MODERNSYSTEMS

WHITEPAPER

NATURAL SELECTION

INSIDE THE BUSINESS CASE, RISKS AND SOLUTION FOR EVOLVING
BEYOND SOFTWARE AG'S NATURAL/ADABAS PLATFORM

Introduction

Software AG, one of the remaining IBM mainframe independent software vendors (ISVs), has been selling the Natural/Adabas application platform for many decades on the IBM platform. While the Natural language has a COBOL-like syntax, it has its own proprietary runtime environment. Adabas, a high-performing, prerelational, inverted-list file structure often, but not always, accompanies many Natural-based mainframe applications. Despite running mission critical applications and transactions on this platform, specific factors are driving companies away from Natural/Adabas.

Who Should Read This: Relevance by Organizational Role

This content specifically helps:

- **CIOs** looking to streamline systems and add efficiencies
- **CFOs** looking to dramatically reduce Software AG license spend
- **Line of Business Managers** looking to integrate platforms for better service levels and performance
- **Business Analysts** seeking to leverage data analytics and/or business intelligence to gain competitive advantage
- **Enterprise Architects** seeking to end dependency on a legacy technology and introduce new agile development methodologies

What You'll Gain From Reading This Document

This document incorporates industry analysis, direct customer feedback and 20 years of project experience to help you **understand, build and execute** on a business case for modernizing Software AG Natural/Adabas environments. The content within:

- Summarizes the market trends driving migration
- Identifies Best Practices and Critical Risk Factors
- Shares two Natural/Adabas modernization success stories driven by stated market trends
- Shares insight into the technology used to modernize Natural/Adabas environments

Relevant Environments, SAG Products

- z/OS, z/VSE, OpenVMS
- Natural: Report and structured mode, Construct-generated programs, Entire-X, Natural Process
- Adabas access from Cobol - ADAPREP, ADAPRI, ADASQL, direct calls

About Modern Systems

A global leader in modernization since 1983, we have:

- Been trusted by Walmart to modernize the world's biggest order processing system
- Completed countless large scale, successful Natural/Adabas modernization projects worldwide
- Developed a solution set that offers the most choice and least risk for modernizing legacy applications and databases

Building the Business Case

With business buyers and IT buyers both striving to extend the life of existing applications, yet still bring additional functionality and value to the business, modernization efforts for Natural/Adabas platforms being driven by specific business outcomes.

There has been an increasing focus on business operations, risk and growth driving modernization rather than simply cost reduction. The focus on business outcomes enables IT and the business to focus investments where the business outcomes are richest and also prioritize initiatives from a portfolio strategy. The following are typical drivers used by business and IT managers to trigger Natural/Adabas modernization efforts.

Software AG's Shift In Strategy Driving Prohibitive Costs

In a press release Software AG wrote: "Software AG's strategy has been strongly focused on growth in the Business Process Excellence (BPE) business line since 2012. As a consequence, the company heavily invested in new products and the expansion of its sales teams."

This change in focus prompted Software AG to maximize the revenue coming in for Adabas while it still could. Most Adabas customers are large enterprises, so each customer could pay a lot more than a typical Open Systems customer. Prices for Adabas quickly jumped into the hundreds of thousands US dollars.

This short term revenue strategy has been effective. In January 2014, Software AG reported a record level of license sales in the final quarter, with Group license revenue climbing 22 percent. However, the number of Adabas installations has steadily declined, and revenue predictions for the same group in 2015 are 9 to 16% lower.

Resource Risk, Lack of Compatibility With Modern Business Needs

The Natural language and Adabas database have been classified as "legacy systems," so finding skilled resources in these areas is more difficult than ever.

Lines of business demand better reporting, integration with mobile, social, analytics, and cloud platforms. These requests expose the inefficiency of maintaining an Adabas platform, which can no longer compete with open systems in time to development, scalability and cost.

The combination of these factors in addition to the prohibitive expense of Adabas platforms is driving change from the office of the CFO to the CIO.

Companies Are Fighting Back

Nissan has used Software AG's Adabas data management software and Natural programming environment to power its order-management system and other business tools since 1983.

However, Nissan sued Software AG after the German software vendor tried to charge the Japanese automaker more than $3 million for the right to hand its applications over to an outsourcer.

"Software AG recently has engaged in a widespread practice of auditing its licenses and demanding more fees from its licensees who are dependent on the Software AG software for their business operations," Nissan said in the court filing.

Nissan in court papers filed called the move by Software AG a money grab from a vendor that "is not experiencing any significant growth from new products."

Executing The Plan: Best Practices and Risk Factors

Understanding the details of an application and its interrelationships with other systems is a necessary step for modernization. A challenge for many companies is that little documentation exists, or is current, for many applications. The tendency is to move forward without it.

You Don't Know What You Don't Know

Obtaining good code understanding and creating documentation helps organizations make better modernization decisions. Companies should not simply take a symmetric view of the transition. Some parts of the portfolio may be moved with little change. For others, the organization may have to evaluate alternative ways to provide the desired function.

For example, replacing Natural applications that create reports with report-writing packages eliminates the need for code migration or to rewrite the application. Understanding the specifics of these applications enables an organization to plan the migration to a report-writing package.

However, understanding the business rules in the code is necessary to support the modernization initiative. For this code understanding and documentation step, leverage an automated discovery and documentation service like BluePhoenix's Legacy Forensics to quickly and accurately understand your Natural/Adabas ecosystem.

Understand Operational Consequence

Users should experience a "Business As Usual" feeling throughout and after the completion of your project. Any operational impact slows project delivery and creates negative perceptions internally about the project's viability. Maintain an on-going data conversion strategy that allows new features to be built while the system is modernized.

Hitting Your Target

The information obtained during a Portfolio Analysis engagement helps define the complete destination architecture. Organizations may have a predilection toward Java in a Unix or Linux environment, or a Microsoft .NET strategy on Windows. The definition of destination architecture, in line with the organization's strategic enterprise architecture strategy, and a subsequent mapping of the old applications and operational issues to this environment, helps identify gaps in capability.

Performance differences are important considerations. Procedural porting may be less elegant architecturally, but more likely to meet performance needs. Process redesign through Architectural Transformation may be necessary to provide the same apparent user response, despite lower absolute performance levels.

Don't Be Afraid To Ask Hard Questions

As part of this planning step, organizations should ask several questions as they evaluate a migration effort:

- What are the implied aspects of total cost of ownership (TCO) and platform selection, such as backup, recovery, security and availability?
- How can we gracefully migrate, while minimizing the effect on current software contracts?
- What are the effects of such a migration on staffing and skill levels?
- What quality of service can be achieved, and will it be sufficient?
- How long will it take to build the same operational skill levels?
- What are the implications of migrations on record retention and audit implications?

Organizations should consider the difficulty of extricating themselves from potentially decades of architectural dependence. Although many technical differences affect such projects, the impact of change on people and the processes they use to deliver and operate applications is significant. Follow these Best Practices to minimize risk of your transition.

Executive Buy-In

Understanding the reach, risk, and scope of modernizing a legacy platform can involve cross-functional resources. It's important for executives to understand and support the proper use and prioritization of tasks for those impacted by the project. Without executive buy-in, the likelihood of your project falling behind anticipated timelines skyrockets.

Proactive, Participative Management

Mobilize stakeholders during the entire project by setting up interactive means of communication (management committees, newsletters, master plans) as well as during the transition period. Consider rigorous management of the following activities:

- Management committees encouraging synchronization, convergence and communication
- Structured accountability at every decision-making level
- Formal information mechanisms (newsletters, minutes from meetings, progress reports during by project team meetings)
- Recognition of successes by IT resources

This level of activity is often necessary to ensure change strategies (communication, transition and training plans) are implemented according to plan.

Strategic Evaluation Of Legacy Functionality

Proper resources and timelines must be applied to assess the project's impact across the business. Legacy artifacts and logic should be measured for relevance to future business goals and requirements. Business logic and processes relevant to the business should be added to the project plan. Those no longer needed should be refined or removed.

Proof-Driven Vendor Selection

A substantial Proof of Concept is necessary to validate vendor technology. This process also identifies aspects of the project that require internal customer resources.

Committment To Timeline

Change requests during the late stages of your initiative could significantly delay or even derail the project. Escalation management procedures ensure alignment around critical issues, avoiding ambiguity and misalignment during the various implementation phases.

Thoughtful, Exhaustive Testing

Cross-departmental representation is essential in developing test cases/scenarios for integration and system testing. Start testing early and involve end users. This reduces risk by identifying and solving unexpected integration challenges, underestimated efforts and undetected functional gaps.

Committment to Performance

Prioritize performance with attention to redundancy, recovery, and database availability. This focus enables implementation of agile software development and creates opportunities to deliver new, value-added services to the business.

Who's Doing It: Singapore Land Authority

Business Requirements for Integration, Data Sharing and Faster Service Spurs Transition

The Singapore Land Authority (SLA) handles the registration of property transactions, the issue of new title documents for all properties in Singapore. The Singpore Titles Automated Registry System (STARS) captures this information, applies current policy rules, and ensures accurate data for the Land Register, which is guaranteed by the government under the Land Titles Act. If this information is inaccurate, or if the updated policy information is not applied, the government is open to legal and financial liability.

Business Drivers for Project

The STARS system was built in 1995 on OpenVMS (DEC-Alpha), leveraging Natural/Adabas as the programming language and database. The objectives of the system were to:

- Automate several aspects of Title Registration for private properties and public housing to save time and reduce errors of manual entry
- Quickly and efficiently catalogue and report on title data

Maintaining the Speed of Business

However, as requirements evolved, the legacy systems were unable to efficiently support business needs.

Specifically, SLA needed increased system flexibility to:

- Respond to policy changes quickly and uniformly
- Allow new types of property transactions to be added to meet future business needs and data sharing requests across government agencies
- Extend capabilities of search and data filtering

The amount of time it would take to develop, then the cost to maintain these features in the legacy Adabas environment proved to be prohibitive for SLA.

Lastly, the STARS system had accumulated nearly 20 years of data and application customization. It was imperative to translate the appearance and functionality of the system in a manner that minimized user impact.

Project Delivery & Feedback

Critical Project Requirements

To ensure accuracy at all points, the SLA mandated:

- Measuring/optimizing data quality and planning data synchronization
- Mapping of existing data, tasks, and functionality from current systems and database to target platform
- Ensuring accuracy of mapping and translation plan
- Plans for ensuring adherence to strict security policies during migration/refactoring

System performance was a top priority as well. The new system had to perform as well or better than the legacy system. Therefore, the SLA requested comprehensive tests (unit, integration and system testing) conducted under the peak load specified within the application requirements.

Lastly, the code produced in the target state had to be of high quality to support STARS' integration requirements. The data from STARS is integrated into multiple form versions and over 100 external applications and databases.

In all, the following legacy workload was refactored to Java with Oracle Database:

- Over 1M lines of Natural code
- Over 150 DDMs representing more than 70 Adabas files
- 700 DCL programs

Performance

Performance benchmarks were set to ensure the refactored application met the needs of the business. Batch was tuned to execute within the batch window without impacting the operational Service Level Agreement. For Online, a 3-second response time was achieved for 90% of the cumulative transaction volume over SLA business hours (8:00am to 5:00pm); and no more than 5 seconds for 90% of the remaining 10% of transactions.

Custom UI Enhancement

All non-popup Natural maps were included in an HTML page that contained a Top Frame, Left Frame and Body Frame. The non-popup Natural map were included in the Body Frame of the web page. All searches and links were associated to the Top Frame and Left Frame and were coded to invoke a New Browser Window. Searches and links were removed from the Body Frame to ensure the 'conversational' integrity between the Web Browser client and the Web Application running on the Web Application server.

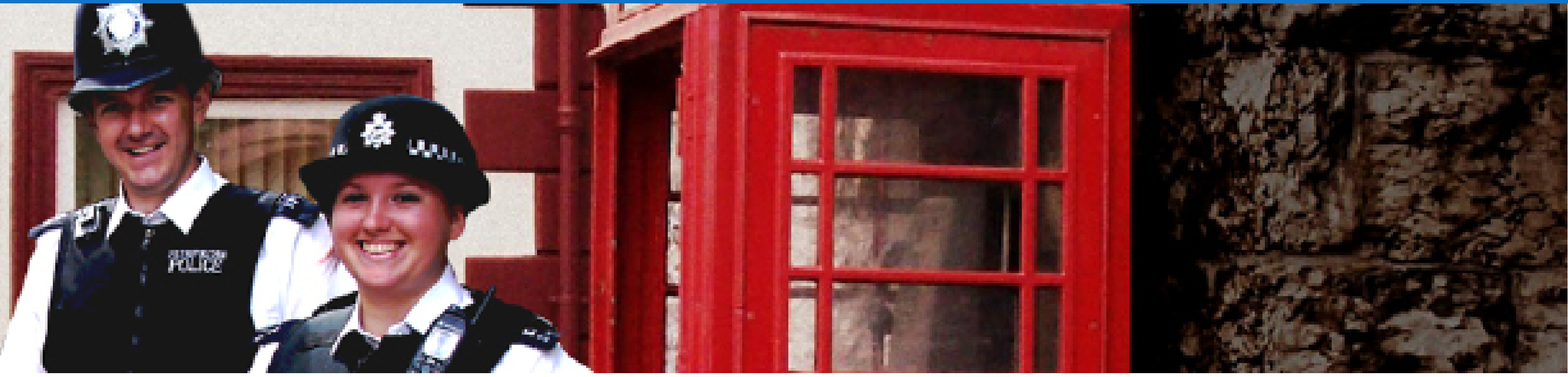
Knowledge Transfer

For optimal value upon delivery, Modern Systems provided Developer Training to the SLA team to ensure proper understanding and usage of the refactored code and supporting framework.

Customer Feedback

"We are pleased with the outcome of our engagement with Modern Systems" says Li Phing. "We were able to achieve our project goals within the timeline and budget specified at the outset of the project."

"Additionally, the refactored code and data tier acted as expected, enabling us to integrate with internal and external systems as needed, at the level of performance we required. The modernized STARS can now deliver data in ways that will provide new value to customers countrywide."



Who’s Doing It: Police Mutual

Business Requirements for Integration, Speed to Service and Microsoft stack drive transition

Police Mutual Assurance Society (PMAS), founded in 1922, offers financial advice and a range of products designed especially for the members of the UK Police services and their families. The ultimate goal of this modernization project was to help over 200,000 police officers receive savings, investments and insurance services quicker.

Business Drivers for Project

PMAS was using a DEC-Alpha (Open VMS) environment to handle critical customer information and share data between different financial platforms. This environment and its surrounding systems became increasingly difficult to support and often lacked the ability to integrate modern systems used to gain insight from customer data.

Modern Systems technology and services were used to migrate the legacy language and database from Natural/Adabas to a new environment leveraging Microsoft SQL Server Enterprise, Windows Server, Visual Studio and Hyper-V.

The Results

David Loughenbury, CIO of Police Mutual, acknowledged the business benefits of moving to the new platform. “The old platform required significant manual work to export data and had no rules or intelligence for automation. The new platform allows us to process and share data between business groups quicker with less risk. This project is the first phase of modernizing our overall infrastructure, reducing operating costs and adding integration with products like Microsoft Dynamics CRM. Now, our services and marketing teams can leverage data to do what we do best- know our customers and their needs.”

The year-long project translating over 1 million lines of Natural and Adabas data was not without its challenges.

“The DEC-Alpha environment came with issues like file versioning and dynamically submitted Natural code for the DCL. Luckily, the source environment is ASCII like Windows so we were able to minimize codepage challenges”, says John Regan, VP of Delivery at Modern Systems.

Adds Loughenbury, “We knew there would be some difficulty with the legacy environment, but we selected Modern Systems because of their significant experience, proven tools, and on-shore support. Modern Systems has worked closely and reliably with our teams from the inception of the project through the sizing and planning, right through to the delivery. The project has been completed within timescale and budget and this enables us to move onto the next stage of our IT strategy. The services provided by Modern Systems really accelerated the migration timeframe and de-risked delivery.”

How It's Done: Modern Systems Solution Details

Our solution delivers MORE than just a one to one conversion. Customers receive a fundamentally optimized application, infrastructure, and business function delivery vehicle. The following breaks down the technical details of our solution, followed by the applicable platforms for delivery.

Principal Solution Benefits

Modern Systems leverages automated data collection and code refactoring technology. Our solution guarantees a functional match to the source environment while generating fully maintainable Java or C# code. We convert Adabas databases directly to DB2, Oracle or SQL Server. Specific benefits include:

- **Faster time to production** for modernized applications
- **Reduced cost and complexity** of migration engagement
- **Minimized impact on users** and business units

The maturity of our solution enables us to optionally deliver seamlessly though partners, who often use us to either complete or enable the scope of their core services.

Portfolio Analysis

It's necessary to have a complete, holistic analysis of the legacy system to ensure accurate results. Report output represents the legacy environment, covering technical inventory, business logic and project risk factors. This output can also reduce maintenance costs of the legacy environment by mapping out functional code and identifying "dead code". The output of this stage typically includes:

- Documentation of embedded languages, databases, 3rd party calls, etc
- Discovery of business rules, facilitation of business logic extraction
- Dead code identification and removal Excess and/or problematic inventory identification and removal
- Project Plan: timeline, duties, processes for error handling, and change control

Target System Build, Data Migration

Once inventory and legacy functionality is validated, the target environment is built and tested. Upon acceptance, Modern Systems began code refactoring and migration. The following process is automated and executed using Modern Systems technology:

- Generate Natural unload programs to run against Adabas
- Resulting data is transferred to the target environment
- Generate programs to transform the data
- Generate loader utility, load the data to target
- Validate the data migration has been successful using check-sums and hash algorithms
- Full comparison of data from Adabas vs target
- Perform pre-delivery testing on every Adabas file that will be migrated at Go-live
- Add new tables and columns required as needed to target database
- Preserve archived data

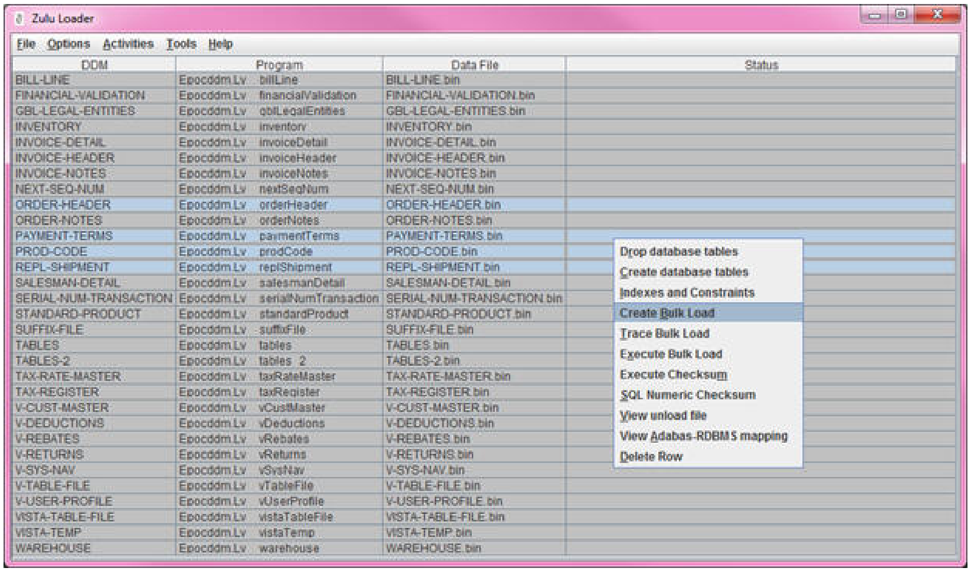
How It's Done: Modern Systems Solution Details

Code Refactoring

Natural code is refactored to Java or C# using our automated solution. Refactored programs can use a mix of Adabas and relational data facilitating gradual transition. Adabas-relational database synchronization reads Adabas PLOG and updates the converted database, to facilitate parallel execution during transitioning. The logic of the application and presentation layer are refactored directly, then refined to meet requirements articulated specifically for the target state.

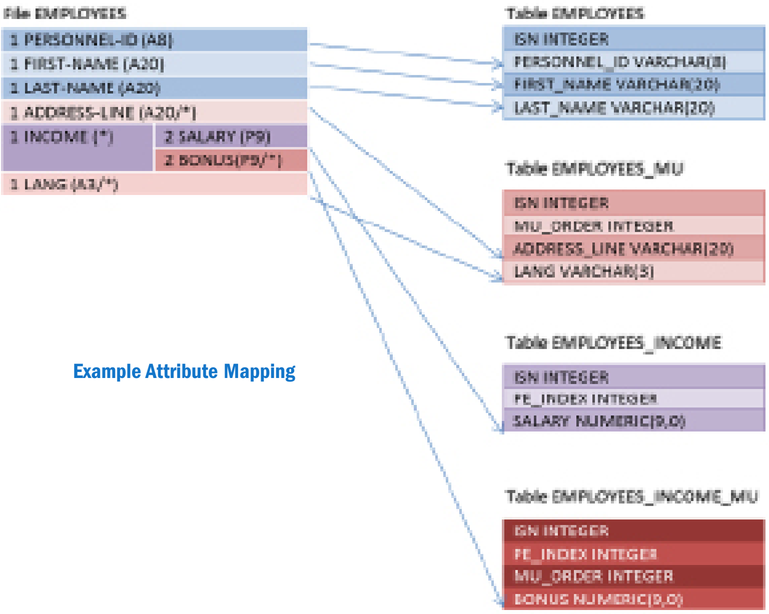
Testing and Delivery

A mutually acceptable Severity Index and Response matrix is created to classify and address bugs found in testing. Modern Systems works together with customers to validate unit tests and functionality of all refactored code.



DDM	Program	Data File	Status
BILL-LINE	Epocddm.Lv billLine	BILL-LINE.bin	
FINANCIAL-VALIDATION	Epocddm.Lv financialValidation	FINANCIAL-VALIDATION.bin	
GBL-LEGAL-ENTITIES	Epocddm.Lv gblLegalEntities	GBL-LEGAL-ENTITIES.bin	
INVENTORY	Epocddm.Lv inventory	INVENTORY.bin	
INVOICE-DETAIL	Epocddm.Lv invoiceDetail	INVOICE-DETAIL.bin	
INVOICE-HEADER	Epocddm.Lv invoiceHeader	INVOICE-HEADER.bin	
INVOICE-NOTES	Epocddm.Lv invoiceNotes	INVOICE-NOTES.bin	
NEXT-SEQ-NUM	Epocddm.Lv nextSeqNum	NEXT-SEQ-NUM.bin	
ORDER-HEADER	Epocddm.Lv orderHeader	ORDER-HEADER.bin	
ORDER-NOTES	Epocddm.Lv orderNotes	ORDER-NOTES.bin	
PAYMENT-TERMS	Epocddm.Lv paymentTerms	PAYMENT-TERMS.bin	
PROD-CODE	Epocddm.Lv prodCode	PROD-CODE.bin	
REPL-SHIPMENT	Epocddm.Lv replShipment	REPL-SHIPMENT.bin	
SALESMAN-DETAIL	Epocddm.Lv salesmanDetail	SALESMAN-DETAIL.bin	
SERIAL-NUM-TRANSACTION	Epocddm.Lv serialNumTransaction	SERIAL-NUM-TRANSACTION.bin	
STANDARD-PRODUCT	Epocddm.Lv standardProduct	STANDARD-PRODUCT.bin	
SUFFIX-FILE	Epocddm.Lv suffixFile	SUFFIX-FILE.bin	
TABLES	Epocddm.Lv tables	TABLES.bin	
TABLES-2	Epocddm.Lv tables 2	TABLES-2.bin	
TAX-RATE-MASTER	Epocddm.Lv taxRateMaster	TAX-RATE-MASTER.bin	
TAX-REGISTER	Epocddm.Lv taxRegister	TAX-REGISTER.bin	
V-CUST-MASTER	Epocddm.Lv vCustMaster	V-CUST-MASTER.bin	
V-DEDUCTIONS	Epocddm.Lv vDeductions	V-DEDUCTIONS.bin	
V-REBATES	Epocddm.Lv vRebates	V-REBATES.bin	
V-RETURNS	Epocddm.Lv vReturns	V-RETURNS.bin	
V-SYS-NAV	Epocddm.Lv vSysNav	V-SYS-NAV.bin	
V-TABLE-FILE	Epocddm.Lv vTableFile	V-TABLE-FILE.bin	
V-USER-PROFILE	Epocddm.Lv vUserProfile	V-USER-PROFILE.bin	
VISTA-TABLE-FILE	Epocddm.Lv vistaTableFile	VISTA-TABLE-FILE.bin	
VISTA-TEMP	Epocddm.Lv vistaTemp	VISTA-TEMP.bin	
WAREHOUSE	Epocddm.Lv warehouse	WAREHOUSE.bin	

Example Screen for Migrator



Example Attribute Mapping

How It's Done: Modern Systems Solution Details

Target Environment Services

Ultimately, your modernization project depends on how it works in production. However, few companies have resources with equal understanding of Natural/Adabas and target states like Java, C#, SQL Server, DB2 or Oracle.

Modern Systems works as a connector between application and infrastructure teams, interpreting the conversion data to help meet requirements for target system performance. Our expertise on both sides of the project can help ensure ongoing success. Our post-production services include:

- Target Environment Architecture
- Target Environment Deployment
- Ongoing Application Maintenance

Target Environment Architecture and Deployment

Natural/Adabas requirements for performance don't always translate directly to modern infrastructure. We can ensure your design has the proper sizing and resources necessary for success in production. We work with several enterprise infrastructure providers, leveraging best practices and resources. Our supported infrastructure platforms include:

- HP
- IBM Z-Series, SoftLayer
- OpenStack
- EMC
- Amazon Web Services
- Microsoft Azure

Ongoing Application Maintenance and Management

Once the refactored application goes into production, many companies prefer to focus their best technical resources on augmenting the refactored application, increasing its value to the business. Given our understanding of the application's requirements and usage patterns, companies often hire Modern Systems for application maintenance and management, enabling their IT teams to tackle tasks that add more value to the business quicker.