IBM's Smart SOA^{tm} Approach Delivers 21st Century Transaction Integrity

John Santoro
IBM
January 18, 2008

Table of Contents

1. Executive Summary

Many companies today are looking to Service Oriented Architecture (SOA)-based applications to give them greater business agility. This agility translates into applications that developers can create in less time, that better leverage enterprise assets, and require less effort to adapt to changing business conditions. However, in their eagerness to realize the benefits of SOA, companies often neglect to consider that in order to be successful, applications must be "enterprise ready." Enterprise readiness means that applications are built with reliability and scalability.

Each day businesses execute billions of business transactions - think of the world's financial exchanges, credit card purchases, or on-line commerce. A failure in these systems could cost a company millions of dollars in lost revenue and lost customer satisfaction. As a result, a key aspect of enterprise readiness is **transaction integrity**. Ensuring transaction integrity is the ability of an SOA platform to maintain data in a reliable and consistent state, regardless of any system or business failure that may occur. Since the hallmark of SOA applications is their integration of enterprise-wide resources, transaction integrity is essential to guaranteeing that SOA applications update applications without putting critical data at risk.

No matter how quickly or elegantly a developer can create an application, if that application cannot ensure transaction integrity it jeopardizes every application that it touches. Enterprise ready applications need to guarantee transaction integrity in each component of the SOA platform, including the business process, enterprise service bus, database, and hardware. If any one of these components fails to ensure transaction integrity, the entire application is at risk. This need for comprehensive transaction integrity should keep awake at night everyone in an organization from the CEO, IT executives, administrators, and developers.

IBM has been guaranteeing transaction integrity (and a good night's sleep) to its customers for over forty years. Pioneering transaction processing with CICS, IMS and DB2, IBM has created an environment in which all components work together, from software to hardware, to deliver transaction integrity. IBM's success in transaction processing is evident in the fact that today CICS and IMS systems together perform over eighty billion transactions per day! IBM has taken these unrivaled technologies, which have provided transaction integrity to many of the world's largest businesses, and incorporated it into its SOA platforms, most notably with its WebSphere products, on a range of hardware, from distributed to mainframe. IBM's transaction integrity pervades all layers of its platform, from the business process, to the enterprise service bus, to the database, and to the hardware itself. With this comprehensive transaction integrity, IBM's SOA platform delivers "enterprise ready" transaction processing for the 21st century.

2. What is Transaction Integrity?

When critical processes fail, the business can lose revenue and reduce customer satisfaction and loyalty. Moreover, if the processes do not execute reliably, the business exposes itself to increased financial and regulatory risk. SOA-based processes act as a horizontal thread that access enterprise application content, represented as services, from vertical, silo-ed applications. Although the individual applications ensure data consistency within their own boundaries, they have no way to ensure consistency when an SOA application updates their data. In these instances it is the responsibility of the SOA application to maintain the integrity of the data in the applications that it accesses. The SOA application provides this integrity by establishing a transaction that guarantees that it will update all services in unison.

Without transaction integrity, application data can become inaccurate and inconsistent. As a result, SOA processes must be able to guarantee reliable execution. They must be able to recover from IT problems, such as server outages, or business problems, such as an item that is shown to be in stock but actually is not on the warehouse shelf. Transaction integrity is the ability to ensure that any IT or business execution failure can be corrected and application data restored to an accurate, consistent state.

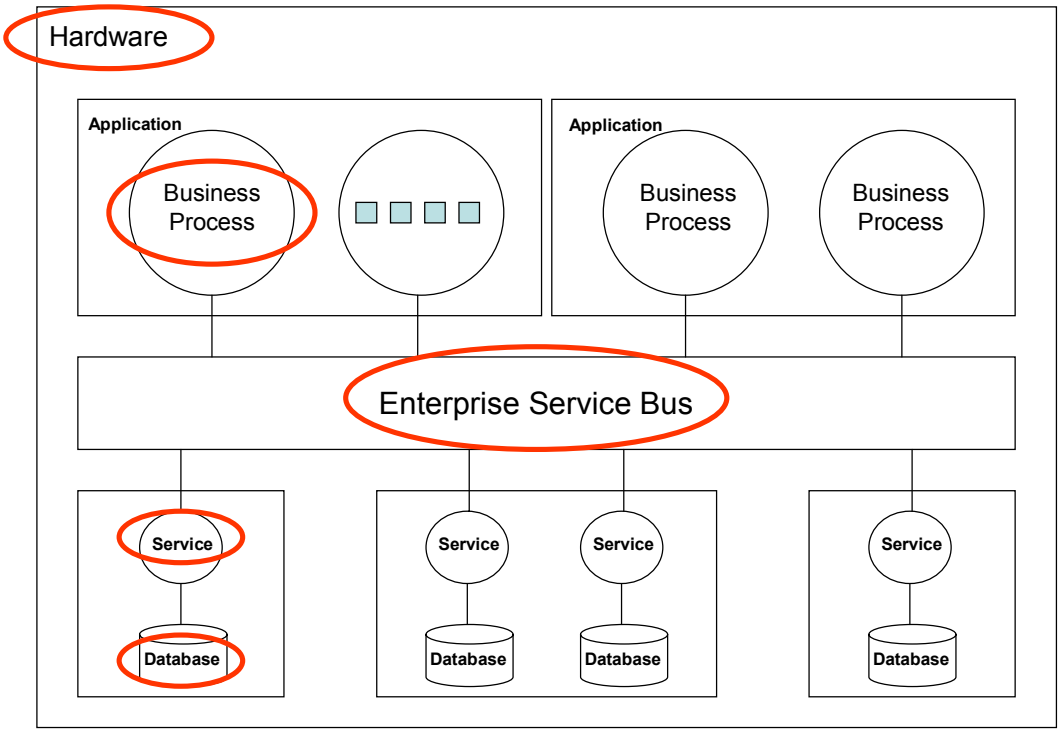
For enterprise business processes, transaction integrity has different implications depending on whether the process is synchronous or asynchronous. Synchronous processes complete nearly instantaneously, and do not involve events that require the process to wait, such as responses from human participants. For these processes, transaction integrity follows the "ACID" principle – transactions must be Atomic, Consistent, Isolated, and Durable. Generally, ACID qualities are achieved by the process' beginning a transaction and then committing or rolling-back based on whether all steps in the process execute successfully. Such a transaction ensures that the affected data remains in a consistent state. Asynchronous processes can take hours, days, or weeks to execute, and thus require a different idea of transaction integrity. For asynchronous processes, such as those that require interaction with human participants, data consistency cannot be a measure of transaction integrity, as it may take some time until all steps in the process are executed and thus all database changes are complete. As a consequence, transaction integrity for asynchronous processes is the guarantee that process instances or states will not be lost. As long as the process instance remains in the system despite any application or hardware failures, the enterprise ultimately can reach a consistent state. An SOA platform must support transaction integrity for both synchronous and asynchronous processes in order to provide truly enterprise ready applications.

IBM has been delivering transaction integrity for over forty years. For today's SOA applications IBM takes its unrivaled transaction processing capabilities in CICS and IMS and translates them to its WebSphere platform, from the business process and enterprise service bus layers through to the database and the hardware.

3. Where is Transaction Integrity Required?

Transaction integrity requires that every component of an application can recover from any IT or business failure. These failures could take place in the business process, the services that the process invokes, the enterprise service bus, the database, or hardware.

Where is Transaction Integrity Required?



An SOA solution that cannot ensure transaction integrity in each component gives an organization a false sense of security. An application may be able to withstand certain types of failures, but ultimately it will be dangerously exposed to a failure to its Achilles heels. With such a failure, a compromised business process can cause a loss of process state or data integrity. While process state and data integrity may sound like purely technical problems, this consequence of incomplete transaction integrity can have a profound affect on your business.

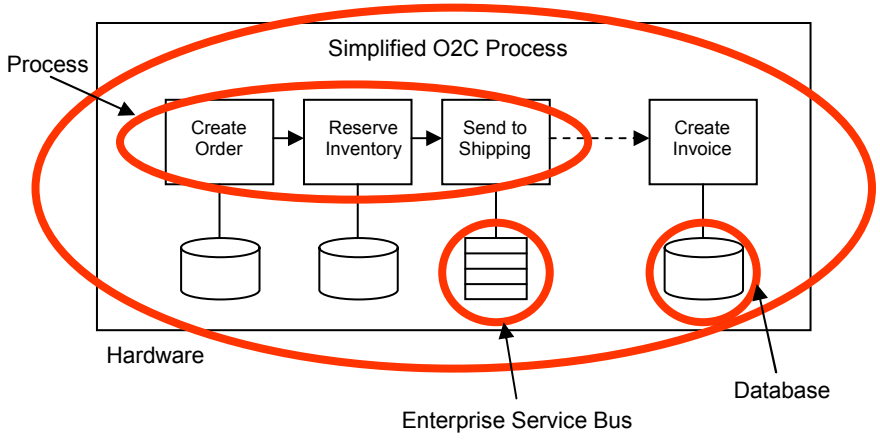
4. How Does Transaction Integrity Affect Your Business?

In order to better understand the importance of transaction integrity, let's look at a typical SOA process and how it is affected by the need for transaction integrity. Order-to-Cash (O2C) is a prototypical SOA process because it integrates multiple applications, including order management, credit management, billing, and collections. A successfully implemented O2C process allows an organization to track an order from its entry, through the shipping of the products, and to the collection of funds. The process allows the organization to view outstanding orders and follow up on orders that have not been shipped. After the order has shipped, the organization can use the process to monitor when the customer will pay the invoice. By integrating order-to-cash as a unified process instead of relying on a sequence of un-integrated applications, the organization gets a single view with which to follow an order through its lifecycle.

This single view requires that the integrated applications work in concert. However, the applications touched by the O2C process are otherwise independent and have separate sources of data. As a result it is essential for the process to ensure transaction integrity throughout its duration. For example, when an order is shipped, inventory must be adjusted accordingly to reflect the new stock level. When a payment is collected, both the account balance and the organization's cash balance must be updated. If a customer returns an order, both the account balance and inventory must be adjusted. Ensuring that these synchronized updates occur regardless of business or IT failure requires transaction integrity.

The Order-to-Cash process requires transaction integrity at all platform layers in order to succeed. While recording the order and reserving the inventory, the process must maintain a transaction that ensures that either both database updates succeed or both fail. After these steps the process must send the order information to the Shipping department. Since this step may take several days to complete, the process must enqueue the order information. With the order on a queue, transaction integrity means that the process instance and the message will not be lost until the order has been shipped, even if the server goes down. After the order has been shipped, the process must create an invoice. Transaction integrity in the database can ensure that the invoice refers to an existing customer and that the customer charge is in the range of valid values. Finally, the hardware contributes to transaction integrity by providing a cluster of servers that can retain transaction states even when one of the servers goes down.

Transaction Integrity in an Order-to-Cash Process



Without transaction integrity throughout the entire SOA platform, the benefits of an integrated order-to-cash process become serious business risks. Instead of improving order fulfillment accuracy, an O2C process without transaction integrity could result in higher order fulfillment error rates. Instead of reducing receivables and speeding collections, an O2C process without transaction integrity could result in inefficient collections and high costs for dispute resolution. Without transaction integrity, the organization could lose the knowledge of profitable and unprofitable customers that helps them target customers more effectively.

How could transaction integrity make the difference between an order-to-cash process that improves information, profit, and customer loyalty and a process that provides unreliable information, reduces profit, and drives away customers? IBM makes the difference through its comprehensive transaction integrity.

5. How Does IBM Provide Transaction Integrity?

IBM provides transaction integrity throughout its SOA platform, including the business process, enterprise service bus, database, and hardware components. Not only does IBM ensure that each element of the platform can recover from a failure, but it also makes them so reliable that they are unlikely to fail in the first place.

5.1 Integrity in the Business Process and Its Services

The business process can invoke several services, each of which will update its own system resources. In order to ensure that the process updates all of these resources consistently, it must create a transaction that spans the calls to each of the services. With a transaction in place, either all resources will be updated if the process is successful, or none will be updated if the process fails. The process engine can commit or rollback the transaction by coordinating with the service's resource manager, such as a database. The process engine can communicate with resource managers when they support the standard protocol for coordinating transactions.

However, some resources cannot be controlled by the process engine in this fashion. Such steps require a separate "un-do" step to restore the application to a consistent state. For example, SAP BAPI's do not support the standard transaction protocol, so they cannot be rolled back. Physical steps typically require a physical un-do step; for example, picking an item from inventory can be un-done by returning the item to the shelf. IBM supports the "compensating" steps in WebSphere Process Server by allowing the developer to assign an "un-do" step to each step in the process that cannot be rolled back automatically. Other vendors require the developer to create separate process logic to keep track of successfully completed tasks and perform the associated compensating tasks. In addition to the transaction and compensation support, IBM provides transaction integrity for long-running tasks by defining timeouts for human-based tasks so that someone will be alerted to take action if a task is not completed within a specified amount of time.

By supporting process transactions and compensating tasks, IBM provides transaction integrity to the business process layer and its services for both short and long-running business processes. With a long-running process, transaction integrity also requires that the process instance will persist along with its data and state until the process has completed, regardless of the time it takes to complete it and any server outage that may occur while it is running. In

addition a process may become long-running by invoking asynchronous services, such as services invoked by putting a message on a queue. For such a process transaction integrity will guarantee that the message is delivered onto a queue successfully for the transaction as a whole to succeed. The application then must rely on the transaction integrity of the enterprise service bus to guarantee the delivery of the message and the completion of its intended activity.

5.2 Integrity in the Enterprise Service Bus

With SOA applications, a business process invokes services as a way of re-using existing application function instead of re-creating it. However, using another application's services makes the process vulnerable to changes in the function, format, and location of those services. An organization can use an enterprise service bus (ESB) as a way of de-coupling the business process from the service, converting that vulnerability to flexibility to adapt to change. To provide that flexibility, logic within the ESB performs transformation to adapt to the service's message format or protocol and routing to find and invoke the desired service. Further adding to its flexibility, an ESB based on an enterprise messaging solution like WebSphere MQ can provide asynchronous, store-and-forward, and publish-and-subscribe message delivery.

When applications invoke services through an ESB, it is harder for them to control the integrity of these de-coupled resources. For example, if a customer enters an order through an application's front end, how can the organization guarantee that the order will reach the appropriate backend applications and in the required format? If there is a failure somewhere within the Bus, can the organization monitor the message flow so that no order is lost?

In order to ensure that service calls that flow through it will update resources reliably and consistently, the ESB also must contribute transaction integrity. IBM's WebSphere MQ provides enterprise messaging that runs on a wide variety of platforms and offers guaranteed asynchronous message delivery with full transaction support. This support means that if a message server goes down, the message will not be lost. If a target message server is not available, WebSphere MQ will hold on to the message and deliver it when service is restored. If WebSphere MQ sends messages to several targets, it will guarantee that all of them will receive them or none will, retaining transaction integrity. With WebSphere ESB a message flow that invokes a service has a built-in retry capability, improving the quality of service for the flow. IBM's ESB further enhances transaction integrity by offering the guarantee that a message is delivered once and only once.

After a business process puts a message on a queue and the ESB delivers it to its final destination, an application can process the message and perform the desired function; for example, inserting an order into a database. The application can use the ESB's transaction integrity to ensure that the message is not removed from the queue until the application successfully has completed its task, thereby guaranteeing that important data will not be lost.

5.3 Integrity in the Database

Since most application data ultimately resides in a database, the database is an essential element of transaction integrity. IBM's DB2 plays its part in transactional integrity by participating in process-initiated transactions as well as providing additional data integrity. Semantic integrity and referential integrity ensure that database tables have correct and consistent data. For semantic integrity the database ensures that keys are unique and specified fields have values. For referential integrity the database ensures that tables that refer to other

tables use data that is consistent across them. The database also can enforce rules regarding the values of specific fields; for example, by requiring that an employee salary is within a certain range. When a business process or service attempts to update the database, DB2's semantic and referential integrity will provide additional assurances that the database will be updated accurately and consistently. In addition, when the database fails DB2 can ensure that data returns to a consistent state by rolling back data. With large applications with many users, DB2 can use table and row-level locks to ensure that concurrent access does not result in inconsistent or inaccurate data. By adding reliable information delivery across the enterprise using Information Server and Master Data Management, IBM elevates a database's basic transaction integrity to an unrivaled "Information Integrity."

5.4 Integrity in the Hardware

System z's unique advantages over distributed systems make it a key component of an enterprise ready solution. By leveraging Parallel Sysplex and the Coupling Facility (high speed hardware interconnects that link applications on different processors together), System z optimizes the performance and reliability of SOA platform components such as WebSphere Application Server and DB2. Sysplex provides greater flexibility to distribute workloads across servers, delivers more efficient use of shared resources, and ensures reliability if a server fails. In addition, System z can provide 99.999% uptime (down less than five minutes per year) through built-in redundancy of components, hot pluggable replacement parts, remote repair, and failure prediction. System z further helps to ensure transaction integrity by using z/OS's Recoverable Resource Services to manage transactions across resources like CICS, IMS, WebSphere Application Server, WebSphere Process Server, and WebSphere Enterprise Service Bus. Like System z, System p employs LPAR's (Logical PARtions) to allow different workloads to share processors, memory, and I/O. This sharing results in a greater reliability for transaction integrity, and also improves hardware utilization. Other SOA vendors do not optimize their software to leverage the strengths of the hardware as does IBM with System z and System p, and thus they cannot deliver this degree of transaction integrity at such an effective cost.

6. IBM is the Leader in Transaction Integrity

Transaction integrity is the key to making SOA applications reliable and scalable in enterprise environments. Transaction integrity ensures that an SOA application that integrates services from separate applications can update these applications consistently. Without consistent data, SOA applications become a liability to a business instead of a way to innovate and create business value.

Transaction integrity, along with performance and scalability, make an SOA platform today's transaction processing environment. With its forty year history of providing a platform on which to run mission-critical applications, IBM is the leader in transaction processing. IBM has taken these years of experience in running business-critical applications and has applied it to its distributed platforms and WebSphere products. These WebSphere products, including WebSphere Process Server, WebSphere ESB, WebSphere Message Broker, and WebSphere Application Server, continue to take inspiration and technology from the transaction processing abilities of CICS and IMS. IBM also infuses transaction integrity into its database, DB2, and its hardware platforms, System z and System p, so that each element of an SOA application contributes to its overall reliability. As a result of this comprehensive approach, IBM's

unparalleled transaction integrity makes it the platform on which businesses can trust their SOA applications.

IBM Corporation
Software Group
Route 100
Somers, NY10589
USA

Produced in the United States
January 2008