maintain originally in your heritage application.



No vendor lock-in, complete control – We believe that you should never exchange one dependency for another - quite often more proprietary than the previous. Take ownership of your applications and the destiny of your systems. With native database modernization you retain complete control of your database. You receive full management capabilities of the entire database, controlling who may change structural metadata, providing a full audit trail, etc.

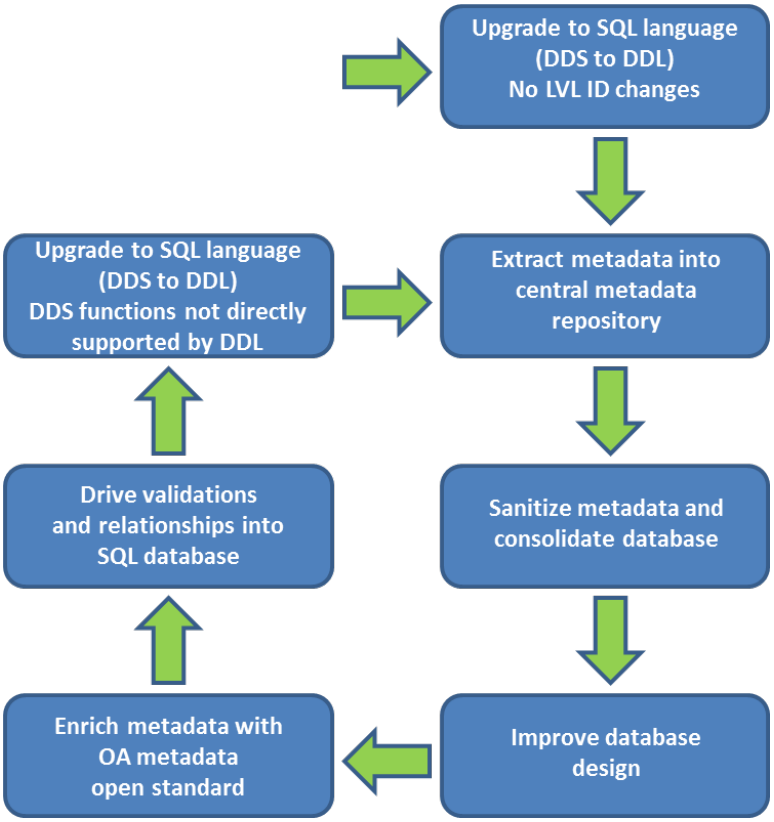
Non-disruptive, low risk – Designed to facilitate a completely transparent, non-disruptive migration from DDS to the native SQL engine offers a low risk implementation with little to no downtime for users. Your DDS and DDL-based databases remain fully in synchronization to allow transaction process mirroring.

No LVLID changes in Phase 1 of database modernization – No LVLID changes during Phase 1 (initial DDS to DDL migration) of the database upgrade process, hence no recompilation of any code.

Gradual sanitizing of metadata, structural metadata and database – One of the key considerations of our entire modernization philosophy and roadmap, is to gradually, modernize your system as you perform routine maintenance. The benefit is immense and you implement the very desirable “continuous improvement” philosophy.

Small incremental steps, fast ROI – We believe that you should introduce a long term strategic roadmap, focusing constantly on ROI and addressing the fundamental causes and perceptions of your system being regarded as "legacy". We always approach this from the 80/20 rule perspective, where experience indicates that 80% of your transactions are generated by 20% of your application. These can be migrated quickly and with significant ROI, but must be part of a long-term strategy.

How to migrate to SQL



Additionally, there is no need to modernize the entire system. In our experience, the most sensible way is to consistently consider the ROI and benefit of modernizing a specific function. The perfect initial candidates are the 20% of business function that generate 80% of transactions. Modernizing those first, will change perceptions and reduce your maintenance burden dramatically. Other candidates for modernization are the programs that require continuous maintenance. You can track these down by carefully analyzing your maintenance requests. By refactoring these programs, operational and architectural deficiencies can be corrected.

Clearly so-called CRUD (Create, Read, Update, Delete) or BREAD (Browse, Read, Edit, Add, Delete) functions which provide relatively little in terms of ROI and should be left for last, or whenever you have a maintenance request.

Migration To IBM PureSystems

Adsero Optima has been optimized specifically for IBM PureSystems. Our PureSystems-Ready enterprise modernization solution allows you to achieve an increased return on investment on the expert integrated system. Companies are able to keep operational costs down while delivering new functionality. PureSystems Ready solutions improve performance; cut costs and future-proof IT investment. AO provides a simple migration path to IBM PureSystems.

For more information on IBM PureSystems:
http://www.ibm.com/ibm/puresystems/us/en/index.html

Use Case – Cape Gate

Cape Gate is one of the Top 3 Steel Producers in South Africa. It is running a HEAVILY customized old System/38 based version of the JBA ERP application dating back to 1986. Due to the level of customization, they believed that new releases of the ERP application were not likely to benefit them, so they bought the source code in 1986 and have been self-sufficient and self-reliant for any required enhancements and changes. Cape Gate runs one of the top RPG development shops in South Africa and are currently on IBM i Version 6.1 with an upgrade to version 7.1 being planned. The order entry module, for instance, has been essentially redeveloped over the years from the ground up, providing Cape Gate with significant competitive advantage. Their own investment into their system exceeds 150 man-years of effort, excluding the original base application - mostly RPG III.

After critically considering the functionality offered, and distinguishing between operational and functional deficiencies, the company recognized that there remained much value within their heritage applications. Cape Gate also realized that their competitive advantage was essentially "hidden" within their heritage application. If this could be recovered and encapsulated, significant additional ROI was possible.

Cape Gate was not that concerned about modernizing their UI, as they run a typical heads down, blue collar data capture production environment. In that kind of environment, limited function character based workstations remain the best solution.

Challenge:
Due to the age of the systems and the monolithic nature of the code, the maintenance burden was excessive. While Cape Gate was happy with the functionality offered by the systems, they were frustrated by this excessive maintenance burden and the delays involved when introducing changes.

To their shock, they discovered that the best functional fit they could get from the latest and greatest ERP applications available, was in the mid 60% range! In their words "we could get all the bells and whistles, in the latest "gee whiz" presentation layer, but essentially all their competitive advantage would be lost. Additionally, if they had to add the lost functionality, they would lose a number of years developing that functionality, whilst essentially running two systems in parallel (and ending up with the same result). The combined cost was simply more than they were prepared to consider. Another realization was the fact that these replacements would have required the replacement of IBM i and re-skilling of their staff. Replacement simply meant: throw everything you have in the trash can.

Solution:
After research indicated that they had significant value in their current application, Cape Gate decided to modernize their IBM i database. The initial results with IBM i based modernization tools was not very positive, for a variety of reasons. Some of these included inter alia, the nature of modernization projects at that stage ("Big Bang"), the perceived risk, the disruption, availability of skills and the fact that many of the solutions were what can be best described as "band-aid" solutions. A modernization tool was in fact purchased, but their experience with it, especially its use of "surrogates" as an interim step, dissuaded them from continuing with the tool.

After much frustration, they found Adsero Optima's native database modernization solution for transforming their application's database objects from format to native DB2 SQL. This has allowed them to implement the solution gradually and non-disruptively and preserve their investment in the IBM i.

Conclusion

The IBM i is a very powerful platform. Despite the fact that many advancements have been introduced over the years few developers have appreciated the true value of these advancements and continue with systems that are slow and difficult to manage - perceived as "legacy" primarily due to a lack of application agility.

Organizations migrating legacy database objects from the current format to native DB2 SQL have experienced many advantages. They are able to benefit from the significant value invested in their heritage applications and at the same time eliminate unwanted constraints.

By implementing native database modernization they are able to extend the life of their heritage applications into the future while remaining competitive and take advantage of many new technology advancements such as Business Analytics, Mobility, Cloud Computing and PureSystems. All of which can now be leveraged because database modernization has taken place.

Through native database modernization, it is possible to implement a solution with minimum disruption using a low-risk, gradual, iterative modernization process to ensure no loss of business continuity and to support parallel databases during the process. This automatic, gradual and non-disruptive process allows implementation without the use of surrogates, without Level ID changes or access to source code. This can be achieved without any risk or down time.

The final result is a modern application architecture with a clear separation between the database, the user interface and business unique application logic. This means that companies can reclaim their agility, regain their competitive advantage and respond more rapidly to changes in the business environment.

One question remains: Are you going to gain a long-term competitive edge with your IBM i heritage applications or lose the significant value that's still left?

Sources:

1. Gartner Executive Programs' Worldwide Survey of More Than 2,300 CIOs Shows Flat IT Budgets in 2012, but IT Organizations Must Deliver on Multiple Priorities http://www.gartner.com/it/page.jsp?id=1897514
2. DDS and SQL – A Winning Combination for DB2 for i http://www.adsero-optima.com/resources

About TEMBO Application Generation (Pty) Ltd

TEMBO Application Generation (Pty) Ltd. specializes in the development of enterprise modernization solutions for IBM Power Systems running IBM i. Our flagship product, Adsero Optima (AO) modernizes legacy databases to the powerful SQL, extending applications and providing a solid foundation for future modernization technologies and projects.

Our primary focus centers on full spectrum modernization deployments that result in improved IT efficiencies, specifically DB2 SQL Migration. We offer deep expertise in Design Recovery (refactoring), BPI, Business Agility, SOA, SaaS and Cloud Computing.

To fully exploit the power of IBM i, we leverage ILE RPG IV, Embedded SQL, SQL (DDL, DML), Stored Procedures, looksoftware and similar tools, Zend framework, PHP, JAVA, .NET, AJAX and RPG Open Access (ROA).

TEMBO
Application Generation (Pty) Ltd

inFORM Decisions
Leaders in Document Automation

To learn more, visit info.informdecisions.com/content_html/ao.asp

AO
Adsero Optima™
Reclaim Your Heritage

Americas: inFORM Decisions • www.informdecisions.com, Phone: (949) 709-5838 • (800) 858-5544 • Email: info@informdecisions.com
Africa: TEMBO Application Generation • www.adsero-optima.com • Email: salescontacts@tembotechlab.com
Asia Pacific: TEMBO Application Generation • www.adsero-optima.com
Europe Middle-East: TEMBO Application Generation • www.adsero-optima.com • Email: phillipd@tembotechlab.com