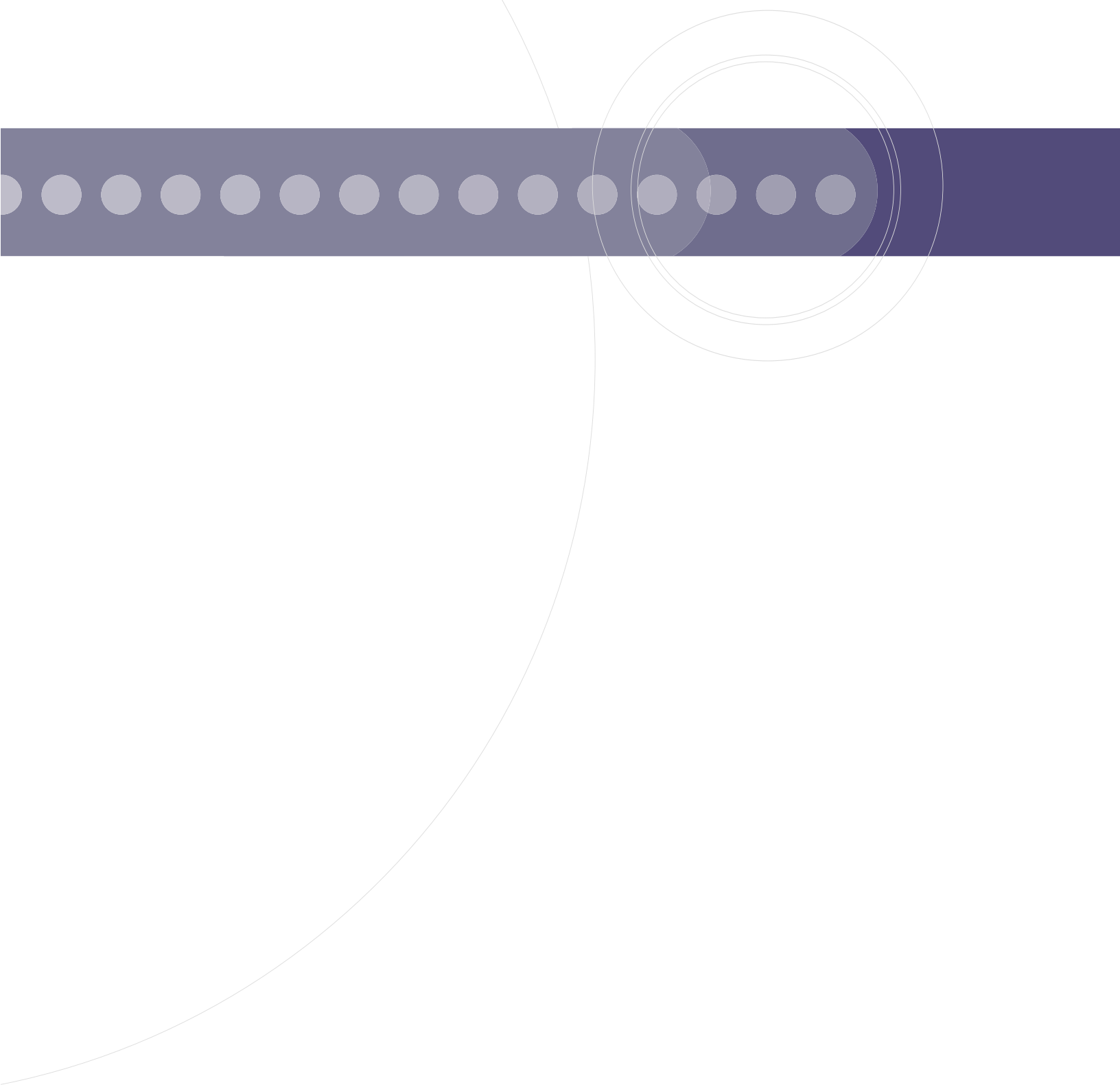


The Complexity Advantage:

Solving the Requirements of Complex Data Transformation in Business Integration





This document contains Confidential, Proprietary and Trade Secret Information ("Confidential Information") of Informatica Corporation and may not be copied, distributed, duplicated, or otherwise reproduced in any manner without the prior written consent of Informatica.

While every attempt has been made to ensure that the information in this document is accurate and complete, some typographical errors or technical inaccuracies may exist. Informatica does not accept responsibility for any kind of loss resulting from the use of information contained in this document. The information contained in this document is subject to change without notice.

The incorporation of the product attributes discussed in these materials into any release or upgrade of any Informatica software product—as well as the timing of any such release or upgrade—is at the sole discretion of Informatica.

Protected by one or more of the following U.S. Patents: 6,032,158; 5,794,246; 6,014,670; 6,339,775; 6,044,374; 6,208,990; 6,208,990; 6,850,947; 6,895,471; or by the following pending U.S. Patents: 09/644,280; 10/966,046; 10/727,700.

This edition published December 2006

Table of Contents

Executive Summary	2
Introduction	3
Types of Complex Data	4
Complex Structured Data	4
Semi-Structured Data	6
Unstructured Data	7
Business Integration Techniques	7
Traditional Middleware	8
Modern Middleware	9
Enterprise Service Buses	10
Adapters	10
Next-Generation Data Transformation	11
Next-Generation Data Transformation in Financial Services	16
Conclusion	18



Executive Summary

Complex data at best is a significant cost factor in business integration—at worst it can cripple a business integration initiative. Complex data is created by an ever-growing array of systems, including legacy systems, modern-day business-to-business (B2B) transaction systems, middleware, and personal productivity software such as Microsoft® Office applications. In today's IT environment, complex data represents an ever increasing proportion of enterprise data.

This paper describes the types of complex data in business integration, and the problems and costs associated with attempting to use current generation business integration techniques to transform and integrate complex data.

The paper goes on to describe the technical and usability requirements for a solution to the problem of complex data transformation in business integration that complements existing IT investments. It provides an example of such a solution, and presents three generic case studies of enterprises that have solved the problem of complex data transformation to gain competitive advantage and high return on investment (ROI) from business integration in dramatically shortened timescales.

For many enterprises, integrating complex data remains an error prone and costly process that requires IT development. Today's business integration technologies and approaches cannot automate the transformation of complex data. Moreover, today's business integration techniques do not support the integration of complex data effectively because they:

- Do not support the broad and diverse set of data formats required by enterprises
- Create/require data transformation logic that is fragmented across many technologies and approaches
- Do not support the transformation of many complex data types

Next-generation data transformation solutions support a broad and diverse set of data formats, integrate data transformations across many technologies and automate the transformation of complex data. By leveraging next-generation data transformation in business integration, enterprises can effectively enable their core real-time enterprise computing initiatives and achieve the “complexity advantage”¹ in business integration.

¹ The term complexity advantage was borrowed from the business book title “The Complexity Advantage, How the Science of Complexity Can Help Your Business Achieve Peak Performance” by Susanne Kelly and Mary Anne Allison (McGraw Hill Professional Book Group 1999).

Introduction

The fundamental problem in business integration is reconciling related data or messages that exist in multiple application systems; differences in the expression or model of that data—i.e., the semantics (meaning), as well as the syntax (format) of the data—prevent its use in a consistent, enterprise wide fashion. The role of data transformation in business integration is to provide mappings and translation for these inconsistent data expressions and models and to allow sharing of such data among the various applications.

This paper addresses the increasing importance of complex data in business integration and the shortcomings of today's business integration techniques in dealing with it. The paper presents a next-generation solution to the problem of complex data transformation in business integration.

We define complex data to include:

- Complex structured data, which is data that is well-defined by metadata, such as data that is represented in eXtensible Markup Language (XML) documents with deeply hierarchical and recursive structures.
- Semi-structured data, which is data with only partial metadata to describe it—for example, data defined in COBOL copybooks and via vertical industry “standards” such as Electronic Data Interchange-X.12 (EDI-X.12), Health Level 7 (HL7), and many others. (We use quotation marks around “standard,” because many user implementations of such standards render the associated metadata invalid.)
- Unstructured data, which contains essentially no metadata to describe it—for example binary files, spreadsheets, documents, and print streams.

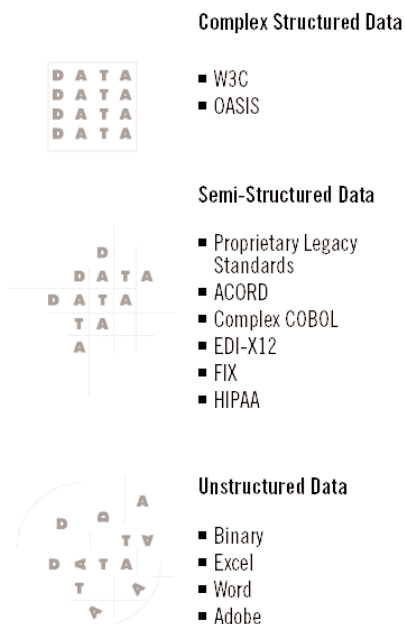



Figure 1: Types of Complex Data



Complex data is inundating today's enterprise and is proliferating rapidly in volume and diversity. Complex data contains increasingly important, and at the same time difficult to access, high-value business information. For example, a high percentage of "material" corporate financial information is maintained in complex data formats, such as Excel spreadsheets. The increase in the accumulation and importance of complex data is in many cases impacted by government regulations that control its lifecycle and may prohibit its deletion.

The ability to transform complex data is critical in today's enterprise. Business drivers, such as regulatory compliance (e.g., Sarbanes-Oxley) and operational efficiency (e.g., the "real-time enterprise") require high-performance, mission critical applications that exploit real-time data transformations between a variety of complex data formats.

The requirements to transform complex data for integration into core business systems and processes go far beyond the capabilities of existing business integration technologies and approaches, in part because some of the business drivers are new (e.g., compliance regulations that govern the life cycle of unstructured data), but also because the problem of complex data transformation is a difficult one. Business integration solutions have been available for more than a decade; if the problem of complex data transformation had been easy to solve, every integration vendor would provide a solution. Since this is not the case, enterprises have to look elsewhere: typically to the costly IT development of integration code.

Current business integration technologies and approaches do not provide a complete integration solution for three reasons:

1. The sheer diversity of data formats: supporting the variety of data formats shared between applications, business processes, or trading partners using today's technology makes business integration extremely expensive.
2. **Data transformation functions are fragmented across many technologies:** point solutions may embed data transformation logic in many different places, e.g., in brokers, adapters, extract transform-load (ETL) tools, and most importantly, in expensive and error-prone custom integration code. This means that even successful data transformation efforts may not be reusable or maintainable.
3. **Lack of support for transformation of complex data:** 75% or more of the business data that needs to be integrated is not well-structured; important business information contained within such data formats is beyond the reach of today's business integration techniques.

Finally, software licenses account for only between 20% and 30% of the total cost for a typical business integration project; data transformation costs account for a disproportionate share of the remainder. This share grows rapidly as the amount and types of complex data increase.

Types of Complex Data

As described in the introduction, complex data in the enterprise can be divided into three broad categories: complex structured, semi-structured, and unstructured.

The following sections describe these types of complex data in business integration and provide context for a deeper discussion of the problems associated with complex data transformation.

Complex Structured Data

Structured data is well defined, typically by way of record definitions, database-schemas or some other form of metadata. This precise data definition allows business applications to process such data with relative ease.

Programming languages have always provided capabilities for defining and manipulating data, and the business value of corporate data includes both the data itself, and its associated data definitions.

The first major effort by IT vendors to structure and “manage” enterprise data centered on databases. The relational data model is the clear winner in database technology—it “normalizes” data and stores it in simple tables. However, the need to share data between disparate applications and systems across the network and the Internet has pushed relational database technology beyond its limits—and resulted in the need for a widely adopted vendor-independent and highly flexible data definition ‘language’ for use in integrated, networked applications and the Internet.


XML is that highly flexible data definition language. XML, and the HyperText Markup Language (HTML), are both derived from the Standard Generalized Markup Language (SGML). XML is designed specifically for Web documents and allows designers to create custom tags that control the definition, transmission, validation, and interpretation of data. As we will see, many of the business integration challenges involving complex structured data center on the transformation of business data to and from XML.

XML’s phenomenal success in the marketplace has resulted in a broad set of extensions and enhancements, including various cross-industry and industry-specific XML standards that are being created with regularity. However, this success comes with a price of data transformation complexity in several common business integration scenarios.

1. Transformation of complex legacy formats to XML: Many XML standards, when implemented, produce exceptionally large and complex document structures. For example, the Association for Cooperative Operations Research and Development (ACORD) XML standard develops and maintains various electronic standards for the insurance, reinsurance and related financial services industries. The ACORD XMLife Business Message/Transaction Specification contains more than 10,000 elements and attributes, and numerous recursive sub-structures. Insurance companies wishing to leverage this standard to realize operational efficiencies must map legacy data structures, often described as complex COBOL copybooks, into the ACORD schema.

2. Transformation of complex legacy formats to Web services: XML’s cross-application compatibility hits a stumbling block in situations where bandwidth is an issue. For example, in a Telco switching fabric, XML presents too great of a performance hit, transmitting up to 10 times as much data as a non-XML data transfer. Telecommunications carriers and their customers and partners can only take advantage of XML Web services at the “edge” of the network where more lenient performance requirements permit its use. This imposes a complex data transformation challenge in mapping between custom wire formats within the switching fabric and the XML-based Web services. Similar issues exist in financial services, defense, and the transportation industry where real-time data distribution requirements abound.

3. Transformation between different versions of complex XML standards: There is an explosion in XML-based vertical standards as each industry develops more precise tagging instructions to assure more complete communications among Web services throughout the supply chain. Application providers, integration software vendors and enterprise IT must not only build



solutions that adapt rapidly to these new standards, but must deal with the problem of versioning. The versioning problem manifests itself as yet another complex data transformation challenge because standards evolve and change so frequently, enterprises must maintain multiple versions of the same XML standard concurrently. Thus creating many-to-many mapping problems between different versions of the same standard.

4. Transformation of unstructured data that is embedded in XML messages: Vendors are facing up to the problem of “opaque” or non-XML data in Web services messages. While there is an effort under way to standardize the way XML works with opaque data, currently there are five different approaches under consideration, each with its own strengths and weaknesses. Regardless of which of these standards is ultimately adopted, the challenge of dealing with opaque attachments is fundamentally a complex data transformation problem.

XML's strength is its ubiquity. However if the most important issue for an application is performance, or if it involves a complex, multiple-industry software product, or if an application's usefulness depends on opaque data, then XML's pervasiveness won't solve the problem of complex data in business integration. In fact, it may further complicate it. Moreover, the promise of Web services is that it will remove most of the stumbling blocks that exist between vendors' software. Yet the most basic building block of Web services, XML, continues to trip up vendors and users and creates a complex data transformation problem for business integration.

Semi-Structured Data

We defined semi-structured data as data that suffers from only partial metadata to describe it. The most prevalent types of semi-structured data in the enterprise today are the myriad customer-specific EDI implementations and legacy COBOL-derived data². The primary use and critical importance of such data has been in legacy transaction system integration with modern Web-based applications, and legacy B2B applications.

EDI has been in use since the 60's and is used by more than 95% of the Fortune 1000. However, it is expensive and cumbersome to implement and maintain, and for those reasons is used by only 2% of all other companies. Hence, a common business integration scenario is the evolution from traditional EDI standards to XML, Web services, and other Internet-based technologies, which are much less expensive to acquire and easier to implement. This migration and the relationship of many EDI systems to legacy COBOL-based systems cause three distinct complex data transformation related problems for enterprises:

1. Transformation of EDI variants to their new XML standards-based counterparts: Even though there may be a specification or standard in place, business users tend to modify the specification during initial implementation and may continue to modify it as they maintain their applications. This practice increases implementation costs, limits flexibility, inhibits reuse and creates long-term maintenance problems. Moreover, this practice creates complex data transformation requirements when migrating from EDI to XML.

2. Transformation of XML into a form usable by legacy applications, many of which reside on legacy mainframe systems: Many EDI systems require integration between the EDI format and a legacy format such as COBOL. The same systems must now support the addition of complex XML. In many cases the additional step of transforming the legacy data into XML must be supported while IT continues to maintain the EDI-to-legacy mappings. This is because EDI is not going away entirely and must now coexist with XML.

² Legacy COBOL-derived data is a perfect example of the prevalence of semi-structured data in the enterprise, thanks to COBOL programming constructs such as REDEFINES, OCCURS, and COMPs.

3. Maintenance of the proliferation of point data transformation solutions: The explosion of EDI and XML implementations cited above has led to a proliferation of point solutions for the transformation of semi-structured data in the enterprise. This in turn has led to the fragmentation of data transformation logic and functionality, which for many enterprises has created a substantial maintenance burden.

Unstructured Data

We defined unstructured data as data with no metadata to describe it. Unstructured data may be produced in a variety of ways—for example, Microsoft Office documents (Excel®, Word®, and PowerPoint® files), Adobe® PDF files, print streams, and any number of legacy formats are unstructured. While such unstructured data has been part of the enterprise for decades, the business value of this data has recently increased markedly in several ways:

- 1. More and more corporate financial information is stored in unstructured documents such as Excel spreadsheets:** this data is typically not available to IT financial reporting systems and can not be examined and reported on in a timely manner as required by regulatory compliance.
- 2. A great deal of valuable corporate data is available only in unstructured legacy formats:** such data exists in the unstructured output of business systems—for example, the customer billing information in print streams.
- 3. The important trend of content³ convergence:** this trend is manifested in the desire to reach unstructured resources and pull them closer to business processes. Business integration solutions need to fully understand the complete business process model to allow relevant content (transactional, compliance, intelligence, and other information) to arrive in applications integrated correctly.

While the business value of this complex data has steadily increased, mechanisms for integrating the data into core business systems have not kept pace. Custom integration code is the mechanism that an enterprise typically uses to transform unstructured data. This approach is used whenever they require specialized transformation logic that their integration platform does not provide out-of-the box, when input formats change often, or when business partners implement a standard loosely.

Without such IT development, the best that can be done with unstructured data is simple searching. This adds the burden of creating and maintaining custom data transformation code. This maintenance is an ongoing problem due to frequent changes in data formats.

Business Integration Techniques

Business integration began as the solution to a problem of making disparate computers in a heterogeneous computing environment communicate with one another. A complete discussion of business integration technologies and approaches in use today is beyond the scope of this paper—but it is important to understand how they have resulted in business integration techniques that have helped exacerbate the complex data transformation problems we face today.

³ The definition of content includes paper records, faxes, electronic documents, Web pages, email, and multimedia objects, which are all pieces of information “content” and are collected, derived, and generated as part of business processes, namely unstructured data.

Three categories of current business integration techniques will be discussed: those using traditional middleware; those using modern middleware; and those using adapters. Adapters warrant a separate category because they span the other categories in their usage. It will become clear that the business integration techniques in use today fail to address the requirements of data format diversity, fragmented data transformation logic, and the lack of support for transformation of complex data.

Traditional Middleware

The first approaches to business integration were early implementations of a software category called middleware. Middleware lies between the operating system and the application code, and provides a distributed platform—or abstraction layer—that hides the inconsistencies between underlying hardware and/or software architectures on different machines. Middleware provides these higher-level abstractions and application services utilizing a range of different styles and technology components.

The following table lists various traditional middleware styles, technologies and their shortcomings when it comes to solving the problem of complex data transformation.

Middleware Style	Description	Complex Data Transformation Support
Remote Procedure Call (RPC)	RPC is a synchronous protocol (i.e., the client waits for a response) that treats a remote function call as though it were a local subroutine.	RPC provides essentially no support for application level complex data transformation.
Message-Oriented Middleware (MOM)	MOM is an asynchronous protocol that supports event driven architecture, a variety of message broker and integration broker products, and a communication paradigm known as publish/subscribe.	MOM systems do not directly address the problem of complex data transformation. But many of these systems rely on expensive to maintain custom integration code to solve the problem.
Transaction Processing Middleware (TP)	TP middleware addresses some of the problems (e.g., data consistency, timing and communication errors) of distributing a formerly non-distributed application.	Distributed TP systems such as CICS from IBM and BEA's Tuxedo are still in wide use today, and these systems do not support complex data transformation out-of-the-box. Again, these systems promote the development of costly custom integration code to solve the problem.


Object-Request Brokers (ORBs)	ORBs represent the combination of RPC-based systems with the concepts of object technology. Most ORBs are based on the Object Management Group's Common Object Request Broker Architecture (CORBA).	In these systems, transformation of complex data is typically hand-coded, and has traditionally been achieved only through costly "black belt" programming.
Application Servers	The most common form of application server is based on Java 2 Enterprise Edition (J2EE) specifications. J2EE app servers are used primarily for new program development, but also provide integration mechanisms, such as those specified by the J2EE Connector Architecture (J2EE CA).	Unfortunately, in a pure app server environment, tool and runtime support for automating any type of data transformation is non-existent.
Enterprise Application Integration (EAI) Brokers and Message Brokers	The fundamental job of an EAI or message broker is to transform data from one application-specific format to the format needed by a different application, and to route the data between application systems, either within an enterprise or between trading partners (as in the case of B2B integration).	Support for the automation of complex data transformation varies in brokers. However, due to the limitations of current generation data mapping and translation tools, integration of complex data in the broker environment requires a substantial manual programming effort.
Business Process Management Systems (BPM)	BPM involves the definition of the steps of a business process, and the orchestration of various applications and sub-processes to manage the execution of those steps.	BPM systems do not directly address the problem of complex data transformation.

Modern Middleware

Web Services Integration

A Web service can be broadly defined as a software system that is identified by a Universal Resource Identifier, whose interfaces and bindings are defined and described using an XML-based mechanism called the Web Services Definition Language (WSDL), and whose messages are encoded using the Simple Object Access Protocol (SOAP) and delivered over a transport protocol such as HTTP ⁴.

⁴ In fact, Web services can use a far broader set of protocols, transports, and formats than this description might lead one to believe. WSDL supports the definition of services in terms of operations and messages, which are described abstractly and then associated with a specific network protocol via a binding, to define a Web services endpoint.



From the standpoint of complex data transformation, there are three important points to take away from our examination of Web services:

- First, Web services per se do nothing to solve the problem of data transformation. SOAP supports the attachment of opaque data payloads of virtually any format, but the need to transform such data remains.
- Second, the ubiquity of Web services and associated technologies means that enterprise systems and architectures are becoming increasingly XML-centric, which requires out-of-the-box XML support for data transformation solutions. However the technology that underlies today's solutions—eXtensible Stylesheet Language Transformation (XSLT)—does not support complex data.
- Third, and perhaps most important, the drive toward Web services-based integration requires the “re-articulation” of legacy interfaces as XML. Accomplishing this requires the “mining” of those interfaces, in the sense that an interface's data parameters must have associated transformation maps and runtimes to turn them into XML. Many legacy interfaces contain complex data, so the push toward Web services integration will likely create an even greater need for a clean solution to the problem of complex data transformation.

Enterprise Service Buses

The Enterprise Service Bus (ESB) is the newest category of enterprise infrastructure software. Because it is new, the content and characteristics of vendors' ESBs vary somewhat. In general, ESBs attempt to provide a service-oriented, high-performance, standards-based business integration infrastructure.

While ESB data transformation capabilities go significantly beyond those of mapping disparate XML schemas, they share the same shortcomings as traditional middleware when dealing with complex data:

- Complex data transformation cannot be accomplished with the off-the-shelf adapters that most ESBs' vendors make available with their products. ESBs still require the development and maintenance of custom data transformation code.
- ESBs are optimized for XML-oriented data transformation and rely on XSLT-based tooling for defining transformations. This code is difficult if not impossible to write in XSLT, which has limited support for complex data types.

Informatica Corporation delivers data integration software and services to solve a problem facing most large organizations: the fragmentation of data across disparate systems. Informatica helps organizations gain greater business value from their information assets by integrating their enterprise data. Informatica's open, platform-neutral software reduces costs, speeds time to results, and scales to handle data integration projects of any size or complexity. With a proven track record of success that extends back to 1993, Informatica helps companies and government organizations of all sizes realize the full business potential of their enterprise data. That's why Informatica is known as the data integration company.

Adapters

Adapters come in several forms, most notably technology adapters (used, for example, to transform a message from one transport protocol and/or data format to another), and application adapters (used, for example, to provide API translation from a commercial off-the-shelf ERP or CRM package, typically to an integration platform.)

Adapters obtain data from a source and deliver it to a target, typically transforming some or all of the data along the way. Adapters typically support only unidirectional, proprietary transformation between application APIs or data formats—to the canonical format of the integration broker, or in the case of “open” adapters, to XML.

Technology adapters have a strategic place in the integration landscape for two reasons:

- First, the formats they transform include the myriad of technical protocols, proprietary data formats, and ever-changing vertical standards and site-specific implementations of those standards.
- Second, some technology adapters have “hooks” that allow the insertion of complex data transformation code into the message handling stack. Though this doesn’t solve the problem of having to hand-code such transformations, it does allow a clean architectural approach to complex data transformation ⁵.

The situation is different for application adapters. The use of application adapters as a solution to complex data transformation is a potentially disastrous architectural approach. Inserting complex data transformation code into an application adapter, which typically has no architectural mechanism for customization, will result in “spaghetti” code.

It is widely assumed that the need for application adapters will all but disappear eventually, because packaged application vendors are all moving towards exposing their product interfaces as Web services, and encoding their data in XML. However, the cost of migrating application adapter customizations involving user written data transformation code to new-interfaces could be prohibitive and create a barrier to the adoption of Web services for many organizations.

Next-Generation Data Transformation

“Next-generation data transformation” refers to the data transformation capabilities required to address current and emerging business integration drivers. These drivers—including new regulatory requirements, the desire to create a “real-time enterprise” and the need to build B2B trading communities, among many others—have affected the scope of data that needs to be integrated, and are mandating the transformation of increasing amounts of complex data.

Complex data transformation is not supported by today’s integrated development environments, integration toolsets, or even by higher-level XML standards such as XSLT, as noted earlier. Moreover, XSLT generators and other current transformation tools provide an inadequate metaphor for handling complex structured data. These tools use a “hard mapping” approach which involves the graphical mapping of source and target fields by connecting them visually, using lines or connectors. This metaphor does not scale to very large complex structures such as the ACORD XML example highlighted earlier.

As a result, the function of complex data transformation has become an increasing bottleneck in achieving value and ROI from business integration. The easy integration problems, such as connectivity, have been solved. Some of the more difficult problems have been addressed with adapters, with mixed results. Most of the integration problems that remain have to do with the transformation of complex data. To recap, complex data is:

- Complex structured data—for example, large XML documents with deeply hierarchical and recursive structures.
- Semi-structured data, which is data with only partial metadata to describe it—for example, data defined in COBOL copybooks and in loosely implemented vertical industry “standards” such as EDI-X.12 and HL7.
- Unstructured data, which contains essentially no metadata to describe it—for example binary files, spreadsheets, documents, and print streams.

And as noted earlier, recent standards (mostly XML-based) do not solve the data transformation requirements for business integration in today’s enterprises, and in fact the sheer number of

⁵ Some vendors have expressed this issue. For example, IBM’s WebSphere® Business Integration Data Handler for Complex Data exploits Inform

standards (as well as their unavoidable ambiguity, relatively rapid change, and frequently loose implementations) brings its own set of headaches.

So, what are the technical and usability requirements for a next-generation solution to the problem of complex data transformation? A next generation data transformation solution must allow any form of enterprise data to be transformed from within one comprehensive and user-friendly environment. It must enable real-time execution, eliminate costly IT developments and enhance the usability of existing infrastructure software.

Throughout this paper, we have emphasized three general problems with existing approaches to data transformation: 1) failure to support the diverse set of data formats, 2) fragmented transformation logic due to implementation of point solutions, and 3) the inability to transform complex data without IT development. The table below presents specific requirements categorized to align with these three general problems:

Requirements for Complex Data Transformation

Requirements Addressing Data Format Diversity	
Any-to-any data transformation	Support for any document, record, message or file format, including complex data, without compromising development or runtime flexibility/performance.
Transformation abstraction and flexibility	Support for variant data structures, mapping, and translation logic within a transformation—using a simple and easy to use component model abstraction rather than IT development.
Bi-directional data flow	Support for mediation and transformation of data from its original format into XML, and vice versa (from XML to any data format) with a single, executable transformation definition.
Pre-built transformation functions	Provide out-of-the-box support for a vast range of data manipulation, modification, and transformation functions.
Built-in standards frameworks	Support predefined standards-based frameworks, including pre-built parsers, serializers and XSDs for healthcare, finance, manufacturing and other industries (e.g., HIPAA, HL7, EDI, ACORD, and FIX).
Standards-based	The solution must leverage XML, Java, Web services, X-Path, and XSD data types and extensibility, using JavaScript and allowing the use of XSLT only when appropriate for maximum flexibility and conformance to standards.

Requirements Addressing Fragmented Data Transformation Logic	
Reusable transformation logic	<p>Provide a centralized repository to store and manage all transformation logic (e.g., definitions and runtime execution code) so that once a transformation is defined it becomes available for all applications.</p> <p>Provide an extensible component model for transformation functions in order to ease migration of legacy transformation business rules, and allow for their encapsulation and reuse.</p>
Write transformation once, deploy many times anywhere	<p>Provide a small-footprint, embeddable transformation runtime that is independent of the platform/transport, middleware or hosting/OS environment. Enable many usages of the same transformation and separation of the maintenance of high-value transformation logic from the mechanics of information exchange.</p>
Consistency of information	<p>Allow different projects in different departments to get the same consistent data by reusing transformation logic.</p>
Data transformation abstraction layer	<p>Isolate applications from changes to the underlying data sources, eliminating the need to modify and test hundreds of data interfaces that may be impacted by changes in the schema definition of an underlying data source.</p>



Requirements Addressing the Transformation of Complex Data—Without IT Development	
Data visualization	Enable data architects to visualize any type of unstructured, semi-structured or complex structured data format, dramatically accelerating data interface definition in preparation for mapping and translation.
Example-based mapping and component configuration	Provide both graphical drag-and-drop mapping from a representative data source, and declarative transformation definitions, using sophisticated code generation to help avoid costly IT development.
Parsing-by-Example: Infer-from-sample	Automate discovery/import of document metadata to create mapping and translation logic from an example document, record, message or file format.
Parsing-by-Example: Learn-from-specification	Automate discovery/import of data specifications (such as HL7) to enable the import of a schema, specification or metadata definition, allowing for rapid adoption of vertical industry and proprietary data standards.
Efficient, optimized processing	Support transformations that bypass all but desired data at runtime, to reduce the overhead typically associated with transformation processing.

Informatica is the first software company to deliver next-generation data transformation technology with its Informatica ContentMaster™ product line. Informatica ContentMaster allows any form of enterprise data to be transformed from within one comprehensive environment, eliminating costly IT development and enhancing the usability of existing infrastructure software.

Informatica ContentMaster provides an enterprise-level solution by satisfying all of the requirements listed above. Informatica ContentMaster seamlessly integrates with any type of traditional or modern style of middleware, providing the opportunity to produce sizable productivity and maintenance cost savings across all such middleware.

The following sections present case studies of enterprises in three separate industries that have successfully deployed Informatica ContentMaster to achieve the complexity advantage in business integration by obtaining superior results in business integration projects that involve complex data.

Next-Generation Data Transformation in Healthcare

This case study illustrates how Informatica ContentMaster addresses the problem of data format diversity, by using Parsing-by-Example technology to avoid custom conversion code—and the enormous development effort that would have entailed.

The customer is a health services organization which built one of the largest public health networks in the world and provides comprehensive health insurance, medical care and services to 16 hospitals, more than 1,300 primary and specialized clinics, over 3.7 million patients, and a network of state-of-the-art pharmacies, dental clinics, laboratories, diagnostic imaging facilities, and specialists.

In order to consolidate patient records, the customer had to integrate information from a number of legacy systems and documents, including: over 300 disparate data formats; highly proprietary data used by various hospitals and clinics; complex data produced by legacy applications that could not be changed; and a large variety of document formats used by hospitals and clinics that needed to be integrated.

The customer determined that the best way to ensure a successful EAI implementation was to exploit the power of XML. The challenge was to find a technology that would easily deal with the numerous formats and efficiently transform the data into a pre-defined XML schema.

The customer first tried writing integration code for each proprietary application, but that was shown to be a very costly approach. Building a single converter for any of these hundreds of proprietary systems required weeks of development and hundreds of lines of code, and this approach did not address the need to transform the many different document formats.

Informatica ContentMaster was the only solution with the ability to handle the variety of files and applications, and provide the required performance for a complete data transformation solution.

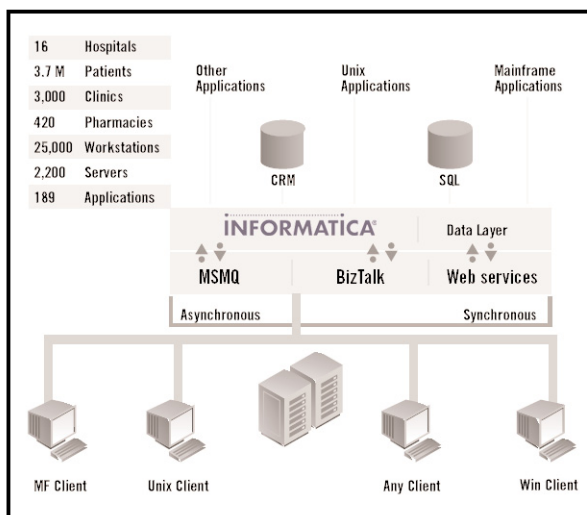


Figure 2: Solution Implementation

Informatica ContentMaster provided the customer with a next generation data transformation solution, rapid ROI, and significant time and cost savings. Because Informatica ContentMaster's patented Parsing-by-Example technology works on any file format, the customer did not have to build a special conversion tool for each format. In addition, no maintenance is required for the parsers. As a result, more than a year of development and code writing were eliminated.

The health services organization's ROI was driven by shorter development cycles—in some instances, the development cycle was shortened from one month to a single day; automated complex data integration resulting in the elimination of months of development and testing; and dramatically simplified maintenance. In addition, the same solution can now be deployed to support any future architectural or platform changes.

The solution has enabled savings of four person years on initial deployment, as well as one person year annually for maintenance. The customer's immediate ROI is 140%, combined with maintenance ROI of 300% annually.

Next-Generation Data Transformation in Financial Services

This case study illustrates how Informatica ContentMaster addresses the problem of fragmented data transformation logic, by allowing a given data transformation solution to be deployed to multiple infrastructure software platforms within the enterprise.

The customer is one of the world's leading financial services organization, one of whose primary value add services is equity research to private clients and market data feeds. End-users rely on information such as equity assessments to make significant buy/sell decisions. Analysts update information daily on the equities they handle and deliver it to market data engines. The timeliness and accuracy of this information is a key competitive edge for the customer.

The business integration scenario was to replace the existing system, and to put in place a solution capable of extracting the structure of, and tracking changes to, documents coming from various data sources. The new system environment combined an integration broker, workflow management system and enterprise content management system.

A key challenge was finding a single solution to complex data transformation that could interoperate with these disparate application and integration infrastructure platforms.

The data transformation challenges involved the extraction and change tracking of a wide variety of formats, including PDF, Word, HTML, Palm OS, Blackberry, and market data feeds. Additionally, approximately 50 feeds, whose formats are updated nightly, had to be concurrently analyzed in near real-time.

Using Informatica ContentMaster, the customer created a solution that allows them to publish their equity research—in many different formats—to end-users in near real-time.

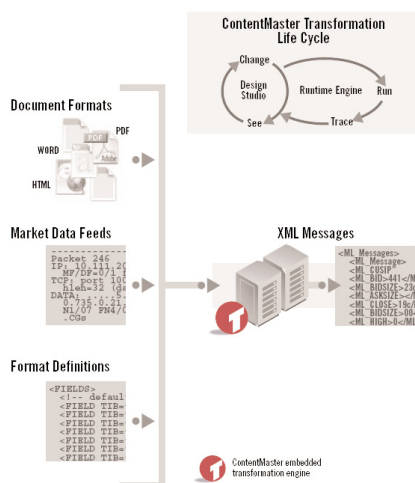


Figure 3: Financial Services Solution

The customer can now generate and deliver equity research data such as Equity Assessments within hours instead of days, across a wide number of delivery channels, resulting in increased quantity and quality of research and higher-value service to their customers.

The Informatica solution went from concept to production in two days per feed, rather than the 3-4 weeks required for custom code. And here again, the customer can deploy the same solution at other points in the enterprise architecture if desired.

This solution also had a significant impact on project ROI and Total Cost of Ownership (TCO). The customer estimated a 300% improvement in project time-to-market and a 500% improvement in maintenance and extensibility over the old system.

Next-Generation Data Transformation in Telecommunications

This case study illustrates how Informatica ContentMaster addresses the problem of having to write custom code for complex data transformation, and also demonstrates how this Informatica ContentMaster capability enhances the value of an in-place integration platform.

The customer is a leading telecommunication company that provides more than \$34.5 billion in voice and data services for both commercial and consumer customers. In order to automate the order management process for services and equipment for its business partners, the customer has built a B2B Gateway. The business case for the gateway had been built around widespread adoption by business partners.

The data transformation challenge was significant in that many business partners, including important wholesale customers, submitted orders and service requests in a variety of formats, including Microsoft Excel and Word, and Adobe PDF. In order to ensure a high rate of adoption, the B2B Gateway had to accommodate these different formats.

Initially, the customer decided to support XML and Excel formats and later extend to others, including Microsoft Word documents and Adobe PDFs. At first it was thought that this capability required the creation and maintenance of custom coded solutions for each format, taking many man weeks per format and adding significant complexity, cost and risk to the project.

Using Informatica's patented Parsing-by-Example technology, Informatica professional services created parsers from multiple document formats into XML—without programming—in just a few weeks. Informatica ContentMaster Studio was used to create transformations for Microsoft Excel and Word, and Adobe PDF, which were embedded into the customer's B2B Gateway. This solution easily delivered the required throughput of 500 documents per day.

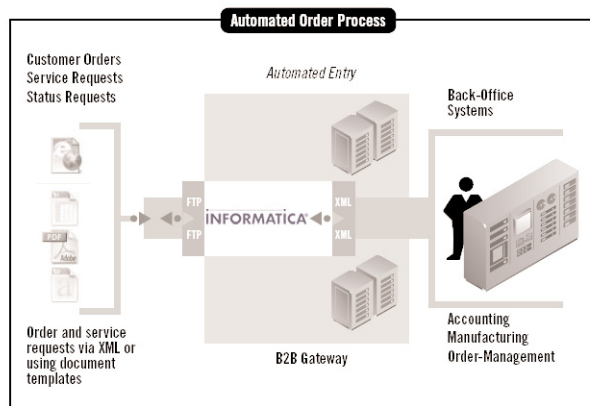


Figure 4: Automated Order Processing Solution



Thus Informatica's technology achieved the seemingly unachievable for the customer—supporting the integration of multiple formats into the B2B gateway in a fraction of the time and cost anticipated for custom development. The Informatica solution also “future-proofed” the system by providing data transformation modules that can be deployed at different or multiple points in the architecture.

Informatica's technology helped deliver business value by broadening customer adoption, accelerating time-to-benefit, and lowering development cost for the B2B Gateway. The

Informatica solution produced 400%+ ROI on the first phase of the B2B Gateway implementation alone saving approximately \$1.5 million in development costs versus the original approach.

Conclusion

This paper has described the types of complex data and business integration techniques. It has presented the case that the major remaining integration problems—in terms of cost and difficulty—have to do with the transformation of complex data.

A viable solution to the problem of complex data transformation must meet three major technical requirements:

- Support for a broad and diverse set of data formats, in order to make business integration less expensive
- Transformations that are integrated across many technologies, in order to make data transformation solutions maintainable and reusable
- Automated support for integration of complex data, so the enterprise can integrate all of its data without resorting to costly, un-maintainable, non-reusable IT development

In terms of usability and maintainability, the solution must not require IT development. Instead, a complete solution will allow transformations to be created, either automatically from a specification (or other metadata), or from a sample document, and/or by way of a point-and-click user interface.

The next-generation data transformation solutions from companies like Informatica, the leader in this emerging enterprise infrastructure software category, meet these requirements, automate the integration of complex data, and allow today's enterprise to achieve the *complexity advantage* in business integration.



INFORMATICA[®]
The Data Integration Company™

Worldwide Headquarters, 100 Cardinal Way, Redwood City, CA 94063, USA
phone: 650.385.5000 fax: 650.385.5500 toll-free in the US: 1.800.653.3871 www.informatica.com

Informatica Offices Around The Globe: Australia • Belgium • Canada • China • France • Germany • Japan • Korea • the Netherlands • Singapore • Switzerland • United Kingdom • USA

© 2006 Informatica Corporation. All rights reserved. Printed in the U.S.A. Informatica, the Informatica logo, and, PowerCenter are trademarks or registered trademarks of Informatica Corporation in the United States and in jurisdictions throughout the world. All other company and product names may be tradenames or trademarks of their respective owners.

J51055 6754 (01/05/07)