Critical Success Factors Deploying Pervasive BI

September 2007

James Markarian
Informatica
Chief Technology Officer

Stephen Brobst
Teradata
Chief Technology Officer

Jeff Bedell
MicroStrategy
Chief Technology Officer

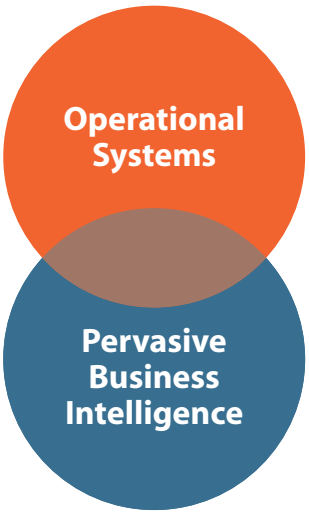

Table of Contents

Acknowledgements

We would like to thank the support team that made this work possible. It takes a minimum of two marketing staff to keep up with one CTO – and we had three CTOs writing this paper. Many thanks to Dan Graham, Jorge Lopez, Julianna DeLua, Amy Phillips, Tiffany Gumfory, and Cheryl Bowden.

Pervasive Business Intelligence

Historically, business intelligence and data warehouses have been associated with back office employees. In the 1990s, data was integrated and loaded nightly, optimized for trend reporting and strategic analysis, and delivered via reporting tools. Back office planners running pre-defined reports were the bedrock users of business intelligence (BI). Over time, knowledge workers evolved to demand richer, more diverse insights. As usage matured, requirements to include predictive analytics, event-driven alerts, and operational decision support have become the norm. While demand for near real-time information always existed in front office operational communities, the costs and complexity of loading data multiple times per day kept data out of reach. This caused expert database administrators and innovative software vendors to find ways of reducing data latency between source systems and the data warehouse (DW) to serve up "just-in-time" insights throughout the business day.

For many, the dividing line that kept (DW) information isolated from the front office vanished. Going beyond reporting, visionary enterprises found ways to enhance revenue and cost efficiencies across multiple business process and knowledge user communities. Front-line employees, suppliers, consumers, and business partners are finally being offered BI capabilities for tactical decision support. In this way, pervasive BI emerged within enterprises that compete using data analysis.

Pervasive BI Applications

Just as important as the executives who formulate corporate objectives are the rank and file employees who execute those objectives. Thousands upon thousands of small decisions made daily by employees, customers, and business partners add up to profitability and brand image[1]. Yet many front-line workers have historically made decisions without access to the necessary facts to achieve alignment with corporate objectives. The goal of pervasive BI applications is to take the data that produced the back office ROI of more than 400%[2] and deliver it to front-line employees in a form appropriate to their job functions – with similar results. There are thousands of business process steps in a typical enterprise where pervasive BI insights can be added. What are leading companies doing with pervasive BI? Here are a few highlights:

- Inbound and outbound cross selling by Customer Service Representatives
- Point-of-sale fraud detection
- Automated insurance claims triage and fraud detection
- Personalized next best offer on web sites, ATMs, and POS devices
- Supply chain business activity monitoring
- Retail out-of-stock monitoring on promotional items
- Labor and crew scheduling
- Optimizing trailer and container loading and routing
- Anti-money laundering
- Dynamic pricing and yield management
- Real-time manufacturing line quality alerts

Pervasive BI injects new workloads into the major BI subsystems. The classic extract-transform-load (ETL) subsystem now must cope with near real-time integration of data from multiple continuous data sources. The DW must support a mixed workload of constant data loading, complex reporting, and tactical requests. Subsequently, the BI platform must analyze all the data and deliver timely information through reports, alerts, dashboards, and operational applications. A new information supply chain must be established to deliver pervasive BI solutions.

[1] Frank Rohde, "Little Decisions Add Up," Harvard Business Review, June 2005

[2] IDC, The Financial Impact of Business Analytics

Pivotal Role of the CIO

The hardest part of deploying pervasive BI is changing the way an enterprise thinks about BI. The CIO must be a change agent – a thought leader in evangelizing the possibilities of a new vision and the required capabilities. By reaching out and educating enterprise executives and line of business owners, the CIO helps the organization conceive new ways to increase revenue or reduce costs using low latency information. This usually begins with the question, "What would we do differently if all employees had integrated analytic information in minutes instead of overnight?" The first step often involves educating the organization about how other enterprises have gained competitively from pervasive BI solutions. Second, the CIO must convince the IT architects and developers that pervasive BI can be implemented successfully. Many technical skeptics will surface, often with a vested interest in older architectures and designs. While the skills needed may exist within the IT organization, the J2EE/.NET developers and BI architect communities have rarely collaborated in the past. The *show me* skepticism of legacy developers must be satisfied to move forward. CIOs are accustomed to the challenge of organizational change, but that doesn't make it easy.

Integrating pervasive BI with operational applications does not have to be expensive or revolutionary. One key to controlling costs and risk is to deploy just-in-time or right-time data by choosing the right latency for data delivery. Data freshness service levels should be driven by business process requirements rather than market hype around so-called real-time data warehousing. Hence, right-time data delivery means fresh enough data delivered at the lowest possible cost. Similarly, pervasive BI is not a single project or expense. Typically, many small pervasive BI projects evolve over the course of multiple years – each one building on top of and adding to the infrastructure previously deployed. Another key to controlling costs is to have a reference architecture in mind as the infrastructure is built out. A good reference architecture provides a blue print for incremental deployment so as to minimize dead-end projects.

Pervasive BI Reference Architecture

So what is pervasive BI? Pervasive BI is the ability to deliver integrated right-time DW information to all users – including front-line employees, suppliers, customers, and business partners. It provides an enterprise with the necessary visibility, insight, and facts to make smarter decisions in all processes at all times. In most companies, this means leveraging the existing BI/DW infrastructure by providing decision services to multiple operational business processes.

The pervasive BI reference architecture illustrates both transactional services (i.e., OLTP) and decision-making services as peers in the existing infrastructure. Enterprise users may access the IT infrastructure via web portals, web applications, POS terminals, self-service kiosks, hand-held devices, and interactive voice response servers.

Transaction services are applications that provide the enterprise bookkeeping function. This is where we find traditional call center automation (operational CRM), enterprise resource planning (ERP), supply chain management (SCM), and legacy applications.

Data integration services bridges multiple domains, providing both continuous streams of information, as well as batch file data acquisition. Acquiring changed data from the transactional repositories; the data integration services extract, discover, cleanse, transform, and deliver data to multiple subscribers.

Decision repositories are the enterprise data warehouses, data marts, and operational data stores. They ingest and persist the results of data integration services and provide high-speed access to a wide variety of data content.

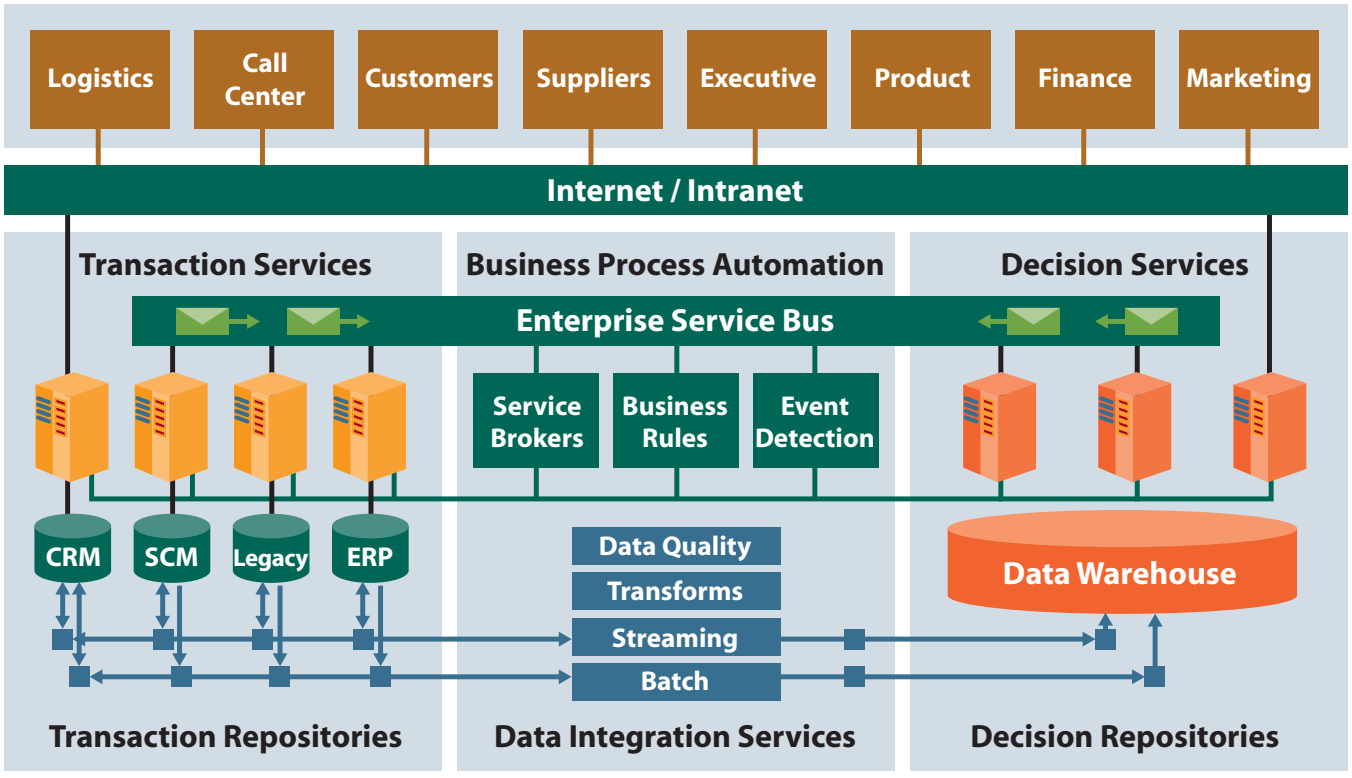

Figure 1. Pervasive BI Reference Architecture

Decision services are used to analyze facts, patterns, and relationships in enterprise data repositories and deliver relevant information. This part of the architecture focuses on BI and applications accessing the data warehouse. This includes reporting, data mining, tactical applications, operational applications, and strategic applications, such as market segmentation, risk analysis, category management, profitability analysis, and financial planning.

Enterprise application integration is largely achieved using an Enterprise Service Bus (ESB), messaging middleware, J2EE and .NET developer tools, and service-oriented architecture (SOA). Included here are numerous middleware services, such as adapters, transforms, agents, publish and subscribe, and information routing.

Business process automation is a collection of capabilities to oversee and orchestrate processes. This includes Business Process Management (BPM), Business Activity Monitoring (BAM), and Business Rules Engines (BRE). These systems manage SOA workflow, detect events, send alerts, and allow business users to dynamically change business rules in real time.

Decision Repositories: Active Data Warehousing

Traditional data warehousing focuses on delivering strategic decision support. Having a single source of truth for understanding key performance indicators (KPIs) with sophisticated what-if analysis for developing business strategy has certainly paid big dividends in competitive marketplace environments. Data mining techniques further refine business strategy via advanced customer segmentation, acquisition and retention models, product mix optimization, pricing models, and many other similar applications. The traditional data warehouse is typically used by decision makers in areas such as marketing, finance, and strategic planning.

Unlike strategic decision support, which is typically used by a relatively small number of high-level decision makers, tactical decision support is used to empower the in-the-field decision makers. Information leverage is maximized when hundreds or thousands of people who are making decisions every minute have the right information to improve the quality of their decisions. A tactical decision may be related to the treatment of a customer when a call is received by a customer service representative (should the fee be waived or not?), the re-routing of trains when a derailment takes place, the markdown pricing on a specific item in a specific store, or one of many other decisions that gets made many times per each hour in the course of running a business.

While no single tactical decision is strategic in nature, the ability to execute such decisions incrementally better than the competition thousands of times per day definitely has strategic implications. Leverage from the data assets managed in the enterprise data warehouse comes when they are put into the hands of employees throughout an organization – not just those in the corporate headquarters. However, providing the information by itself is not good enough. The information must be actionable, and business processes must be aligned to make effective use of the information from the active data warehouse.

Tactical Decisions

Delivery of tactical decision support from the enterprise data warehouse requires a reevaluation of existing service level agreements. The three areas to focus on are data freshness, performance, and availability. In a traditional data warehouse, data is usually updated on a periodic, batch basis; refresh intervals are anywhere from daily to weekly. In a tactical decision support environment, data must be updated more frequently. For example, while today's sales figures shouldn't be needed to make a strategic decision, access to up-to-date sales and inventory figures is crucial for effective (tactical) decisions on product markdowns. Batch data extract, transform, load (ETL) processes will need to be migrated to trickle feed data acquisition or frequent mini-batch data acquisition in a tactical decision support environment. This is a dramatic shift in design from a pull paradigm (based on batch scheduled jobs) to a push paradigm (based on near real-time event capture).

The stakes also get raised for query performance service levels in a tactical decision support environment. Tactical decisions get made many times per day, and the relevance of the decision is highly related to its timeliness. Unlike a strategic decision, which has a lifetime of months or years, a tactical decision has a lifetime of minutes or even seconds. How trains get rerouted after a derailment is a decision that must be made quickly. Every minute of delay in the decision has a severe economic cost in terms of further schedule delays. Similarly, the next best offer presented to a consumer on the web site has to be fast – the competition is only three mouse clicks away. Tactical decisions must be made in seconds or minutes so the opportunity to execute that decision isn't lost.

Of course, a tactical decision is typically more narrowly focused than a strategic decision and thus, there is less data to be scanned, sorted, and analyzed. However, these tactical decisions are by no means simple. The rerouting of trains will depend on many factors

ranging from domino effects if train cars are not where they are expected to be a day from now, evaluation of customer impacts if freight arrives later than expected, relative profitability of affected customers, delivery compliance on shipments to date, and so on. It turns out to be a very complex optimization problem – albeit with more selective data access than in strategic decision support where large numbers of customers, shipments, or assets would be examined over a longer period of time. Furthermore, the level of concurrency in query execution for tactical decision support is generally much larger than in strategic decision support. Now imagine strategic and tactical decision support queries co-existing in the enterprise data warehouse. Clearly, there will need to be distinct service levels for each class of workload. Computer resources will need to be allocated to queries differently according to the type of workload.

Prioritizing Mixed Workloads

For data warehouses focused purely on strategic decision support, managing resources typically amounts to allowing concurrent execution up to some maximum number of queries. In most implementations, all queries are treated equally in the allocation of resources. When data warehouses focus only on long-term decision making, the need for immediate answers and clever resource allocation is not paramount. These days, data warehouse environments are not so simple.

Mature data warehouse deployments need to handle a mixed workload, including combinations of reporting, OLAP, ad hoc queries, data mining, continuous data loading, tactical decision support, and event-driven triggered decision execution. Unfortunately, many implementations do not manage resources in an optimal way for mixed workloads. The reality of a mature data

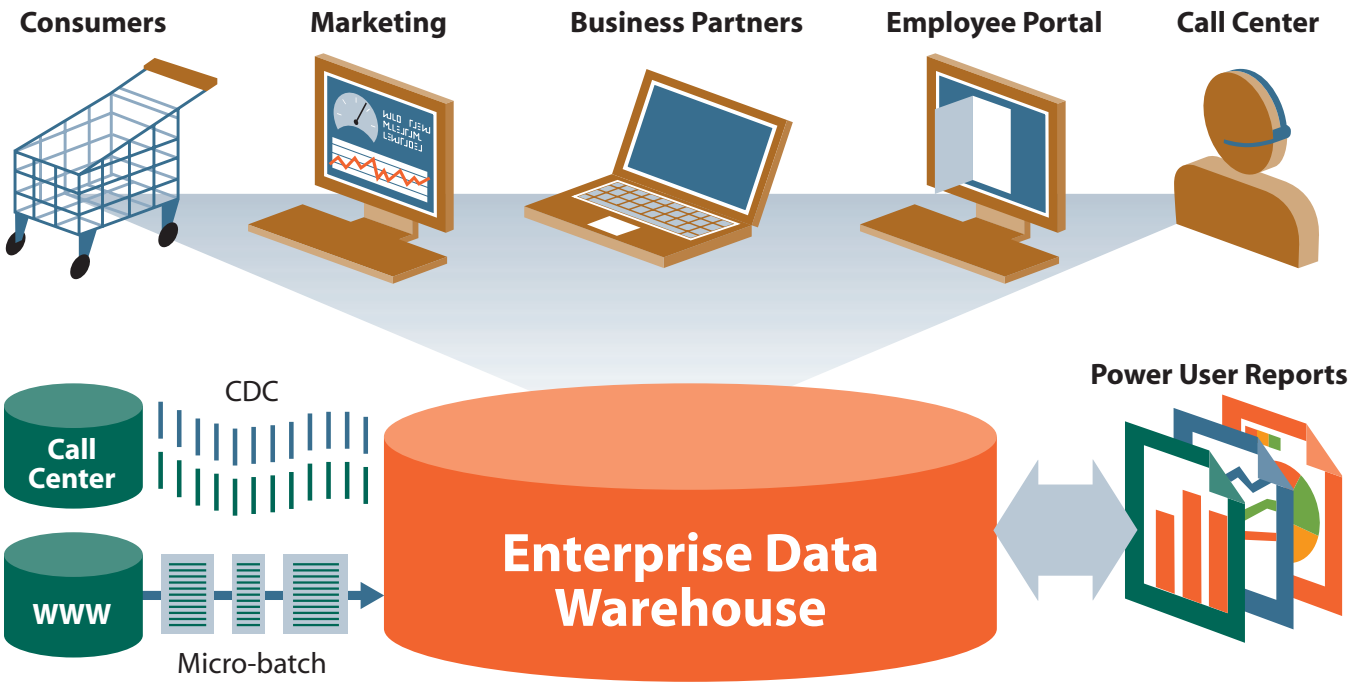

Figure 2. Tactical Decisions, Strategic Decisions, and Continuous Load

warehouse deployment is that different aspects of the mixed workload have very different service level requirements. Giving a data mining query with a two-hour service level and a tactical decision support query with a two second service level equal priority access to resources will not be optimal. Without prioritization, the two second tactical query may become a poor performing 200 second query.

Proactive resource management yields much better results. There are two primary aspects of proactive resource management – query scheduling and prioritized resource allocation. Query scheduling determines when a query gets released into the database for execution. Short queries are typically launched immediately, whereas resource intensive queries with flexible service levels can be deferred until computing resources are freed up. Proactive query scheduling prevents thrashing inside the database server caused by over committing resources. Estimates from the cost-based optimizer provide the basis for the scheduling decisions.

Dynamic prioritized resource allocation should be used to manage access to computing resources once a query is scheduled for execution. The total capacity of the machine must be divided up among concurrently executing queries. Overlapped I/O and CPU execution takes place via multi-tasking and micro-scheduling of queries within the RDBMS processes and tasks. How resources get allocated in this complex execution environment is based on the resource allocation policies implemented in the data warehouse database. Many relational databases lack mature algorithms for managing resource allocation by priorities, making it difficult to meet service level goals. Some use static mixed workload management which permanently allocates resources to workloads. Lacking dynamic resource prioritization, some CPU resources will sit idle while tasks are backing up trying to finish their job. This thorny computer science problem should be thoroughly understood and reviewed before embarking on pervasive BI projects.

Mission Critical Data Warehouses

Availability is typically the neglected stepchild in terms of service levels for a traditional data warehouse. Given the long-term nature of strategic decision making, if the data warehouse is down for a day, the quantifiable business impact of waiting until the next hour or day for query execution is not often large. Not so in a tactical decision support environment. Incoming customer calls are not going to be deferred until tomorrow so that optimal customer care decisions can be instantiated. Down time on an active data warehouse translates to lost business opportunities. As a result, both planned and unplanned down time will need to be minimized for maximum business value delivery.

Active data warehousing is clearly established as the new breed of decision support. Providing both tactical and strategic decision support from a single, consistent repository of information has compelling advantages. Such architectures naturally encourage alignment of strategy development with operational execution. However, radical rethinking of existing data warehouse architectures will need to be undertaken in many cases. Evolution toward more strict service levels in the areas of data freshness, performance, and availability are critical. Even if business stakeholders are not yet ready to embark upon the business process redesign necessary to integrate tactical decision support into their operations, it will be important to at least construct a road map for data warehouse deployment that anticipates eventual evolution toward an active data warehouse infrastructure.

Data Integration Services

Any cogent strategy for delivering pervasive BI will involve a new approach to data integration. Pervasive BI isn't just delivering tactical insights to a broader audience. It also requires an architecture that reliably delivers the necessary data, at the right frequency, with requisite flexibility and quality.

What is a Data SLA?

The success of any initiative relies on clearly framing the requirements around data sufficiency and the proper frequency for updates. Thinking of this in terms of a Data Service Level Agreement (SLA) is a useful way of quantifying user expectations and intrinsic capabilities of the integration architecture. This also includes assessment of how data providers to the architecture will allow fulfillment of knowledge worker expectations.

There can be numerous components of a Data SLA. While not a complete list, the most critical are data freshness, data completeness, and data accuracy. A good Data SLA will define expected, acceptable, and unacceptable values for each of these factors. It will also provide a clear agreement with the users and a prioritized delivery plan. Considerations for each category include:

Data Freshness

When considering real-time decision making, a change in mindset toward data freshness is needed. Fresh data, delivered immediately upon creation or change, enables agility. For example, enterprises need agility to rapidly react to fraudulent financial transactions, quickly respond to retail stock-outs, or deliver traffic warnings to on-board delivery truck navigation systems. Understanding the business "speed of need" allows architects to determine which data must be delivered at what intervals.

Not all data requires real-time delivery. Fast enough should be the architect's goal. When an airplane flight is cancelled, the airline has just a few minutes to collect and analyze the data needed to compute 100+ passengers' new itineraries. In contrast, an online global positioning system supplier may gather road outages and detour data in real time, but only integrates and publishes this information every few hours. Fast enough is defined differently by the airline and GPS supplier based on the impact of each degree of latency on performance metrics, such as revenues, costs, and customer satisfaction.

Data Completeness

Tactical decisions need to be made in context. Data context comes from compiling all relevant data from every possible source. Real-time information sources, such as message queues, ESBs, and web services, must be tapped to supply data to the data warehouse. Information should also be extracted from XML interchange formats, such as SWIFT, HIPAA, X12, EDIFACT, and even PC documents, such as Word, Excel, and PDF documents. These unstructured data formats must be included as part of the data integration plan for pervasive BI.

Data Cleansing and Accuracy

Making decisions based on inaccurate data is, at best, a risky proposition. Data that arrives in real time to be used in real time will naturally need to be cleansed in real time. Cleansing data in real time is essential to avoiding unnecessary or incorrect actions. The data integration platform possesses the ability to standardize, cleanse, and match data across all data sources to ensure all relevant data is linked and de-duplicated within the data warehouse. For example, using data quality rules, the data integration subsystem can report and alert stakeholders in real time of possible fraud, identity theft, or money laundering. In the age of hackers and advanced crime rings, fast reaction time can help prevent organizations from incurring major financial losses, as well as legal and other damages. As with the other elements of the Data SLA, the investment in cleaning categories of data must be adjusted according to the business value of that data.

How do We Achieve the Data SLA?

There are four major requirements for achieving the Data SLA: real-time collection, micro-batch, change-data-capture, and real-time resilience.

Real-Time Collection

Real-time data collection pushes data to the database from the systems of origin in one of three ways: messaging, changed-data capture, or web services. Messages are delivered via message queuing, EAI, or ESB middleware (See the *Reference Architecture* illustration). Two important aspects to consider are the semantics of handling real-time information and participation in messaging service level agreements.

Basic integration with messaging systems is straightforward for data integration systems that provide always-on processing. However, real-time sources cannot be treated as peers of their batch counterparts. Processing real-time data requires data integration engines to provide the same types of set-oriented processing (aggregation, joining, and ranking) as in batch processing. This often proves fatal for integration engines designed to expect end-of-data markers. Instead, true real-time data integration engines can map set-oriented semantics to data streams. This means identifying data windows by elapsed time, message count, data markers, or latency between messages.

Guaranteed Message Delivery (GMD) chains require that all systems involved provide both message delivery acknowledgements and recovery capabilities. Delivering a single message is relatively straightforward: a message is received, transformed, and ingested into the target RDBMS. Once the data is successfully committed, an acknowledgement is sent to the messaging system. When dealing with sets of records, the data integration platform must be able to handle units of work involving multiple messages as atomic units which must get delivered and acknowledged together or not at all. For example, an invoice with 150 line items

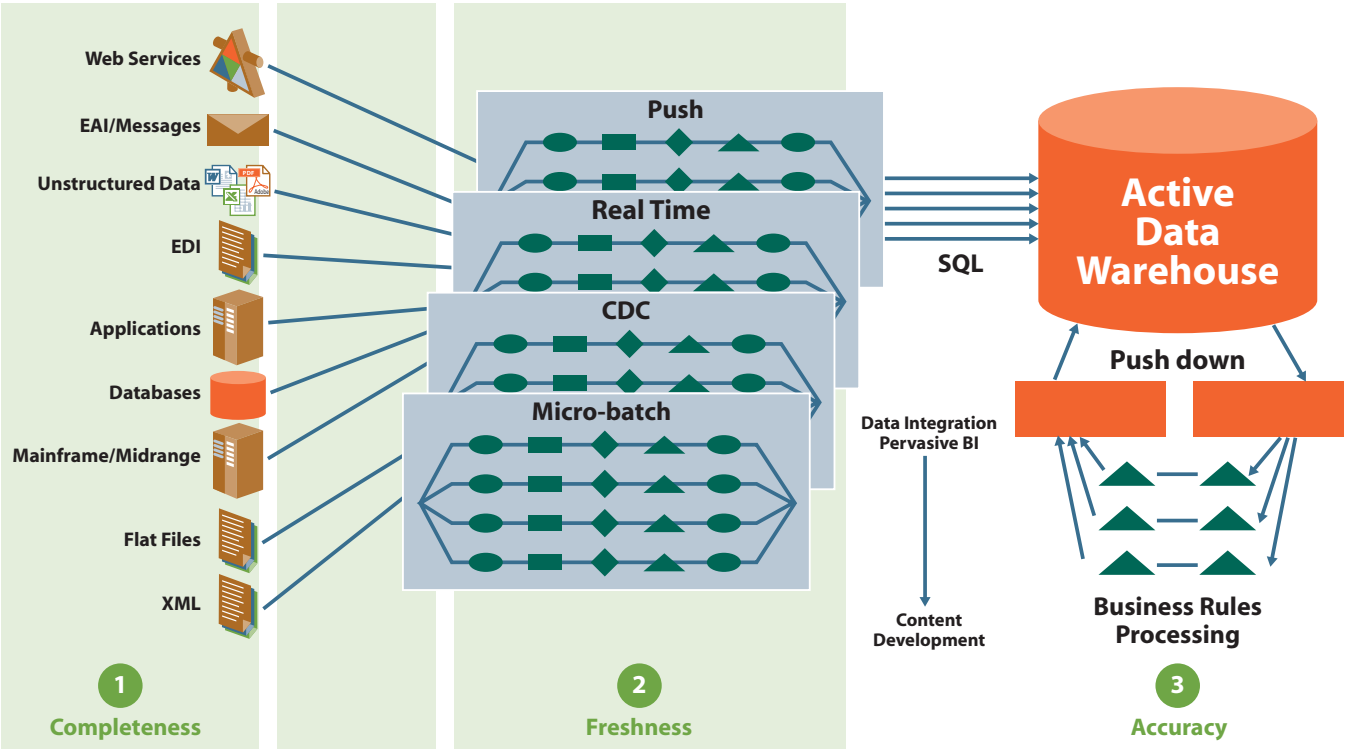

Figure 3. Data Integration Services for the Attainment of Data SLA

may be sent in three messages, but it is still one unit of work. In order to handle failures in the unit of work case, the integration system must perform its own recovery logging to avoid violating requisite write semantics such as "write only once" (i.e., no duplicate records caused by improper abort recovery). All of this is aimed at data freshness and completeness for pervasive BI use.

Changed Data Capture

Changed Data Capture (CDC) is another useful technique for capturing data in near real-time. It generally works by sniffing changes from the DBMS recovery log journals as transactions are being committed to the database. Changes may also be captured via operating system or transaction system exits as in the case of VSAM or IMS respectively. Changed data is propagated to the subscribers of the CDC service. This approach can have several advantages over message-based systems. Foremost, it is completely non-intrusive into source applications. CDC mechanisms are also completely insulated from being subverted or avoided. CDC systems can work with legacy DBMS systems, such as IMS and IDMS, without resorting to screen-scraping or modifying COBOL or REXX code. CDC supports data freshness and completeness for pervasive BI.

Micro-Batch Model

As a step toward near real-time processing, many organizations are moving to a micro-batch model of frequent, small, scheduled *pulls* of information. This can be especially useful for cases like call detail records or click stream data processing where the files are created and made available for processing at periodic intervals. Despite the name micro-batch, the data volumes can still be enormous with respect to the time available to process them.

This places a premium on performance and scalability in the data integration platform. The platform must provide a pipelined architecture capable of partitioning data across threads and processes on a single SMP server or across multiple nodes of an MPP server. The data integration platform workload manager must balance I/O intensive operations against CPU intensive operations for optimal throughput. It must be able to specify node affinities for specific processing to take advantage of lookup caches and data services, while minimizing network or process hops. Lastly, it must take advantage of native RDBMS interfaces and loaders to optimize moving data into the target data warehouse. This includes applying push-down optimization to move certain types of processing into the RDBMS for better co-location with dependent data and performance.

Real-time Resilience

To ensure the reliability of data integration services, a data integration platform must be expressly engineered for high availability. The platform must possess capabilities for resilience, failover, and recovery. It must also work in conjunction with those same capabilities provided by the surrounding DBMS, messaging, OS, and hardware environment.

Resilience is the first line of defense. Dropped database and network connections should be reestablished if possible. Writes should be re-tried and FTP sessions re-initiated. Hardware and software failures must be monitored and failover measures taken when they are detected. What happens when active jobs fail? The integration platform must provide automated recovery mechanisms, including check-point/restart, compensation, and restart-from-beginning. In CDC environments, processing must resume from the last position in the copied transaction journal for short-duration outages. Archive logs may have to be scanned in the cases where the outage time is greater than the lifetime of the recovery journal (redo log). This applies to both long duration outages for the integration platform and the data warehouse.

Long outages in queuing environments require persistent queues to retain changes while the integration services failover to available hardware nodes. Long outages by the data warehouse DBMS can be handled by either delaying de-queue operations or pooling data in temporary file-based persistent stores by the integration platform.

Decision Services – Pervasive BI for Front-Line Users

Leveraging corporate data to improve the decisions of non-traditional BI users is the core value of pervasive BI. To achieve its full potential, the infrastructure must deliver more than raw data or simple views of the transactional systems. The data must be transformed both analytically and graphically to fit the front-line users' skills, operational business needs, and preferred delivery methods and devices. These transformations must occur quickly to satisfy the business's decision processes and SLAs. Consequently, the BI platform needs to provide the levels of service, analytic scale, and flexible delivery mechanisms required by a broader and more diversified user population.

BI Service Level Agreements

The primary operational requirements for a pervasive BI platform are scalability, performance, and reliability. Intersecting with these operational requirements is a much more diversified user population consisting of non-traditional BI users including front-line workers, partners, and customers. As a result, pervasive BI applications produce a highly diverse query workload. While traditional BI users rely more on summary reports to monitor the company's objectives, front-line users need atomic-level details to make immediate decisions about specific customers or shipments. This requires both historical and current integrated data, which must be delivered according to the business's SLAs.

The BI SLAs will almost always need support for large numbers of concurrent requests, which in turn, requires the system to minimize the processing overhead of each individual request. To achieve the first goal requires an efficient BI architecture, which includes multi-threading to optimize the use of multiple processors in a single server (scale-in), server clustering (scale-out), and fully utilizing 64-bit memory spaces common in today's server platforms. To achieve the latter goal, the BI platform must resolve BI queries using a true ROLAP[3] approach that fully leverages the processing capabilities of the data warehouse.

The concept of ROLAP is often misunderstood and is implemented differently across BI vendors. To clarify, the most efficient ROLAP implementation (true ROLAP) resolves a BI request to a query, including aggregations, calculations, and filtering tasks, that is passed to a database, and only the result is retrieved. This technique minimizes the number of intermediate data structures and other temporary result sets. Significantly, the use of intermediate data structures in pervasive BI is not always possible, especially for ad-hoc requests with near real-time query response requirements. A true ROLAP approach generates highly optimized database-specific SQL, calculating complex analytical functions without relying on any intermediate data structures such as cubes or caches. Consequently, the BI platform can provide both data and user scalability while continuing to meet the performance requirements of near real-time requests.

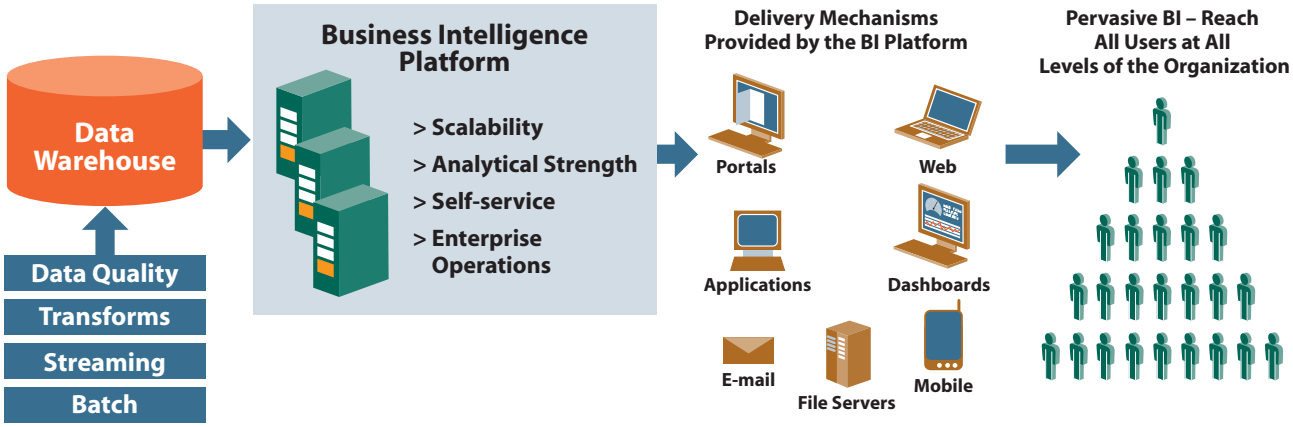

Figure 4. BI Platform Powers the Decision Services Subsystem

[3] Relational Online Analytical Processing (ROLAP)

As with data integration services and decision repositories, the BI platform must provide mission-critical levels of availability. The BI platform achieves high availability through comprehensive monitoring capabilities, automated administration, clustering capabilities, load balancing, failover support, and self-tuning engines that adapt in real time to changing load conditions.

Pervasive Analytics

A more diverse user population requires a wider range of analytical functionality and information styles. Therefore, the BI platform must support extensive analytical functions, which can range from simple and complex calculations to data mining models and predictive analytics. For example, traditional BI users, such as business analysts, might interactively perform root-cause analysis and build comprehensive models for predictive analysis. Subsequently, front-line users can seamlessly use the predictive models in tactical real-time requests. This near real-time data mining is moving towards mainstream use.

A primary benefit of pervasive BI is delivering blended data. This mixture of right-time analytics and historical data provides a more complete picture. For example, a call center representative's (CCR) display can combine customer purchase history, customer lifetime value and churn likelihood scores, a set of next-best-offer proposals, and summaries of web site click streams within the past ten minutes. By understanding the value of the customers, where they have been (via their click stream), and best match recommendations, the CCR has the highest likelihood of resolving any issues and successfully up-selling customers. With traditional BI, end users often start with summary data and drill down to transaction detail to understand root causes and identify specific problem areas. With pervasive BI, users typically start with the transaction detail required to support a specific business process, but then require broader analytics including summaries and historical information to optimize their decisions. This is commonly the case for users who require near real-time responses.

Delivery Mechanisms for Pervasive BI

The BI platform serves as the delivery vehicle for information using two basic delivery models (push and pull) and a range of delivery mechanisms. The pull model commonly involves pervasive BI users initiating queries from within their applications of choice. In contrast, the push model is initiated by the system itself, alerting users based on a set of exceptions and/or rules. Alerts can also be triggered by events raised directly on an ESB.

In both models, the pervasive analytics must be served via the optimal delivery mechanism. Three main categories of delivery mechanisms exist: traditional BI Interfaces, custom or composite BI applications, and direct integration with operational and existing applications.

Traditional BI interfaces are optimized for traditional BI users such as business analysts or executives' web-based reports and dashboards. Other BI interfaces, such as email delivery and, more recently, wireless BI interfaces are common ways of deploying tactical information. Both methods provide the ability to deliver data to users without requiring access to a full web or client/server application. Email and SMS are particularly well suited for delivering real-time alerts and serving analytics to large user populations, such as delivering warning alerts to executives or individually tailored promotions to customers.

Composite BI applications support a particular set of business processes and combine operational workflows with pervasive BI analytics. They provide decision makers with all the pertinent information in the same user interface as the related operational functions, so the user can make a decision and immediately implement it. The most efficient way to build composite applications is through the use of web services, service-oriented architectures (SOA), and open standards, such as Simple Object Access Protocol (SOAP) and eXtensible Markup Language (XML). The BI platform must be able to share pieces of BI functionality, and also deliver analytics and alerts as XML or highly formatted HTML. The BI platform plays the role of decision services middleware and as such must provide a web services SDK, including the ability to deploy and host web services using Web Services Description Language (WSDL) and Universal Description Discovery and Integration (UDDI) registration. Alternatively, customers may "assemble" composite applications by leveraging a portal

server. Therefore, the BI platform should provide out-of-the-box integration for leading and open source portals including JSR-168 and WSRP portlets. The role of the BI platform in both cases is to provide the pervasive analytics services.

With the third type of delivery mechanism, the pervasive BI platform integrates seamlessly into existing operational applications; the operational applications are augmented with pervasive analytics. For example, the CCR can bring up their existing support application which is now extended to include BI content. The benefit of this approach is that front-line users do not need to learn a new interface, although it may not provide as much flexibility as the composite approach. This level of integration can be developed using .NET and J2EE, or, increasingly, Flash/Flex. The BI platform should expose rich APIs based on common standards such as Java and XML. It should also provide plug-ins for common IDEs, such as Eclipse and Flex Builder, to facilitate integration with existing applications.

Decision Services for Pervasive BI

Like data integration services and decision repositories, BI platforms are rapidly adding new capabilities to support real-time operational processes, as well as a much broader community of users. Business intelligence is permeating hundreds of business processes to deliver actionable information throughout an entire organization. As critical information is deployed to diverse users, the BI platform must provide support through SOA applications, portals, dashboards, batch systems, and other middleware subsystems, while still serving the traditional BI users through a rich set of interfaces.

The long-term advantages of a pervasive BI system are realized as operational and traditional BI needs blend, and the infrastructure (the data services, the data warehousing platform, and the BI platform) is leveraged for both strategic and operational applications.

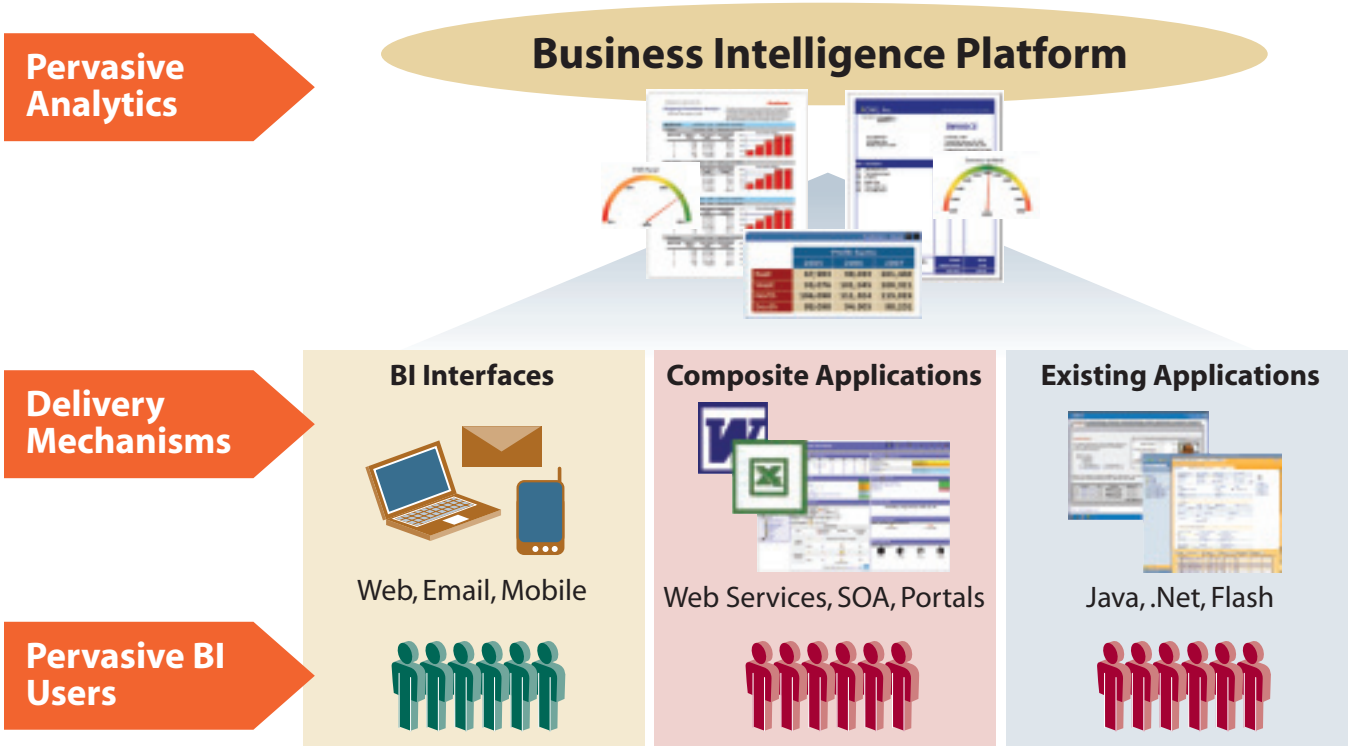

Figure 5. BI Platform Delivery Vehicles

Success Factors Summary

The three major subsystems – Data Integration Services, Decision Repositories, and Decision Services – are part of an information supply chain. Data starts in raw form, goes through transformations, storage, distribution, and packaging until it reaches the final consumer. All three are needed to support pervasive BI.

Pervasive BI exposes a need that was often not present in traditional BI – SLAs in all three subsystems. This is because the front-line user has near real-time performance expectations, 24 hours per day, and 365 days per year. It is irrelevant to the call center or web site consumer what part of the infrastructure is failing or slow. Thus, the critical success factors focus on formal SLAs for:

- Data freshness, cleansing, accuracy, and completeness.
- Scalability in terms of concurrent users by delivery mechanism.
- Mixed workload management to ensure service level performance goals.
- Tactical query response time measures by type of user and analytic.
- High availability metrics by user community and delivery mechanism.

Failure to negotiate and meet these SLAs puts company revenues, costs, and reputations at serious risk. The light-hearted early days of putting ETL scripts, schema changes, and new reports into production using haphazard tools and processes must be replaced with rigorous quality testing, strong operational procedures, and failover systems that ensure end-to-end information availability. The pervasive BI infrastructure must be integrated into the mainstream of existing IT operations.

Other critical success factors are architectural. Flexibility and versatility are needed to future-proof the IT infrastructure from ever increasing communities of users and devices. Many organizations have turned to portals and web services for flexibility and versatility. The same is true with data integration services where real-time data collection must accommodate new sources and types of data on a regular basis.

For many organizations, pervasive BI is the next step. Existing ETL tools evolve into data integration servers. Existing data warehouses are *activated*. And existing BI platforms renew themselves as decision services. What this requires is a new set of service levels recognizing the performance demands and end-to-end mission critical availability requirements of an Active Data Warehouse. Well defined SLAs clarify design goals and enable thoughtful business executives to connect the dots between fresh data and competitive advantage.

Glossary

Active Data Warehouse
An active data warehouse (ADW) is a traditional data warehouse extended to provide operational intelligence based on historical data combined with today's up-to-date data. The ADW supports mixed workloads from an enterprise data warehouse that serves as a single source of truth for decision making with predictable service levels for query response times, near real-time data freshness, and mission critical data availability. Moreover, the ADW integrates into the overall enterprise architecture to deliver decision services throughout an organization.

BI platform
A middle-tier server and services between the business knowledge worker and the decision repositories that process requests and deliver the results to an appropriate end-user device.

Business intelligence
An interactive process for exploring and analyzing structured, domain-specific information (often stored in a data warehouse) to discern trends or patterns, thereby deriving insights and drawing conclusions.

Data mart
A subset of enterprise data, typically pre-joined and highly aggregated, populated into an information repository to support a unique set of application-specific data analysis requirements.

Data warehouse
The primary repository of historical data for an enterprise. This repository is used for a wide range of analytics and reporting. It contains the raw material for management's decision support environment. The critical factor leading to the use of a data warehouse is that a data analyst can perform complex queries and analysis.

Enterprise service bus
An enterprise service bus (ESB) is middleware that supports inter-process communication and application interactions. ESBs offer program-to-program communication protocols, such as SOAP/HTTP, SOAP/MOM and message oriented middleware (MOM or MQ). ESBs provide support for web services standards, including XML, WSDL, BPEL, WS-Addressing, and WS-Security.

Extraction, transformation, and loading
Extraction, transformation, and loading (ETL) is a process in data warehousing that involves extracting data from outside sources, transforming it to fit business needs, and ultimately loading it into a data repository, such as a data warehouse, data mart, OLAP cube, or other construct.

Integrated development environment
An integrated development environment (IDE) is a programmer workbench that contains a range of tools and features for programming, debugging, testing, and deploying software applications. An IDE may also include development frameworks, project management, software configuration management, and component design and assembly.

J2EE (Java 2 Enterprise Edition)
J2EE is a widely-used platform and specification for server programming in the Java programming language. J2EE includes several API specifications, such as JDBC, RMI, e-mail, JMS, web services, and XML, and defines how to coordinate them. Java EE also includes specifications for Enterprise JavaBeans, servlets, portlets, and Java Server Pages.

Key performance indicators
Key performance indicators (KPIs) are financial and non-financial metrics used to quantify objectives to reflect strategic performance of an organization. KPIs are used in business intelligence to assess the present state of the business and to prescribe a course of action.

Microsoft .NET Framework

Microsoft .NET Framework is a software component that can be added to, or is included with, the Microsoft Windows operating system. It provides a large body of pre-coded solutions to common program requirements and manages the execution of programs written specifically for the framework.

Massively parallel processing

A high-performance computing architecture that allows a large (massive) number of loosely-coupled compute servers to work together in solving high-end scientific and business applications. The defining characteristic for an MPP architecture is the parallel software and a scalable switching interconnect that allows for potentially thousands of CPUs to work together in solving a single, large application problem with linear scalability.

Portlets

Java or .NET servlets that run in a portal. Low-level, point-to-point components offer access to application programming interfaces, uniform resource locators (URLs), or structured query language (SQL) statements.

Push down optimization

A data integration technique whereby data transformation, often in the form of an SQL statement or clause, is passed to an RDBMS where it is executed instead of in the data integration server. Business intelligence platforms also push down tasks and work into the RDBMS as summarization or formulas. When scalable RDBMS software is available on top of a powerful server, processing time can be dramatically faster than would be the case without push down. Both Informatica and MicroStrategy have push down capability. Teradata provides the scalable RDBMS platform.

Relational online analytical processing

Relational online analytical processing (ROLAP) tools generate SQL queries that are executed against an ANSI compliant RDBMS to summarize and calculate information into a multi-dimensional pivot table format. ROLAP is a more scalable alternative to MOLAP (Multidimensional OLAP) implementations that pre-summarize data into proprietary cube repositories for access by knowledge workers.

Symmetric multi-processor

A symmetric multi-processor (SMP) is a computing architecture for organizing multiple CPUs with shared access to memory and disk resources.

Service-oriented architecture

A service-oriented architecture (SOA) is a design for linking business and computational resources (principally organizations, applications, and data) on demand to achieve the desired results for service consumers (which can be end users or other services). OASIS defines SOA as "A paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with, and use capabilities to produce desired effects consistent with measurable preconditions and expectations."

Simple object access protocol

A simple object access protocol (SOAP) is a protocol for exchanging XML-based messages over computer networks, normally using HTTP/HTTPS. SOAP forms the foundation layer of the Web services stack, providing a basic messaging framework that more abstract layers can build on.

Extensible markup language

Extensible markup language (XML) is a general-purpose markup language that allows its users to define their own tags. Its primary purpose is to facilitate the sharing of data across different information systems, particularly via the Internet. It is a simplified subset of the standard generalized markup language (SGML), and is designed to be relatively human-legible.