Tom Laszewski
Jason Williamson

Prakash Nauduri

Chapter No. 1

"Getting Started with Information Integration"

1
Getting Started with Information Integration

Business change is a constant necessity as a result of increased competition, improved technology, and shifts in consumer patterns. As a result, an enterprise will reorganize, acquire other businesses, create new applications, and downsize others. Throughout these changes, companies are faced with the challenge of efficiently provisioning their resources in response to their business priorities. To deliver data where it is needed, when it is needed, requires sophisticated information integration technologies.

This chapter discusses the basic concepts of information integration and reviews historical approaches to information integration. We will compare data-level integration with process and application integration. This will provide some solid examples for real world decisions, when trying to understand information integration and how this relates to your business and technical initiatives.

This point is often the hard part of any heterogeneous situation. In the latter part of the chapter, you will understand how information integration is used in a **Service Oriented Architecture** (**SOA**) and the impact to a SOA-based system.

Why consider information integration?

> *The useful life of pre-relational mainframe database management system engines is coming to an end because of a diminishing application and skills base, and increasing costs. – Gartner Group*

During the last 30 years, many companies have deployed mission critical applications running various aspects of their business on the legacy systems. Most of these environments have been built around a proprietary database management system running on the mainframe. According to Gartner Group, the installed base of mainframe, Sybase, and some open source databases has been shrinking. There is vendor sponsored market research that shows mainframe database management systems are growing, which, according to Gartner, is due primarily to increased prices from the vendors, currency conversions, and mainframe CPU replacements.

Over the last few years, many companies have been migrating mission critical applications off the mainframe onto open standard **Relational Database Management Systems** (**RDBMS**) such as Oracle for the following reasons:

- **Reducing skill base**: Students and new entrants to the job market are being trained on RDBMS like Oracle and not on the legacy database management systems. Legacy personnel are retiring, and those that are not are moving into expensive consulting positions to arbitrage the demand.
- **Lack of flexibility to meet business requirements**: The world of business is constantly changing and new business requirements like compliance and outsourcing require application changes. Changing the behavior, structure, access, interface or size of old databases is very hard and often not possible, limiting the ability of the IT department to meet the needs of the business. Most applications on the aging platforms are 10 to 30 years old and are long past their original usable lifetime.
- **Lack of Independent Software Vendor** (**ISV**)**applications**: With most ISVs focusing on the larger market, it is very difficult to find applications, infrastructure, and tools for legacy platforms. This requires every application to be custom coded on the closed environment by scarce in-house experts or by expensive outside consultants.
- **Total Cost of Ownership** (**TCO**): As the user base for proprietary systems decreases, hardware, spare parts, and vendor support costs have been increasing. Adding to this are the high costs of changing legacy applications, paid either as consulting fees for a replacement for diminishing numbers of mainframe trained experts or increased salaries for existing personnel. All leading to a very high TCO which doesn't even take into account the opportunity cost to the business of having inflexible systems.

Business challenges in data integration and migration

Once the decision has been taken to migrate away from a legacy environment, the primary business challenge is **business continuity**. Since many of these applications are mission critical, running various aspects of the business, the migration strategy has to ensure continuity to the new application—and in the event of failure, rollback to the mainframe application. This approach requires data in the existing application to be synchronized with data on the new application.

Making the challenge of data migration more complicated is the fact that legacy applications tend to be interdependent, but the need from a risk mitigation standpoint is to move applications one at a time. A follow-on challenge is prioritizing the order in which applications are to be moved off the mainframe, and ensuring that the order meets both the business needs and minimizes the risk in the migration process.

Once a specific application is being migrated, the next challenge is to decide which business processes will be migrated to the new application. Many companies have business processes that are present, because that's the way their systems work. When migrating an application off the mainframe, many business processes do not need to migrate. Even among the business processes that need to be migrated, some of these business processes will need to be moved as-is and some of them will have to be changed. Many companies utilize the opportunity afforded by a migration to redo the business processes they have had to live with for many years.

Data is the foundation of the modernization process. You can move the application, business logic, and work flow, but without a clean migration of the data the business requirements will not be met. A clean data migration involves:

- Data that is organized in a usable format by all modern tools
- Data that is optimized for an Oracle database
- Data that is easy to maintain

Technical challenges of information integration

The technical challenges with any information integration all stem from the fact that the application accesses heterogeneous data (VSAM, IMS, IDMS, ADABAS, DB2, MSSQL, and so on) that can even be in a non-relational hierarchical format. Some of the technical problems include:

- The flexible file definition feature used in COBOL applications in the existing system will have data files with multi-record formats and multi-record types in the same dataset—neither of which exist in RDBMS. Looping data structure and substructure or relative offset record organization such as a linked list, which are difficult to map into a relational table.
- Data and referential integrity is managed by the Oracle database engine. However, legacy applications already have this integrity built in. One question is whether to use Oracle to handle this integrity and remove the logic from the application.
- Finally, creating an Oracle schema to maximize performance, which includes mapping non-oracle keys to Oracle primary and secondary keys; especially when legacy data is organized in order of key value which can affect the performance on an Oracle RDBMS. There are also differences in how some engines process transactions, rollbacks, and record locking.

General approaches to information integration and migration

There are several technical approaches to consider when doing any kind of integration or migration activity. In this section, we will look at a methodology or approach for both data integration and data migration.

Data integration

Clearly, given this range of requirements, there are a variety of different integration strategies, including the following:

- **Consolidated**: A consolidated data integration solution moves all data into a single database and manages it in a central location. There are some considerations that need to be known regarding the differences between non-Oracle and Oracle mechanics. Transaction processing is an example. Some engines use implicit commits and some manage character sets differently than Oracle does, this has an impact on sort order.

- **Federated**: A federated data integration solution leaves data in the individual data source where it is normally maintained and updated, and simply consolidates it on the fly as needed. In this case, multiple data sources will appear to be integrated into a single virtual database, masking the number and different kinds of databases behind the consolidated view. These solutions can work bidirectionally.
- **Shared**: A shared data integration solution actually moves data and events from one or more source databases to a consolidated resource, or queue, created to serve one or more new applications. Data can be maintained and exchanged using technologies such as replication, message queuing, transportable table spaces, and FTP.

Oracle has extensive support for consolidated data integration and while there are many obvious benefits to the consolidated solution, it is not practical for any organization that must deal with legacy systems or integrate with data it does not own. Therefore, we will not discuss this type any further, but instead concentrate on federated and shared solutions.

Data migration

Over 80 percent of migration projects fail or overrun their original budgets/timelines, according to a study by the Standish Group. In most cases, this is because of a lack of understanding of some of the unique challenges of a migration project. The top five challenges of a migration project are:

- **Little migration expertise to draw from**: Migration is not an industry-recognized area of expertise with an established body of knowledge and practices, nor have most companies built up any internal competency to draw from.
- **Insufficient understanding of data and source systems**: The required data is spread across multiple source systems, not in the right format, of poor quality, only accessible through vaguely understood interfaces, and sometimes missing altogether.
- **Continuously evolving target system**: The target system is often under development at the time of data migration, and the requirements often change during the project.
- **Complex target data validations**: Many target systems have restrictions, constraints, and thresholds on the validity, integrity, and quality of the data to be loaded.

- **Repeated synchronization after the initial migration**: Migration is not a one-time effort. Old systems are usually kept alive after new systems launch and synchronization is required between the old and new systems during this handoff period. Also, long after the migration is completed, companies often have to prove the migration was complete and accurate to various government, judicial, and regulatory bodies.

Most migration projects fail because of an inappropriate migration methodology, because the migration problem is thought of as a four stage process:

- Analyze the source data
- Extract/transform the data into the target formats
- Validate and cleanse the data
- Load the data into the target

However, because of the migration challenges discussed previously, this four stage project methodology often fails miserably.

The challenge begins during the initial analysis of the source data when most of the assumptions about the data are proved wrong. Since there is never enough time planned for analysis, any mapping specification from the mainframe to Oracle is effectively an intelligent guess. Based on the initial mapping specification, extractions, and transformations developed run into changing target data requirements, requiring additional analysis and changes to the mapping specification. Validating the data according to various integrity and quality constraints will typically pose a challenge. If the validation fails, the project goes back to further analysis and then further rounds of extractions and transformations. When the data is finally ready to be loaded into Oracle, unexpected data scenarios will often break the loading process and send the project back for more analysis, more extractions and transformations, and more validations. Approaching migration as a four stage process means continually going back to earlier stages due to the five challenges of data migration.

The biggest problem with migration project methodology is that it does not support the iterative nature of migrations. Further complicating the issue is that the technology used for data migration often consists of general-purpose tools repurposed for each of the four project stages. These tools are usually non-integrated and only serve to make difficult processes more difficult on top of a poor methodology.

The ideal model for successfully managing a data migration project is not based on multiple independent tools. Thus, a cohesive method enables you to cycle or spiral your way through the migration process—analyzing the data, extracting and transforming the data, validating the data, and loading it into targets, and repeating the same process until the migration is successfully completed. This approach enables target-driven analysis, validating assumptions, refining designs, and applying best practices as the project progresses. This agile methodology uses the same four stages of analyze, extract/transform, validate and load. However, the four stages are not only iterated, but also interconnected with one another.

An iterative approach is best achieved through a unified toolset, or platform, that leverages automation and provides functionality which spans all four stages. In an iterative process, there is a big difference between using a different tool for each stage and one unified toolset across all four stages. In one unified toolset, the results of one stage can be easily carried into the next, enabling faster, more frequent and ultimately less iteration which is the key to success in a migration project. A single platform not only unifies the development team across the project phases, but also unifies the separate teams that may be handling each different source system in a multi-source migration project. We'll explore a few of these methods in the coming chapters and see where the tools line up.

Architectures: federated versus shared

Federated data integration can be very complicated. This is especially the case for distributed environments where several heterogeneous remote databases are to be synchronized using two-phase commit. Solutions that provide federated data integration access and maintain the data in the place wherever it resides (such as in a mainframe data store associated with legacy applications). Data access is done 'transparently' for example, the user (or application) interacts with a single virtual or federated relational database under the control of the primary RDBMS, such as Oracle. This data integration software is working with the primary RDBMS 'under the covers' to transform and translate schemas, data dictionaries, and dialects of SQL; ensure transactional consistency across remote foreign databases (using two-phase commit); and make the collection of disparate, heterogeneous, distributed data sources appear as one unified database. The integration software carrying out these complex tasks needs to be tightly integrated with the primary RDBMS in order to benefit from built-in functions and effective query optimization. The RDBMS must also provide all the other important RDBMS functions, including effective query optimization.

Data sharing integration

Data sharing-based integration involves the sharing of data, transactions, and events among various applications in an organization. It can be accomplished within seconds or overnight, depending on the requirement. It may be done in incremental steps, over time, as individual one-off implementations are required. If one-off tools are used to implement data sharing, eventually the variety of data-sharing approaches employed begin to conflict, and the IT department becomes overwhelmed with an unmanageable maintenance, which increases the total cost of ownership.

What is needed is a comprehensive, unified approach that relies on a standard set of services to capture, stage, and consume the information being shared. Such an environment needs to include a rules-based engine, support for popular development languages, and comply with open standards. GUI-based tools should be available for ease of development and the inherent capabilities should be modular enough to satisfy a wide variety of possible implementation scenarios.

The data-sharing form of data integration can be applied to achieve near real-time data sharing. While it does not guarantee the level of synchronization inherent with a federated data integration approach (for example, if updates are performed using two-phase commit), it also doesn't incur the corresponding performance overhead. Availability is improved because there are multiple copies of the data.

Considerations when choosing an integration approach

There is a range in the complexity of data integration projects from relatively straightforward (for example, integrating data from two merging companies that used the same Oracle applications) to extremely complex projects such as long-range geographical data replication and multiple database platforms. For each project, the following factors can be assessed to estimate the complexity level. Pretend you are a systems integrator such as EDS trying to size a data integration effort as you prepare a project proposal.

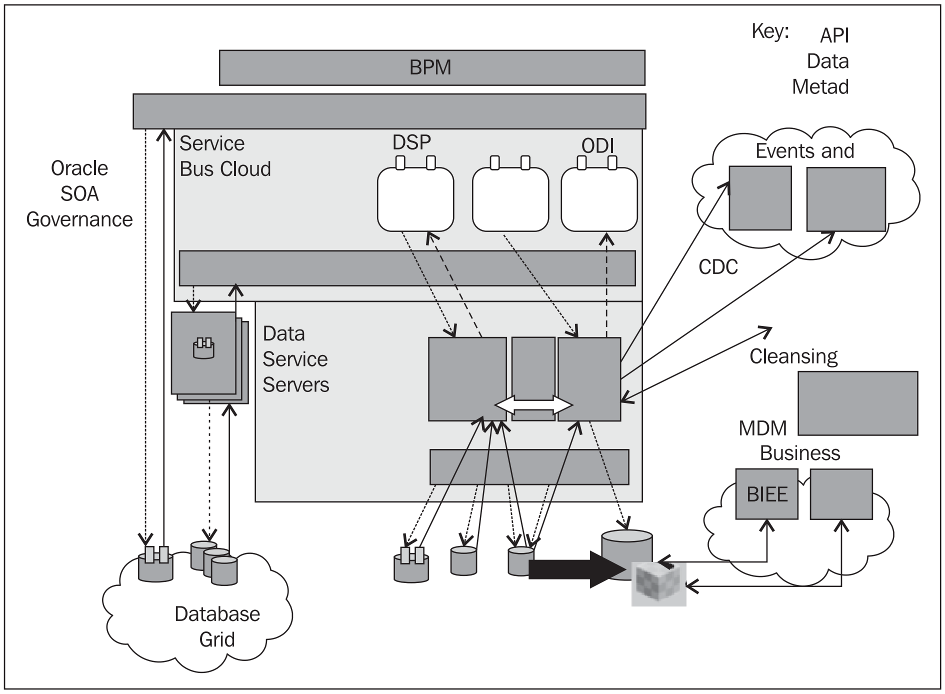
- **Potential for conflicts**: Is the data source updated by more than one application? If so, the potential exists for each application to simultaneously update the same data.
- **Latency**: What is the required synchronization level for the data integration process? Can it be an overnight batch operation like a typical data warehouse? Must it be synchronous, and with two-phase commit? Or, can it be quasi-real-time, where a two or three second lag is tolerable, permitting an asynchronous solution?

- **Transaction volumes and data growth trajectory**: What are the expected average and peak transaction rates and data processing throughput that will be required?
- **Access patterns**: How frequently is the data accessed and from where?
- **Data source size**: Some data sources of such volume that back up, and unavailability becomes extremely important.
- **Application and data source variety**: Are we trying to integrate two ostensibly similar databases following the merger of two companies that both use the same application, or did they each have different applications? Are there multiple data sources that are all relational databases? Or are we integrating data from legacy system files with relational databases and real-time external data feeds?
- **Data quality**: The probability that data quality adds to overall project complexity increases as the variety of data sources increases.

One point of this discussion is that the requirements of data integration projects will vary widely. Therefore, the platform used to address these issues must be a rich superset of the features and functions that will be applied to any one project.

Integration and SOA, bringing it together

We hear from customers over and over again about how difficult it is to add a new interface, support a new customer file, and about the amount of custom code, scripts, and JCL dedicated to simple integration solutions. The outdated proprietary methods of FTP-ing flat files, or calling CICS transactions on another mainframe are replaced by **Enterprise Information Integration** (**EII**) and **Enterprise Application Integration** (EAI) products that are based on standards. EII and EAI tools and technologies give you the capability to create new application interfaces in days instead of months. The following chart shows an example of a reference architecture where integration products are used to bring it all together. Here we will look at the advantages of such an end state.



Architected for the Internet

Technologies used in Legacy SOA Integration can get you on the web, but this does not mean the core of your application is built for the Internet. In an integrated solution, the architecture is built to support the Internet and SOA technologies. Your application architecture inherently includes HTTP, SOAP, web services, HTML-based reporting, business process flows, portals, and business activity monitoring. In other words, your new open-systems application is built to be truly web-enabled.

Scalability

Scalability is not all about being able to handle additional workload without significant degradation in response time. It is also about the ability to handle periodic workload changes such as end-of-year processing, sales traffic, or the retailer's experience during the holidays. Database and application server grids are the perfect match for scalability. Not only can you scale out (add more databases, application server processes, storage, and hardware), but you can also provision your workload to utilize your hardware and software based on the current environment. So for the month of December, when your retail sales are higher, more machines can be dynamically configured to handle sales transactions. When December 31 rolls around and you need to close your books for the year, your infrastructure can be changed to handle financial and accounting transactions.

Availability

Legacy systems can certainly be called reliable, but availability is a whole new subject. In the age of the Internet, clients expect your systems to be up '24/7', throughout the year. Legacy systems are typically down anywhere from two to twelve hours a night for batch processing. Much of this has to do with the lack of concurrency built into legacy databases. When the legacy applications were developed, applications were not expected to be available all day. Businesses did not operate on a global scale in what is often a 24 hour day. IT systems were architectured to match the business requirements at the time. These systems, however, do not match the business requirements of today.

With the advent of grid computing, open systems infrastructure, and the application and database software that runs on top of it are built to operate 24/7, 365 days a year. Maximum Availability Architectures are commonplace in open systems as businesses expect the system to be 'Always on'.

Greater software options

The major software vendors' product strategies are focused on relational database, SQL, Java, .NET, and open systems hardware platforms. Therefore, the entire ecosystem that exists in software development tools, database and application management tools, ISV applications, commercial of-the-shelf (COTS) applications, and hardware and storage support is in the thousands, rather than dozens or perhaps hundreds. The combination of more options, more competition, and more standards-based technologies lower the acquisition cost, support and maintenance costs, and increase customer service. It is only logical that the open market creates more choices for you at a lower cost with better service.

On-demand reporting

One of the main reasons for the proliferation of spreadsheets, Microsoft Access applications, departmental level systems, and 'the guy in accounting who took a night class on Crystal Reports' creating dozens of reports which potentially have queries that run for hours, is that in the legacy world it took way too long for the IT department to respond to new reporting and business intelligence requests. So the user community took it upon themselves to solve the problem. Departmental databases and spreadsheet-driven applications became part of the corporate fabric, and many companies rely on these systems for mission critical processing such as sales forecasting, inventory control, budgeting, and purchasing. The information is in the legacy system, but it is too difficult to get to, manipulate, and report on. With hundreds of open systems reporting, querying, data mining, and BI tools in the market place, users can access the re-architected relational database themselves. Or the IT department can easily create the reports or the BI system that the user community is asking for.

Security

Security is often seen as the domain of the legacy environment, especially in a mainframe. We hear, "My mainframe is so secure that I am never going to move off". On the other hand, we also hear of companies getting off the mainframe because there is no way to encrypt the data. Relational databases on open systems have built-in security, so IT personnel cannot access data that is not part of their daily job. They also offer transparent data encryption. Remember, most security breaches are made by your own people. This is security of data at rest. Then, there is the security of data on the network, and application-based security. This is where open system options such as network encryption, single sign-on, user ID provisioning, federated identity management, and virtual directories, all make sure open systems are more secure than your legacy environment.

Overcoming barriers to change

You can always tell the folks in the room who are just waiting for retirement and don't want the legacy system to retire before they do. Often, we can hear them say, "*This can't be done, our system is way too complicated, only a mainframe can handle this workload, we have tried this before and it failed*". Then you find out that they are running a 300 MIP mainframe with about one million lines of code and about two gigabytes of data. In some cases, you can handle this processing on a two node dual core processor! Or, you may find out that the system is really just a bunch of flat file interfaces that apply 300 business rules and send transactions out to third parties. This can be re-architected to a modern platform, using technologies such as Extract, Transform, and Load (ETL) that did not exist 20 years ago.

You also have to be careful when re-architecting a legacy system, as the business processes and data entry screens, as well as the people who use them, have been around for decades. You have to balance the amount of technology change with the amount of change your business community can digest. You would think that all companies would want an Internet-based web interface to a re-architected system. However, there was one occasion wherein a re-architecture System Integrator (SI) had to include a third-party screen emulation vendor that actually turned HTML into 3270 'green screens'. This was so the users could have the same look and feel, including PF keys, on the web as they did on their character-based dumb terminals.

One of the most discouraging aspects of my role as a modernization architect is that many companies re-architect legacy systems but continue to custom code application features that can be found in 'off-the-shelf' technology products. This happens often, because they don't know the new technologies properly (or even that they exist), developers still like to code, or they cannot change their mindset from 'not invented here' (meaning that we know the best way to do this).

Custom integration applications and utilities

With all the EII and EAI technologies in the market place, we still see modern-day architects decide that they can write their own integration software or write it better than a vendor who has spent years developing the solution. Initial cost (or sticker shock, some may say) is another reason. The client looks at the initial cost only, and not at the cost of maintenance, adding new interfaces, and of supporting another in-house software application. Looking at the total cost of ownership, in most cases, the advantages of using an EII or EAI product will outweigh the use of FTP and flat files, or some variant of this typical homegrown integration application.

Custom workflow

As indicated previously, workflow in legacy applications is often built into the user interface module or is implicitly part of the existing batch system. You would think companies that run legacy systems would have learned their lesson that this makes maintenance a nightmare and ends up costing large amounts of money, since changes to the legacy code or batch systems disrupt the workflow, and vice versa. These new open systems will then have the same problem that exists in legacy code today—the code cannot be changed, as no one knows what impact the change will have on processing.

The real world: studies in integration

In the following sections, we will take a look at several businesses and their challenges for integration. Such solutions as replication, integration with non-Oracle sources, and queuing will be discussed.

Banking case

Gruppo Sanpaolo d'Intermediazione Mobiliare is Italy's second largest bank and among the top 50 banks worldwide. The investment banking arm of the group is Banca d'Intermediazione Mobiliare (Banca IMI). In addition to servicing the other parts of the Sanpaolo Group, Banca IMI provides investment banking services to a wide range of institutions, including other banks, asset managers for major global corporations, and other financial institutions. In the process of buying and selling a variety of financial instruments, Banca IMI must communicate with all of the major exchanges (for example, the New York Stock Exchange). The volume of its daily trades can frequently scale up to hundreds of thousands.

Because its business operations are global, its IT systems must operate 24/7, and transactions with both internal application systems and external trading centers must be processed with minimum error and as close to real time as possible. This is a very demanding business application and a complex example of data integration. The main applications with which Banca IMI must communicate include its own backend administrative systems and legacy applications, the financial exchanges, domestic and international customers, and financial networks. Not only do all of these have different protocols and formats, but there are also differences just within the financial exchanges themselves. The latter is handled through a marketing interface layer with custom software for each exchange. Domestic customers are connected through open standard protocols (financial information exchange standards, FIX) and are used with international customers and other financial institutions.

Banca IMI uses **Oracle Streams** (**Advanced Queuing**) to coordinate transactions across all these stakeholders. Streams is a fully integrated feature of the Oracle database, and takes full advantage of Oracle's security, optimization, performance, and scalability. Streams can accommodate Banca IMI's very high level of transactions close to real time and still perform all the transformations required to communicate in the various protocols needed. Scalability is very important to Banca IMI and the primary reason why it moved to Oracle Streams from a previous solution that relied on another vendor's product. "*We're happy about what Advanced Queuing offers us*", says Domenico Betunio, Banca IMI's manager of electronic trading. "*Besides speed and scalability, we're impressed by messaging reliability, and especially auditing*".

Education case

The Hong Kong Institute of Education (HKIEd) is a leading teacher education institution in the Hong Kong Special Administrative Region (HKSAR). The institute was formally established by statute in April 1994 by uniting the former Northcote College of Education, Grantham College of Education, Sir Robert Black College of Education, the Hong Kong Technical Teachers' College, and the Institute of Languages in Education, the earliest of which was started in 1939. The Institute plays a key role in helping the Hong Kong government fulfill its commitments: to develop new curriculum; to achieve its goal of an 'all graduate all trained' teaching profession; and to provide for the continuous professional development of all serving teachers. The Institute is organized around four schools with a current enrollment of nearly 7,000 students in a variety of daytime and evening degree programs. The Institute staff exceeds 1,000, almost 400 of whom are teaching staff. Across the Institute, more than 200 funded research and development projects are being actively pursued. The two main languages the Institute must accommodate are English and Traditional Chinese.

Soon after its founding, HKIEd began in-house development of several administrative applications, all running in conjunction with Sybase databases. These applications included student admission, enrollment and profiling, human resources and payroll, smart card management, and a library interface. In 2002, HKIEd purchased a set of packaged applications: Banner, from SCT. SCT's Banner system runs on Oracle and is analogous to an enterprise resource planning system. It supports student services, admission, enrollment, and finance functions, some of which it took over from the in house Sybase applications. The Sybase in-house applications still account for roughly 50 percent of the Institute's administrative applications, including classroom booking, HR, payroll, JUPAS student selection, smart card management, and the library INNOPAC system.

As these vital Institute administrative systems are on two platforms, Sybase and Oracle, there is a requirement to keep the data consistent in more than ten common data fields. They account for less than 5 percent of the total number of fields, which is still significant. Each database contains more than 10,000 records. The number of daily transactions affecting these data fields ranges from 200 to 5,000.

After experimenting with SQL Loader scripts to update the common fields using a batch upload process, HKIEd switched to using the Oracle Transparent Gateway. Since January 2003, HKIEd has been using the gateway to access and update the Sybase database from Oracle to keep the common data fields in sync in real-time. HKIEd has created views in the Oracle database based on a distributed join of tables from the Oracle and Sybase databases. This enables SCT's Banner system to transparently access and update fields in the Oracle and Sybase databases. The system automatically performs a two-phase commit to preserve transactional consistency across the two databases. The Transparent Gateway has NLS support, enabling access to Sybase data in any character set.

High technology case

This case illustrates the use of Oracle Streams for information integration, load balancing, and consolidation.

Internet Securities, Inc. (ISI), a Euromoney Institutional Investor Company is the pioneering publisher of Internet-delivered emerging market news and information. Internet Securities (`www.securities.com`) provides hard-to-get information through its network of 20 offices in 19 countries, covering 45 national markets in Asia, Central and Eastern Europe, and Latin America. Its flagship product, the Emerging Markets Information Service aggregates and produces unique company and industry information including financial, economic and political news, for delivery to professionals over the Internet. The subscription-based service enables users to access and search through a comprehensive range of unique business information derived directly from over 6,800 leading local and international sources. Primarily because of its international clientele, the operations of ISI are run on a 24/7 basis. ISI has offices in 18 locales around the globe, with clients in each locale. Its provisioning operations are centralized and located, along with its headquarters, in New York City.

ISI's content is also global and emphasizes information about emerging markets, which in this context means markets in countries like Romania, Brazil, or China. The content being aggregated arrives in automated feeds of various forms at an average rate of 50,000 documents a day, with hourly arrival rates ranging from 100 to several thousand per hour. All documents, regardless of the source language, are converted to a single encoding standard (UTF8) when being loaded into the ISI document base.

One of ISI's competitive differentiators is that information is retained regardless of age. The size of the ISI content base has grown rapidly to over one terabyte and, in tandem, the level of query activity has grown as well. Until recently, daily operations were run on NT-based systems using Oracle. The need for high scalability and availability while superseding performance prompted ISI to migrate its database operations onto Solaris using Oracle9. Oracle Streams was selected to achieve a major increase in availability and performance. Higher availability is obtained by fully replicating to a secondary server and by being able to perform much faster backups. The performance improvements come from load balancing between the servers and the upgraded hardware.

Overall, Oracle Streams is used for three databases supporting ISI operations. Each database is replicated to a secondary server. The three are:

- **Back-office database** (**100 GB**): This database supports the company's proprietary CRM and authentication systems.
- **Document database** (**50 GB**): This contains metadata about documents and true paths to the physical location of documents.
- **Search database** (**1 TB**): This is the database in which the documents are loaded and where the client queries are executed. Documents are stored as BLOBS. Each record is one document.

The replication for each database is such that either replica can service the functions of both, but for performance (load balancing) and administrative reasons, both are typically serving different operational needs. For example, ISI call center agents primarily use 'Backoffice A' while 'Backoffice B' services all of the client activity monitoring. In the context of the Document database, 'Documents A' is the production machine, and 'Documents B' is the standby. In the case of the huge Search database, 'Search A' is used to perform the document loading that occurs in hourly batches, and 'Search B' is used to receive the users' queries that can be executed on either Search database server.

ISI expected to obtain benefits in two ways: performance and availability. It had evaluated other possible solutions, but these fell short in the performance dimension. Oracle Streams did not. With the Streams-based solution, queries are executing in half the time or better. With respect to availability, switchover in case of a failure is now instantaneous. With NT, ISI was using a physical standby that could be switched over in the best case in 7 to 8 minutes for read-only, and 10 to 15 minutes for read/write transactions.

Summary

In this chapter, we have opened the book, as it were, on many topics in the sphere of integration and migration of data. We began the chapter laying the foundation by addressing some of the reasons that propel an organization to tackle this problem, while addressing some of the unique challenges that one will face both from a technical and business perspective. Next, we began to unpack the concepts and approaches to integration, which will provide the foundation in the coming chapters, when we get more hands-on examples. Finally, we explored several real world examples in multiple industries that have dealt with migration and integration issues. In the coming chapters, we will begin to get more hands-on and take a deeper dive into specific tools and techniques, as well as more case studies throughout to illustrate the problems and solutions that other firms have experienced.

Where to buy this book

You can buy Oracle Information Integration, Migration, and Consolidation from the Packt Publishing website: `http://www.packtpub.com/oracle-information-integration-migration-and-consolidation/book`

Free shipping to the US, UK, Europe and selected Asian countries. For more information, please read our shipping policy.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.

[PACKT] PUBLISHING enterprise
professional expertise distilled

www.PacktPub.com

For More Information:
www.packtpub.com/oracle-information-integration-migration-and-consolidation/book