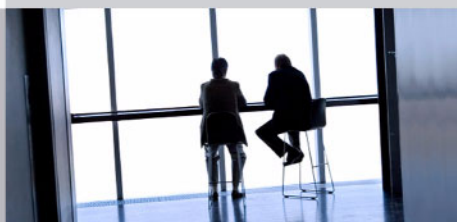


Transaction Processing: Past, Present, and Future



Redguides
for Business Leaders

Alex Louwe Kooijmans
Elsie Ramos
Niek De Greef
Dominique Delhumeau
Donna Eng Dillenberger
Hilon Potter
Nigel Williams



- Transaction processing in today's business environment
- Business challenges and technology trends
- Role of the IBM mainframe



Foreword

Thriving in the Era of Ubiquitous Transactions

I am absolutely obsessed with the concept of Transaction Processing because it is the very essence of every company on earth. A business is not in business unless it is successfully processing transactions of some kind. IBM® has been investing in, leading, and inventing Transaction Processing technology for as long as there has been programmable computing. It is no coincidence that the world's economic infrastructure largely rests on IBM technology, such as mainframes, and fundamental financial activity occurs through CICS®, IMS™, and WebSphere® transaction engines.

IBM continues to evolve the entire compendium of intellectual property as it relates to Transaction Processing so that companies like yours will have the sophisticated technology required to not only support but thrive in the new world of federated environments and interconnected enterprises, and Cloud services and mobile commerce with certainty and control. This melding of modularity and service-oriented techniques together with Transaction Processing science will provide every business the ability to deliver straight-through processing at the lowest possible cost and highest level of transactional integrity.

Steve Mills

IBM Senior Vice President and Group Executive, Software and Systems



Executive overview

The role of IT is becoming more prominent in people's daily lives and we are becoming increasingly dependent on computers. More and more business transactions are being automated, for example, ordering a book at an online bookstore or transferring money to a bank account in another part of the world. No matter the type of transaction, we want it to be accurate and we want to have no doubts about its outcome.

Transactions are also becoming more complex, driven by new ways of conducting business and new technologies. Smartphones now allow us to conduct transactions anywhere and at anytime. Technology paradigms, such as Web 2.0 and business event processing, enable businesses to increase the dynamics of a transaction through instrumentation that captures events, analyzes the associated data, and proactively interacts with the client in order to improve the customer experience. To adapt to the increasing volume and complexity of transactions requires an ongoing assessment of the current way of supporting transactions with IT.

No matter what your business is, you need to ensure that your transactions are properly completed with integrity. Wrong or incomplete results can adversely affect client loyalty, affect company profits, and lead to claims, lawsuits, or fines. Companies need to be able to rely on computer systems that are 100% reliable and guarantee transaction integrity at all times. The IBM mainframe is such a platform.

Clients that have been using an IBM mainframe are conscious of its added value. For this IBM Redguide™ publication, we surveyed a number of companies that use the IBM mainframe and we asked them to tell us its most distinguishing qualities. They answered unanimously "reliability, availability, and scalability." They also do not see an alternative for running their mission-critical business workloads other than the IBM mainframe.

When we surveyed our clients, we also asked them about the future. Clearly, major future trends demand significantly smarter, faster, and bigger transaction processing systems than we have today. Some of these trends are the availability of new computing paradigms, continuing growth of the mobile channel, further integration of organizations, massive growth of unstructured and uncertain data, and increasing complexity of IT systems.

IBM continues to invest in mainframe technology leadership, which protects years of client investments on this platform. Today, well-known transaction processing (TP) middleware, such as the IBM CICS, IBM IMS, IBM z/TPF, and IBM WebSphere Application Server products, and also solutions for service-oriented architecture (SOA) and business process management (BPM) are available and fully optimized on the IBM mainframe running the mission-critical business workloads of many companies the world over.

In 2010, IBM announced the IBM zEnterprise® system introducing a hybrid computing platform that combines the traditional IBM mainframe capabilities and the ability to use IBM blade servers, managed by a single management software. With zEnterprise, you can significantly reduce the complexity of your IT and achieve better service levels, while continuing to benefit from traditional mainframe strengths in transaction processing.



Today's business depends on transaction processing

The global economy is made up of an immense number of business transactions, usually involving the exchange of goods or services for money. These business transactions tend to be short and repetitive. Common characteristics include concurrent access by many users, through multiple interfaces, to shared information.

In this chapter, we review more closely what we mean by a *transaction*, we look at the evolution of transaction processing, and we then consider how it differs from other types of information processing.

What is a transaction

When using the term transaction, clarifying whether you are talking about a *business* transaction or an *information technology* (IT) transaction is important. The difference is a matter of perspective. When you walk into your neighborhood store and buy a book, that is a business transaction. Likewise, these transactions are all business transactions: when you submit an insurance claim, get your oil changed, see a doctor, pick up your prescription from the pharmacy, pass through a tollbooth, fly on an airplane, or even pay your electricity bill.

However, in today's world, where almost everything we do is backed up with computer automation, nearly all of these business transactions drive one or more corresponding activities in the information system. When you buy a book, the cashier enters the amount in the register, rings up a total, and collects your money. That activity might send a number of requests to an information system: to take that book out of the inventory database, approve your credit card exchange, update the store ledger, capture your store-loyalty status, and credit the store clerk with the sale. Each of these activities can be thought of as an IT transaction. Or, all of them together can be thought of as a transaction. Or, some of these activities might be divided into smaller transactions. It all depends on how the application is written.

Why transaction processing is important to the business

What is clear is that the integrity of the business relies heavily on the integrity of these transactions in the information system. If any one of these activities fails to do its job correctly, the business will be out of balance. If the inventory database is not updated correctly, the store will think it has more books on hand than it does. What happened to the book? Did it get misplaced? Did someone steal it? Was it mislabeled? Likewise, if the credit card is not debited properly, the store could be out of money. If the loyalty status is not updated correctly, the customer will miss out on the rewards they deserve. If the ledger is not updated, the cash register will be out of balance at closing.

Any one of these situations is not necessarily detrimental on its own. Any of them can be reconciled later. A thorough inventory of the store and scan of receipts can reveal the legitimate sale of the book. The store can resubmit the credit card transaction. The customer can dispute the loyalty status. But, having to go down that path is painful for customers. It consumes time and resources, can lead to customer dissatisfaction, and leaves the business in an uncertain state for a while. And, what if the scenario is more critical? What if it is a large purchase like a car? What if the purchase amount is large enough to cost the business millions of dollars in lost interest? What if it is the difference between two trains passing safely or colliding on the same track? In the end, how much effort you apply to ensuring the integrity of your business is a critical design choice.

Transaction processing middleware

It is possible to write stand-alone applications to perform transactions in shared, multiple-user environments; however, doing so is complex. Application developers would need to address issues, such as the communications with the user interface: ensuring that only authorized users access the system, preventing concurrent changes, handling failures and backing out partial updates, managing the database connectivity, and numerous other complicated tasks—each of which would have to be rewritten for each new application. This is why we have transaction processing (TP) middleware (see Figure 1).

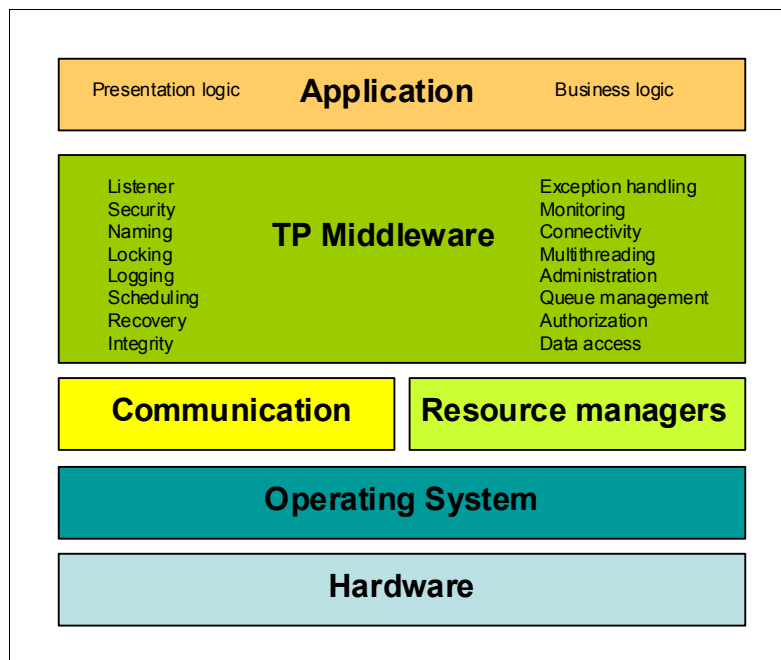


Figure 1 Services provided by TP middleware

For this IBM Redguide publication, we surveyed a significant number of companies that run IBM TP middleware. What we found is that TP is as critical today as it has ever been in the past, and that today the nature of transactions is changing faster than ever. These changes present new opportunities (more channels over which to sell, for example), and of course, new challenges, particularly in the areas of security, scalability, and transaction integrity.

Transaction volumes

Without exception, our customers are experiencing a rapid increase in the rate of transactions. This increase is largely driven by the two billion web users and the explosion in the number of mobile devices now used to initiate transactions. For example, reservation systems that previously supported only travel professionals are now directly accessible by anyone on the Internet. As a result, the number of users has increased from tens of thousands to hundreds of millions.

"We run several thousand transactions per second now that our systems are opened up to the Internet and mobile devices." **(Misha Kravchenko, Vice President Information Resources, Marriott International)**

Nowhere have these trends been more evident than in the finance sector, where credit card companies now need to instantly authorize billions in purchases from merchants around the world, and banks must provide their customers access to banking and investment accounts from anywhere at any time of the day. As emphasized by many of our interviewed financial clients, transaction growth comes from customers needing to know their account position in real time. These trends have led to unprecedented transaction rates, as witnessed below.

"We now have a peak workload of 20 thousand transactions per second." **(Marcel Däppen, CTO for zEnterprise Systems, UBS WM&SB)**

Transaction processing: An evolution

Transactions have been around since the first person used the "barter" system for food or tools. The integrity of that transaction was based on a handshake and trust. The modern transaction is also completed with a handshake but more likely one based on digital cryptography, and trust is based on the integrity of the TP system that processes the transaction.

To understand how transactions are being processed today compared with the past, we introduce you to Waldo. Waldo is 40 years old, married, and has one son (Waldo Jr.). He has a checking account, a savings account, and a stock portfolio. Figure 2 on page 8 shows how Waldo's transactions 20 years ago compare with the way that Waldo Jr. conducts his transactions today.

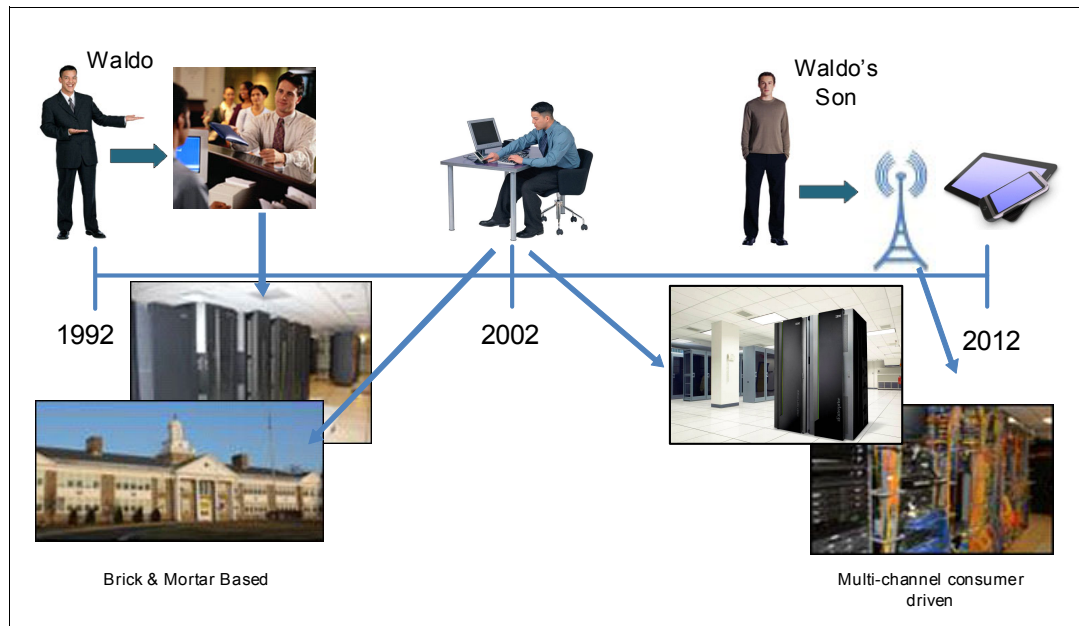


Figure 2 Evolution of transactions over the past 20 years

Transactions in 1992

In 1992, Waldo executed most of his transactions in person, through the mail, or on the phone. ATMs were becoming more popular and were starting to provide a wider range of services, and debit cards were just being introduced. The introduction of Internet banking or online sales sites, such as Amazon or eBay, would not be introduced until the middle of the 1990s. A cellphone was the size of a “brick” and cost thousands of dollars. Brick and mortar banks, retail stores, travel agents, and car and insurance agencies were still heavily used for conducting day-to-day business transactions.

A typical business transaction would involve Waldo interacting directly with a teller, sales person, or business representative. That person would employ a user device (such as a cash register, teller machine, or terminal) connected to a back-end system, where the transaction would be processed. A single business transaction would usually result in one or two back-end transactions. The integrity of that transaction was controlled at the back end and the dedicated connection to the terminal where the transaction was entered. These transactions were also limited to normal business hours. Even booking travel plans might include a visit or call to a travel agent who looked up information in printed catalogs or daily reports before using the “system” to actually verify that the reservation could be made, and then make it. Credit cards and checks were used for large purchases, and cash was used for smaller daily purchases, for example, lunch, coffee, and so on.

Transactions in 2002

In 2002, Amazon and eBay were barely five years old, yet they were already having an impact on the way we were doing business. Waldo had a home computer on which he could access the Internet, and Waldo’s son was growing up with a computer, and network connection, as a normal device in his home. Waldo was conducting a limited number of transactions, mostly involving small purchases at online shopping sites.

When Waldo visited a shop, the sales person would use a web browser to navigate through the new web interface of the back-end system, where the same transactions that were previously executed from terminals would be processed. A single business transaction would still result in a few back-end transactions. These transactions were now available outside normal business hours. The use of debit cards was becoming more prevalent and micropayments of less than a few dollars were becoming the norm. There was also an increase in online searches as consumers began to take advantage of the Internet to look for bargains.

Waldo visited his bank less often because his bank offered a limited set of services (mostly balance, posting inquiries, and transfers) through the Internet banking site. He always withdrew cash from an ATM because ATMs were now present at all the locations that he frequented. The changing habits of Waldo and others meant that the number of Internet banking transactions increased gradually, but at the same time the ratio of transactions doing an update decreased. For one of our interviewed clients, for example, updates represent less than 15% of the Internet transactions.

Transactions in 2012

Waldo's habits have changed a little over the last decade. For example, he is now more likely to use his debit card rather than cash for even small purchases, and his notebook for conducting transactions, such as shopping, travel plans, and banking (for example, account transfers to Waldo Jr.). He expects the same instant access to the Internet sites where he conducts transactions as he expected for ATM transactions in the past. He also expects the Internet banking site to provide a full financial summary of all his accounts, with drill-down and search capabilities. This dramatically increases the volume and sophistication of transactions.

Waldo Jr. is much more adept at using his smartphone, tablet or laptop than his father. He uses social networking sites, blogs, and wikis for interacting and collaborating with his friends. He also uses these tools to help with decisions about where to shop, what to buy, and where to bank. If he is in a physical retail store checking on a product he is interested in buying, he might scan the barcode and compare online prices to the one listed in the physical store. The "e-reputation" of a company is of significant importance to Waldo Jr.

Waldo Jr. also expects a seamless navigation experience. When something is not working properly he wants to chat to an advisor without having to call the Call Center. He expects his bank to be preemptive, for example, to warn him if an impending transaction will cause his account to become overdrawn (so he can call Waldo) and is more than happy to switch banks if these services are only available elsewhere.

Waldo's changing habits and Waldo Jr.'s expectations mean that a single business transaction today is likely to be initiated from a mobile device and often results in a higher number of IT transactions distributed across a larger number of systems, some of which are automatically initiated to provide the additional service that the empowered customers are demanding. As an example, when Waldo Jr. and his friends need to book a room at a hotel, they do not use a travel agent, they use the Internet, generating thousands of search transactions for every room actually booked.

"The number of look-to-book transactions for our hotel chain has changed from 8 - 10 searches for one booking, to potentially thousands of searches for one booking." (Misha Kravchenko, Marriott International)

In many cases, web sites visited by Waldo Jr. automatically “upsell” to him, based on his profile and buying habits. Waldo Jr. also uses applications on his smartphone to help him check in at the airport, buy a coffee, and receive a discount at the “fast-food” counter, and all of these transactions are tied into a loyalty customer program. Transactions are also no longer initiated only by the customer, as businesses push out message notifications related to offerings, account balances, payment due dates, and much more.

Transaction processing: Why it differs

Transaction processing differs from other types of IT processing (business analytics, web collaboration, high performance computing, and others) in a number of significant ways:

- ▶ Most often, a financial element exists, and sometimes the dollar amounts can be significant.
- ▶ A real-time element exists, and normally a user waits for a response (faster is always better).
- ▶ Transactions run in parallel and often need access to the same data.
- ▶ When a transaction runs, it must run in its entirety, and databases must be consistent at all times.
- ▶ Transactions must be secure.

Ever increasing customer expectations mean that availability of the online transaction processing (OLTP) system is more important than ever. In the past, OLTP had clear “open” and “closed” times, but this case is less common today. The expectation for anytime-access to online shopping, reservations, and Internet banking sites means fewer and shorter windows for maintenance, which in turn means that TP middleware must provide the mechanisms for a continuous runtime environment.

“The currently defined maintenance window is actually not acceptable anymore. We have projects in place to reduce and eventually eliminate planned outages.” (ABN AMRO Bank)

The unprecedented levels of user access to online transactions expose security threats that need to be managed. Security must be an integral part of the TP system so that users accessing the system are authenticated and identified, and only authorized users can access transactions. Confidentiality and integrity mechanisms are required to prevent an unauthorized party from obtaining or changing transaction data that is sent over the network.

OLTP versus batch processing

OLTP is a style of computing that supports interactive applications and where work is divided into individual operations. By contrast, *batch processing* is a style of computing in which one or more programs process a series of records (a batch) with little or no action from the user or operator.

Consider the following situation. A large grocery retailer has many stores distributed throughout the country. As customers check out, each grocery item is priced and added to their item bills, and the total is tallied and billed. Rather than sending every grocery item and total shopping bill as a transaction back to the central site at the time the customer checks out, the billing activity is resolved in the local store, and all of the customer transactions are stored in a local file. The store then sends the file back to the central site at the end of each day.

The central site then needs to process all of those transactions that night as a batch job. The batch job might compute loyalty points for the customers who shopped that day. It assesses inventory displacement throughout the day to determine what goods need to be replenished for the next day. And, among other things, it might analyze buying trends to inform the purchasing department to help with vendor negotiations.

Although batch processing has been around since the earliest days of electronic computing in the 1950s, it still has much relevance in modern computing systems because of its economies of scale over OLTP. This is primarily due to the way that multiple transactional updates are done in a single unit of work rather than a large number of individual units of work. For many of our surveyed clients, losing their TP systems during the overnight batch processing when all the account reconciliations are done is the most serious potential business impact.

The demand for 24-hour access to online transactions often means that batch processes must be run in parallel with online transactions. Because the online transactions and batch processes usually require access to the same data, collocation of the OLTP systems and batch processes is a common deployment model. Near real-time batch processing is therefore becoming close to indistinguishable from OLTP.

The importance of getting it right

Transaction processing workloads (both online and batch) are normally the most critical workloads run by an IT organization. No hotel wants to lose business because the online reservations system is not available. No bank wants to suffer the damage to its reputation by being associated with having ATMs that are out of order, or an Internet banking site that does not respond. No shop wants to be out of stock of items because the overnight stock processing batch run failed. For these reasons, TP workloads have to run on the most robust platforms available.

Below are some views from our surveyed clients on the importance of system availability:

- In the case of Marriott International, which is one of the largest hospitality companies in the world, the demands on its hotel reservation system are enormous. Transaction throughput must be rapid and unfailing, 24x7, from anywhere. If problems arise, failover must be swift and absolute.

"We estimate the cost of our reservation system being unavailable at three thousand dollars per second." (Misha Kravchenko, Marriott International)

- In the case of financial institutions, consistent and timely processing of payments can create business value and a competitive advantage.

"To provide unique value to the business, UBS processes SWIFT^a messages within strict time limits." (Marcel Däppen, UBS WM&SB)

a. Society for Worldwide Interbank Financial Telecommunication

- Based on comments from one of our other major financial clients, outages to payment systems at critical times of the day (for example, the CHAPS¹ cut-off time) could cause direct financial loss because of fines and compensation claims, but equally important are the intangibles in the loss of reputation and affected customers taking their business elsewhere.

¹ CHAPS (Clearing House Automated Payment System) offers same day sterling fund transfers.

Another important distinguishing feature of TP workloads, compared to other types of workloads, is the requirement to maintain absolute transaction integrity. We describe this feature in more detail later but first we clarify what we mean. *Transaction integrity* is the assurance that the results you expect are being achieved, whether you are using your mobile phone to transfer money to an account, reserving a hotel room, or paying for a meal at a restaurant. If for some reason, a transaction does not complete successfully, the user should be informed and the databases should be left in a consistent state. Transaction integrity supports virtually all operations of the organization, and it is a vital concern to virtually every business.

“If you’re updating an account in a database, it doesn’t matter whether you’re in IT and responsible for the applications being run or if you’re in a line of business and are responsible for those business transactions. What you want to know is that the update did in fact occur, and that it has been recorded correctly.” **(Rob High, IBM Fellow, Vice President and CTO for Watson, IBM Software Group)**

In the old days, the burden of resolving the design choices for ensuring integrity in your system was left entirely to the application developer. However, we are now able to put more of that burden on the middleware, underlying operating system, and hardware – relying on the infrastructure to achieve maximum integrity. Many of our interviewed clients told us that they rely on middleware so extensively that they have effectively given up responsibility for transactional integrity to the middleware. The developer can now choose the transaction pattern to implement based on the transaction integrity requirements.

Scale-up or scale-out

Traditionally, TP systems are *scaled-up* by adding resources to a single node in the system. This model has the advantage that resources (CPUs, memory, storage, and so on) can be shared by the operating systems, TP middleware, and applications. It also simplifies the task of maintaining transaction integrity because there is a single source of data. This model is most efficient for high-volume workloads, which need absolute assurance for read and update integrity at all times.

An alternative approach is the *scale-out* approach, which means adding more nodes to a system, such as adding a new computer to a distributed software application. Because of the share-nothing characteristic of this approach, there is a need for data partitioning or data replication. Such systems can be used efficiently for high-volume read-only workloads, but are also used for workloads that do not need absolute time-critical data consistency.

There are trade-offs between the two models. But whichever model is used, TP systems have to handle high volumes efficiently, maintain transaction integrity, be secure, avoid downtime, scale gracefully, and be easy to manage. It is a tall order.



Business challenges and technology trends

“Today’s business depends on transaction processing” on page 5 describes how the nature of IT transactions changed over the last 20 years. In this chapter, we look at the key business challenges that our surveyed customers are facing today, and we also review the major technology trends that are affecting their TP systems.

Transaction processing: Business challenges

Based on our interviews, the following major business challenges are affecting the TP systems of our clients:

- ▶ Need to offer new services that attract customers and maintain their loyalty
- ▶ Challenge to react quickly to competition and new business opportunities
- ▶ Control of risks and the ability to respond to regulatory scrutiny
- ▶ Requirement to build partner relationships, and manage acquisitions and mergers
- ▶ Pressure to reduce costs

“The major business trends impacting our TP systems are increasing customer expectation, the need for quicker delivery of applications and more partner integration.”
(China Merchants Bank)

We also recognize a common denominator among these challenges, which is the continuously increasing amount and variety of data that must be maintained and analyzed to be responsive. The explosion of data is not only caused by the increasing number of transactions but also by the automated collection of information from browser, social media sites, sensors, and many other sources.

Attracting new customers and maintaining their loyalty

Today, we all expect instant access to our banks so that we can find information, view and manage our accounts, and pay bills online. Most banks offer some form of mobile banking, and some banks now offer added value through budgeting and cash-flow applications, providing customers a total view of their accounts (even those with other institutions) through a portal or enabling online chat with customer service representatives.

The same increasing customer expectations extend throughout other industries, including travel, retail, and public administration. For example, the ease of use of an online shopping or hotel reservation system has an important impact on our purchasing habits.

"We try to provide a friendly and pleasant online experience to our customers and that also rewards them for their loyalty." (Misha Kravchenko, Marriott International)

And, we tend to be even more demanding when using internal systems, for example, investment traders demand instant access to their financial trading data.

Gaining new customers and achieving customer loyalty are key business challenges that are driving corporations to be more customer-centric, which in turn involves the collection and real-time analysis of more customer data.

Business agility and optimization

Being competitive was always a business goal, however, with the liberalization of markets and the added competition from emerging markets, the challenge is greater than ever. To succeed in today's competitive market, businesses need to simplify the architecture of their TP system to improve business agility, take out cost, and eliminate inefficiencies. The aim is to use real-time information to make better business decisions that optimize return on investment (ROI).

This ROI is witnessed in the travel industry, in particular where room occupancy is a clear measure of competitive efficiency. In the case of Marriott International, business intelligence software looks at member status, inventory status, and dynamic pricing models. Factors considered include whether the customer is staying over on a Wednesday night or through a weekend, and whether the customer is a Platinum member. On the room inventory side, systems also consider whether rooms in the area for which the customer is requesting lodging are in an undersold or oversold status. All of this system intelligence comes together in a "best price, best yield" scenario for both the hotel chain and the customer in less than one second.

"Right-fit pricing is integral to achieving and maintaining competitive advantage. The goal is to book inventory down to the last room available to maximize yield." (Misha Kravchenko, Marriott International)

A similar approach is witnessed in the banking and insurance industries. Financial institutions are trying to develop more agile operating models and product engines that allow them to offer product bundles based on a customer's specific requirements and buying behavior.

A typical banking product package includes a checking account, a credit card, Internet banking, and perhaps mobile banking. Such packages are marketed and sold as a unit. From the perspective of the bank, the advantage is additional sales, customer loyalty, and reduced operational cost. From the customer perspective, the value is in ease-of-use and having a pre-integrated set of products available to match their needs. Adding a differentiated pricing component significantly increases the attraction of the business case because optimized pricing on the packages can help drive sales and shape customer behavior.

The pressure to be the first to offer new services has pushed many organizations to implement packaged applications. Critically important, however, is that such packages should be easy to integrate with existing TP systems. In the case of one of our interviewed clients that uses specialized packages for fraud detection, anti-terrorism, and document handling, the client ensures that these packages are sufficiently flexible to enable integration with their core systems.

Risk and compliance

The challenge of risk and compliance is particularly relevant to the financial industry after the crisis of 2008 demonstrated in the clearest way imaginable that all banks must rethink the way that they measure, track, and control credit risk. Banks are challenged to improve their risk models and their approach to risk analysis, which includes more timely reporting on business risk throughout the organization, and more transparency in reporting risk.

Accurate and consistent risk information throughout business units prevents unexpected losses and civil or criminal penalties. The aim is to eliminate redundancies and reduce risk by improving the timeliness and quality of information that is available. Our interviewed clients stated that they have implemented an audit trail of who does what, and they stressed the need to externalize user privilege rights in a rules engine that has similar service-level agreement (SLA) requirements as the TP systems. Any program to reduce risk by increased data collection invariably has an impact on existing TP systems because much of the risk-related data is processed by these systems.

To remain compliant in the face of regularly changing rules that are imposed by governments and other statutory bodies, financial institutions must enable their TP systems with the sensors necessary to detect and intercept certain events. For example, for FATCA¹ compliance, institutions must develop events and rules-based systems to identify their FATCA-relevant clients and make decisions on which actions to take.

Within the new world of open systems and increased partner integration (see Partner integration), a multi-layered approach to security is required. An important component of such an approach is a hardened security appliance that can protect TP systems from Denial of Service attacks, off-load expensive security processing from the TP systems themselves, and satisfy the specific confidentiality and auditing rules to which they need to adhere.

Partner integration

Most of our surveyed clients have commented on the trend toward expanded partner integration to maximize the distribution channels for their products, or to subcontract parts of their supply chain. Companies that traditionally managed their own distribution channels are changing their business model and offering their services to external distribution channels.

¹ FATCA (Foreign Account Tax Compliance Act) requires Foreign Financial Institutions to identify their US accounts and report them to the US tax authority (the Internal Revenue Service, or IRS).

“One of our goals is to increase the number of distribution channels for our inventory.”
(Misha Kravchenko, Marriott International)

Straight through processing (STP) trading processes that remove the need for manual intervention, or interruptible business processes, today often span different companies. For interoperability, these processes need to be based on standard protocols. The demand for inter-company cooperation increases the need for flexibility within TP systems.

In addition to the trend towards corporate partnerships, the number of mergers and acquisitions is also changing the business landscape. Companies that were competitors become subsidiaries, sharing services and products. Companies that were subsidiaries break off from their holding company, thus creating a need to separate TP systems. Such changes in corporate structures can only be done quickly and cost-effectively if a services-based architecture and flexible infrastructure have been put in place.

Reducing costs

The IT industry as a whole is challenged with improving business flexibility and minimizing risk, but at the same time, CIOs are expected to hold to constant or decreasing budgets. Across the industry, there is a deep desire to eliminate expense where possible.

Over the past 15 years or so, the cost dynamics of supporting corporate IT infrastructures have changed significantly. In particular, we have seen hardware costs decrease, while people and software costs have increased. To control people and software costs, CIOs commonly view centralized TP systems as the future. Centralized infrastructures and processes enable shared services optimization that, in turn, provides economies of scale.

According to one of our interviewed clients, the overall cost of the service layer is greater than the process layer. The overall cost of the process layer is greater than the media access layer. Therefore, the optimal ROI is achieved through service reuse. Our clients also recognize that standardization is another key to cutting costs. Automation, where it makes sense, also helps lower costs within the enterprise.

Specialized systems, such as optimizers and appliances, can play a role in reducing transaction costs in the following ways:

- ▶ Off-loading the processor-intensive part of a workload, freeing up the general processors for other work
- ▶ Simplifying the integration tasks that are required when building multi-platform solutions, enabling new applications to be implemented more quickly and with fewer IT staff

Transaction processing: Technology trends

The business challenges outlined previously, in addition to the ever increasing pace of technology change, have led to a number of technological trends that are having a significant impact on TP systems today:

- ▶ Continued evolution to architectures based on service-orientation and web services
- ▶ Mobile solutions
- ▶ Implementation of Web 2.0 based technologies
- ▶ A new paradigm for application development

- ▶ Opportunities presented by business event processing and the encapsulation of business rules into flexible rules engines
- ▶ Ability to model business processes that have historically involved a lot of paper and manual intervention as automated flows that are managed by business process management (BPM) systems
- ▶ Growing importance of business analytics
- ▶ Emergence of hybrid systems
- ▶ Cloud-based services

Service-oriented architecture (SOA)

Application integration and partner interoperability have been made easier by the standardization of service-oriented technologies, such as web services, which enable loose coupling of applications. The use of web services to reuse existing information systems is a common theme among our interviewed clients.

“We have moved most of our services to stateless web services, which expose core transactions directly.” (Misha Kravchenko, Marriott International)

An approach that is based on service orientation provides a governed library of well-architected service and information building blocks that can speed up the development of new solutions. The value proposition is centered around business agility and alignment of business and IT design and delivery. This proposition remains the cornerstone for our clients in achieving the flexibility that they need to integrate their applications and do business with their business partners by using a clearly defined multi-channel architecture based on a core set of business services. The objective is to allow the same service fulfilment solutions to operate in support of any channel, keeping the business logic completely agnostic and separated from the channel handlers.

The use of open XML-based standards for integration means that existing TP systems need to be XML-enabled, either directly, or by using an intermediate gateway that converts the message payload from XML to the payload format that the existing TP program understands. This need has led to a variety of architectures based on SOA principles and standard implementations that include an enterprise service bus (ESB), which is the component that is responsible for connecting various client types with the traditional TP systems.

There is an inevitable performance cost overhead to loose coupling and XML-based data formats. The performance and security challenges of implementing such architectures often benefit from the inclusion of an integration appliance that can optimize service invocations, enable security, simplify application integration, and speed up the overall deployment.

Although SOA has been talked about for over a decade, it remains a significant part of the TP strategy of our clients. It is also the foundation of other trends and technologies, such as mobile and cloud.

“The use of web services is strategic for the bank.” (Marcel Däppen, UBS WM&SB)

Mobile

Without doubt, one of the most important technology trends over the last few years has been the rise of the smartphone. The speed of adoption of mobile phones and other devices compares similarly with previous technology adoptions, including TV, radio, and the Internet, and means that they will soon overtake PCs as the most common web access device worldwide. Gaining competitiveness through mobile solutions is ranked second by CIOs² after business intelligence and analytics.

The effects of this mobile revolution on the TP systems of our clients are an increasing number of transactions, changing transaction patterns, and a greater fluctuation in the rate of transactions. Many banking customers, for example, check their card balance using a mobile device before a transaction, after a transaction, and even during a transaction.

"We expect more growth coming from the mobile channel and we also foresee a workload increase from new self-service applications." (ABN AMRO Bank)

Organizations that have well-established multi-channel architectures have been better placed to offer mobile access to their TP systems. A channel integration approach based on a set of common reusable services, standard security models, and an ESB offers the best flexibility for growth and future integration of new channels. Figure 3 shows an example of such a multi-channel architecture within a banking context.

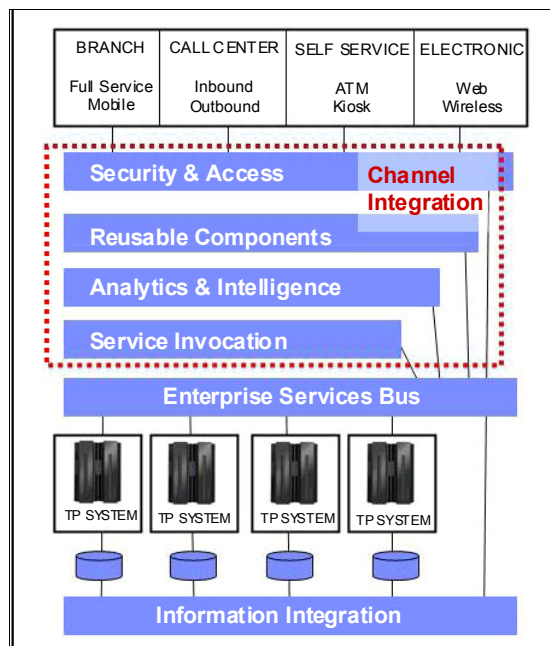


Figure 3 Multi-channel integration of TP systems

Web 2.0 versus Web 1.0

Web 2.0 differentiates itself from earlier Web 1.0 applications where every user of an application consumes and interacts with the same information presented as the author intended. In a Web 2.0 application, users can customize their individual experiences to their specific needs. The user can select what information is to be displayed and how to display it.

² Source: 2011 CIO Study, Q12: "Which visionary plans do you have to increase competitiveness over the next 3 to 5 years?" (<http://www.ibm.com/services/c-suite/cio/study.html>)

Furthermore, the user is not restricted to one information source but can pull information from various locations and mash them together (in a “mashup”) however the user wants.

In Web 2.0 applications, HTTP is commonly used as the protocol for naming and taking action on the resources, and the resource details are usually returned as XML or JavaScript Object Notation (JSON), which is a lightweight data-exchange format that is less verbose than XML.

An ever increasing number of people use Web 2.0 style applications when they use their smartphone, or visit a social networking or blogging site. In a recent survey, 60% of organizations see revenue benefits in Web 2.0 through attracting new customers and retaining existing ones, and 30% see savings that can be made in account management and servicing costs through use of these technologies.

Web 2.0 and related technologies have a particular role in the world of banking in helping to rebuild customer confidence, particularly in providing customers with a more integrated user experience. Figure 4 shows several new types of services that banks are offering receptive clients, like Waldo Jr.

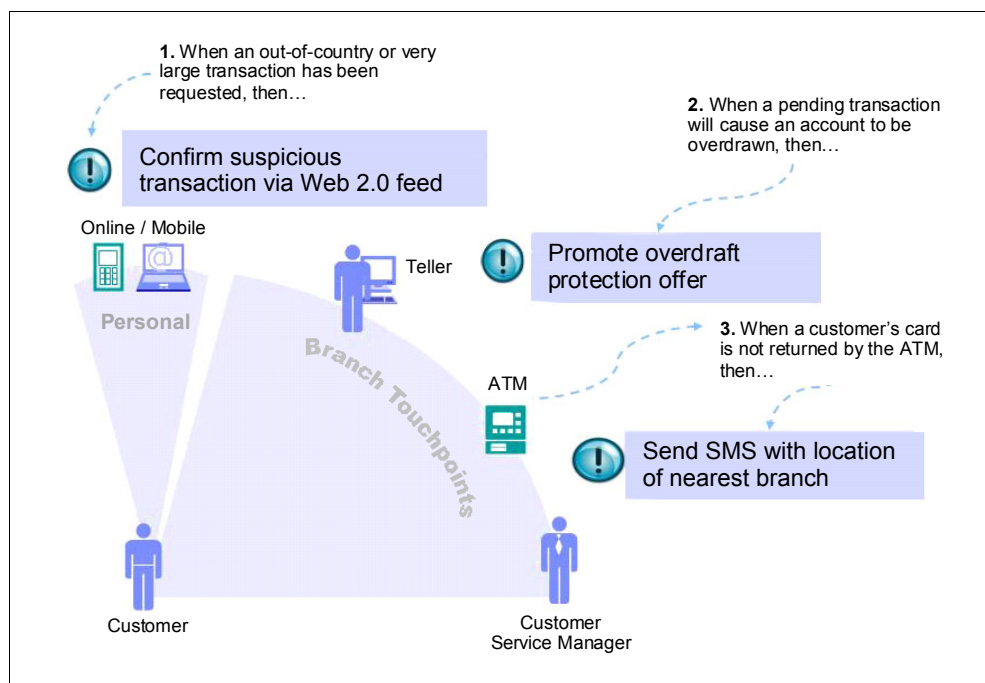


Figure 4 New banking customer interaction scenarios

The figure shows the following scenarios:

1. In the first scenario, large or unusual transactions are captured and highlighted to Waldo Jr. through a Web 2.0 feed. This way makes Waldo more aware of the financial position of his accounts, also improves fraud detection, and therefore potentially saves the bank money.
2. In the second scenario, Waldo is warned in advance of a pending transaction that will cause his account to become overdrawn. This way allows him to make the necessary arrangements, for example, to accept an offer of an overdraft protection offering.
3. In the final scenario, if Waldo's credit card is not returned by the ATM, a Short Message Service (SMS) can be sent informing him of the location of the nearest branch.

These scenarios are real and each of them is being implemented by at least one of our surveyed banking clients. The major impact on their TP systems is that transactional data must now be seen as a resource that can be accessed in a RESTful³ style by exposing feeds or collections of data that can be combined with data from other sources and presented to the customer as an integrated package.

New paradigm for application delivery

The need to offer new services to remain competitive and retain customers is changing the way that applications are developed and deployed. Lines of business are no longer willing to accept long development cycles and are more likely to take matters into their own hands. Figure 5 shows how a new model of application delivery is speeding up delivery times and lowering costs.

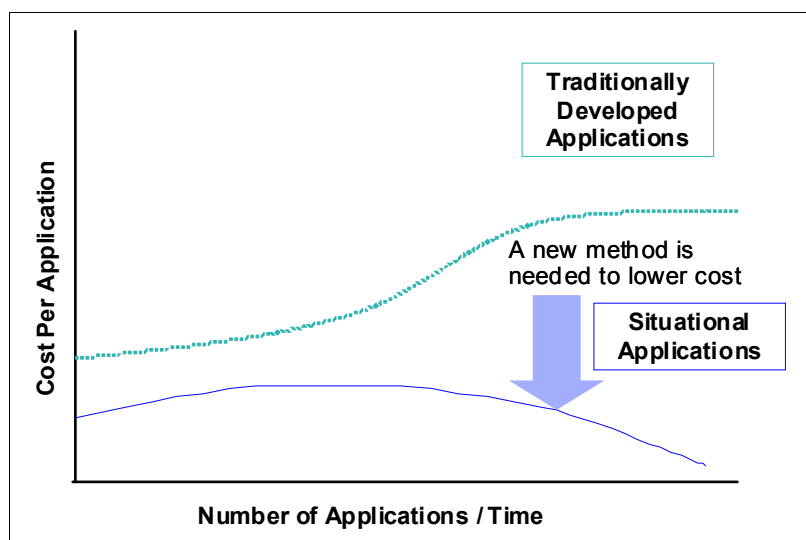


Figure 5 New model of application delivery

The following ways are being used by our surveyed clients to speed up application delivery and cut costs:

- ▶ Taking full advantage of the reuse of existing TP assets to avoid the redevelopment of existing business logic
- ▶ Simplifying the development cycle and speeding up application delivery by minimizing the need to write code
- ▶ Supporting self-service development of situational applications that have a limited life and user community, thus freeing up central IT staff to work on more strategic initiatives

The use of an integrated development environment (IDE) can help companies to accelerate time to market with higher quality solutions, while reducing the costs of maintenance over the lifetime of an application. An IDE can be used to design and construct new TP applications, web applications, and integrated service-based composites quickly and efficiently.

The advent of agile development and situational applications (mashups, new mobile applications, and so on) has an unpredictable impact on TP systems of our clients in that it is difficult to predict the additional demand that will be generated as a result of the launch of the new offering. This situation increases the need for the TP systems to be scalable and reliable.

³ REST (Representation State Transfer) permits the HTTP methods GET, PUT, POST, and DELETE to be used to read and update a resource from an external HTTP client application.

Business events and business rules

Each scenario shown in Figure 4 on page 19 contains a real-time element, a requirement to know where and when an event occurs, a requirement to be able to obtain information about the customer concerned by the event, and a requirement to be able to initiate an action automatically. Here, we start to see the importance of another major technology trend, business event processing.

Business event processing enables the detection and correlation of events, for example, a potentially fraudulent transaction, so that an organization can respond to business situations in a timely manner. It is important to keep in mind that the capture of business events often occurs within the TP system, and the actions taken can also be taken by the TP system. We see therefore that the implementation of this technology is likely to increase the demand on existing TP systems.

“Customers now demand a more real-time dynamic view of their portfolio gains and losses. In some scenarios, event-based solutions will help us to provide this in a more efficient and decoupled way.” (Marcel Däppen, UBS WM&SB)

A business rule management system (BRMS) enables organizational policies – and the operational decisions associated with those policies – to be defined, deployed, monitored, and maintained separately from the core TP application code. By externalizing business rules and providing tools to manage them, a BRMS allows business experts to define and maintain the decisions that guide systems behavior, reducing the amount of time and effort required to update production systems, and increasing the ability of the organization to respond to changes in the business environment. Over time, our clients told us that they expect to increase integration based on event notifications. And, our clients expect that a Decision Manager can make decisions that are based on these notifications.

In addition to the technical changes to TP systems implied by business events and business rules, there is also an organizational impact. Rules and events were previously within the domain of the applications programmer; the approach today is to have these rules and events defined by a business analyst. The key success factors are seamless integration between the existing TP systems and new BRMS system, and well-proven organizational processes that promote cooperation between the application developer and business analyst.

Business process management

The adoption of SOA by itself does not optimize the business processes that ultimately determine the ability of the company to achieve efficiencies and respond to new market opportunities. In the IBM white paper, *Achieving business agility with BPM and SOA together*⁴, the authors argue that the convergence of SOA and business process management offers the best business agility. This view is supported by a significant number of our interviewed clients who stated that they require an industrial-strength BPM solution.

The key distinction for BPM as a discipline is the added focus on flexible and dynamic process design, and also process orchestration and automation through IT enablement. When Waldo opens a new bank account, for example, an automated business process is likely to be kicked off involving a sequence of steps similar to the following steps:

1. Waldo submits an application for a new account through the bank’s portal server.
2. The request is written to the Account Management System.

⁴ <http://ftp.software.ibm.com/common/ssi/sa/wh/n/WSW14078usen/WSW14078USEN.PDF>

3. The Process Manager initiates a credit check service, which is available from a business partner.
4. A sweep of all existing data is done to gather further information about Waldo, for example, what existing accounts does Waldo have.
5. The Decision Manager is called to make a decision about whether the account can be opened.
6. If an intervention by the account administrator is required, the necessary information is presented to the administrator to make the decision.
7. The account is opened in the core banking system and an entry is also made in the customer relationship management (CRM) system.
8. Waldo is sent the details of his new account.

Efficient business processes, such as account opening, loan processing, and mortgage lending, help financial companies to distinguish themselves from their competitors, in addition to realizing savings in their operating costs. This is why a number of our interviewed clients have told us that BPM is strategic for them, for both process choreography and human workflow. An initial step is often the service enablement of their TP applications because the modeling and development of business processes are simplified if core transactions have a clearly defined interface and are callable using standard protocols.

The immediate impact of new BPM solutions on existing TP systems is that transactions are now seen as individual activities within the scope of a process, which can have an implication on the overall integrity of the business process. Process integrity is required in order to protect against incomplete transactions and information inconsistencies. *Process integrity* extends the concept of transaction integrity to the complete business process, providing synchronization among human tasks, services, and information (see “Process integrity” on page 32).

Another more general impact of BPM and SOA is that they lead to business processes that span various architectures and platforms. If we take the example of Waldo opening a new account, we might find that the bank’s portal server runs on an Intel platform, the account management and core banking systems run on the mainframe, the CRM system runs on a UNIX platform, and so on. Key success criteria for BPM solutions, such as this one, are the ability to achieve end-to-end process visibility and the ability to execute and manage large numbers of concurrent processes both consistently and efficiently.

Business analytics

Many companies have built data warehouses and have embraced business intelligence and analytics solutions. Even as companies have accumulated huge amounts of data, however, it remains difficult to provide trusted information at the correct time and in the correct place. The amount of data to mine, cleanse, and integrate throughout the enterprise continues to grow even as the complexity and urgency of receiving meaningful information continues to increase.

Before information can become available in a dashboard or a report, many preceding steps must take place: the collection of raw data; integration of data from multiple data stores, business units, or geographies; transformation of data from one format to another; cubing data into data cubes; and finally, loading changes to data in the data warehouse.

Combining the complexity of the information requirements, the growing amounts of data, and demands for real-time analysis requires specialized analytics engines. It also requires the analytics process to run at the moment a transaction takes place, for example, a credit card transaction or room reservation, which means that TP systems are becoming more integrated with business analytics solutions.

Hybrid systems

Flexible deployment of the various application components that make up a heterogeneous application, including the TP components, means that new business processes can be brought to market faster. The most cost-effective solutions use systems optimized for the current task, whether the task is general transaction processing, or specific tasks, such as complex database queries or XML processing.

However, as business applications increasingly span more platforms, appliances, and devices, this wide range of resources creates management issues for IT shops trying to meet their business objectives. Simply adding servers, routers, and other IT equipment ultimately will not solve these IT challenges and might even make them worse. Even using virtualization techniques can only go so far in helping to manage a massive number of servers, routers, and other devices.

The ability to manage resources for these heterogeneous applications as one logical entity is becoming a strong requirement for IT service providers, as is the ability to monitor the business processes that span the hybrid environment.

Cloud

The promise of cloud computing is to organize and streamline the IT environment to realize IT without boundaries. This goal is made possible through physical consolidation, virtualization, and better IT management. According to one of our interviewed financial clients, it is important that new acquisitions use a cloud-based software as a service (SaaS) approach in order to share banking service factories.

In brief, the goals of a cloud-based solution are to address these areas:

- ▶ To spend less on system maintenance and more on pursuing business opportunities
- ▶ To provide more flexible options for hosting workloads
- ▶ To enable users to deliver new solutions with greater ease
- ▶ To provide better service and responsiveness

But implementing a cloud-computing solution that delivers these promises presents challenges that most organizations do not know how to overcome today, particularly in areas such as security, monitoring, and reliability. Gradually, however, these challenges are being addressed by standardization and middleware solutions, and practical implementations of cloud-based TP systems are emerging.

In summary, it is important to realize that there is no value to the business if the introduction of new technologies, such as the ones we described in this section, has a detrimental impact on the robustness of the underlying TP systems. New styles of applications, new web services, business processes, or new deployment models, should not be introduced if they are not reliable, cannot scale to the demands of use, or leave critical business information vulnerable to corruption or misuse. Maintaining business performance and integrity in the face of change requires a reliable, adaptable, and scalable environment.



Transaction integrity: A matter of trust

No matter what your business sells, where it operates, or who its customers are, its ongoing success might ultimately depend on the answers to three simple questions:

- ▶ Can your customers and value chain partners trust you to correctly and efficiently complete their business transactions?
- ▶ Can you trust your IT systems to ensure that those interactions are being properly completed, recorded, and applied?
- ▶ Do you understand the consequences of failing to ensure transaction integrity?

Transaction integrity

Transaction integrity (TI) is the assurance that the results you expect are being achieved. Customers and value-chain partners must be sure the transactions they conduct with your business are being completed and are correct.

To illustrate what we mean, we look at what happens when Waldo moves some money from his savings account to his son's checking account. It is never acceptable to Waldo for his money to disappear. If Waldo ever doubts the safety of his money, he would probably switch financial institutions.

Waldo's money is represented by data in two databases that cooperate to ensure that the data they contain is always in a known and consistent state. That is, these two databases allow actions or tasks between them to be within a single transaction, as shown in Figure 6 on page 26.

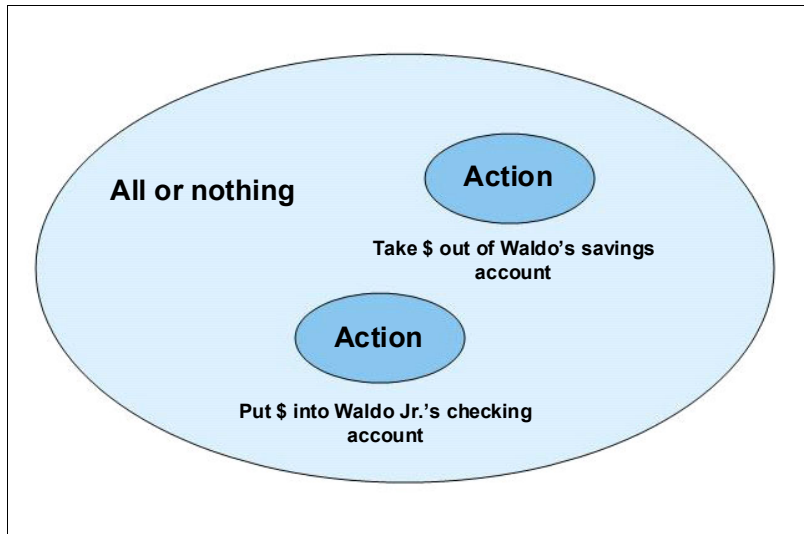


Figure 6 Waldo's transaction

We assume for the moment that the savings account records are kept in one table, and the checking account records are kept in another table; both are in the same database resource manager. In this case, two updates will occur. In the savings account database, the application will read the account balance, will reduce the balance by \$100 (assuming Waldo can cover the transfer amount), and will write the updated balance back to the savings account. Next, the application will read the balance of Waldo Jr.'s checking account, increase the balance by \$100, and write the balance back.

Both updates to the tables better work, or they better both fail. If the debit from Waldo's savings succeeds but the deposit to Waldo Jr.'s checking account fails, Waldo will be out \$100. If the debit from the savings fails and the deposit to the checking succeeds, the bank will be out \$100. The database must internally coordinate these two updates so that they operate *atomically*, as a single unit of work that either fully succeeds or completely fails.

Assuring consistency

Before you start thinking "that sounds easy," consider some of the failure cases that the database must handle. Assume for a moment that the debit works: the savings account is reduced by \$100. Next, assume that the credit to the checking account fails. Before the database can return, it must reverse the debit from the savings account.

But what if the computer or the disk fails before the database can update the savings balance back to its original value? Likewise, what if the database system fails before it can return to your application to let it know that the transaction succeeded or failed? How does your application know what actually happened and whether the transaction completed or not?

To cope with these "what if" scenarios, the database needs to support the ACID¹ properties and have sophisticated mechanisms to ensure they remain internally consistent. In the simplest terms, databases do not really write directly over their original values. They put all the updates on an empty location on the disk. After all of the updates they make in a transaction are written, the last operation they perform is to switch a pointer to refer to that newly written data as the updated record. Then, they can reclaim the old disk space (where the original data was previously written) as an open spot for updates in the next transaction.

¹ The ACID properties are atomicity, consistency, isolation, and durability.

In addition, the transaction manager must keep a log of everything it does. This log is referred to as a *transaction log*. The transaction log is a running record of all the actions performed within a transaction. The transaction manager notes that a transaction was started and which resources will be used in the transaction. It also notes any updates that were made in the transaction, and when the transaction is completed. The transaction log has enough information to essentially reprocess or reverse the transaction at any time.

External commit coordination

We modify Waldo's situation slightly. Assume that his savings account and Waldo Jr.'s checking account information are not simply in separate tables of the same database, but in fact are on completely separate databases. Also, assume that the savings account is kept in an Oracle database, and the checking account is kept in IBM DB2®. Each of these databases has its own built-in transaction manager and is able to coordinate transactional updates within its own database manager.

However, this configuration creates exactly the same problem we had before. The debit from the Oracle database and the deposit to the DB2 database must both occur, or neither of them must occur. The transactional updates that occur in each of these databases need to be coordinated. We need *external commit coordination*, which is something that can coordinate the transaction commit processing across Oracle, DB2, and any other transactional resource manager that might be used by the application during the transaction.

We refer to a transaction that spans multiple different transactional resource managers as a *global transaction*. Compare this definition to what we refer to as a *local transaction*, which is a transaction that is performed and committed entirely within a single resource manager that is controlled by its local transaction manager.

One of the many key items that we get from middleware, such as IBM WebSphere Application Server, is an external transaction manager. When you run your application in WebSphere Application Server and use other resource managers (such as DB2, Oracle, Sybase, WebSphere MQ, TIBCO Enterprise Message Service, and countless other transactional resource managers), WebSphere enlists these resource managers in a global transaction automatically and without any special code in your business application.

Many people will say, "I do not need global transaction support, it is just overhead and in the way." This is like saying "I do not need gas for my car, it just adds weight and it is inconvenient to have to stop at the gas station."

In reality, there are few mission-critical enterprise business applications where you are not trying to drive change across multiple databases, message queues, partners, and so on. In any of these cases, you have essentially three choices:

- ▶ Do not worry, and risk the possibility that changes in one part of your data might be inconsistent with other parts of your data.
- ▶ Write lots of error-recovery logic in your application: adding more to what has to be designed, implemented, and tested, hoping that you caught all of the cases, and taking more time away from your core business logic.
- ▶ Rely on your middleware to coordinate transactional updates across whatever resources you use in your application and to assure you of a binary outcome: either it was all updated consistently or it was not.

The power of transaction processing middleware is that it handles the error processing for you. Embedded within the middleware are decades of experience of managing mission-critical applications and the kinds of problems that can occur in applications that use multiple resources. Yes, that adds some weight to the application server to capture the logic that handles all the various, and sometimes obscure and subtle, conditions that you might encounter. You might never encounter all of the conditions that are possible. Or, more likely, they might occur but you might never know about it except that a client of one of your customers is missing some money, either causing customer dissatisfaction, or perhaps worse, never really knowing it themselves until far too late.

Chances are your application developers will never really know all of the potential failure cases that can occur across the resources they use. Certain really experienced programmers might know some of the cases, or even fewer might know many of them. However, you will rarely be able to afford programmers of that caliber. Having all that experience captured in the middleware is much less costly, which is why our interviewed clients believe that transaction integrity should be handled by middleware.

“If we could not rely on our middleware to protect against unpredictable events, we would need to consider how to manage recovery in our applications when accounting problems occur.” (China Merchants Bank)

Distributed commit processing

When a transaction updates two or more databases on different systems, the challenge of ensuring atomicity is increased because of the risk of network failures that can occur during the synchronization processing. We again look at Waldo’s account opening scenario. His checking account is created on the core banking system and an update is also made in the customer relationship management (CRM) system. As with most business transactions, you do not want one transaction to occur without the other transaction happening. You do not want the checking account to be created without making the update to the CRM system. In essence, we are committing two changes to two resource managers, running on two different nodes, in the same global transaction.

Two-phase commit protocol

We can maintain integrity across heterogeneous resource managers because the protocol for external-commit processing is standardized. The most commonly used standard was defined by the Open Distributed Transaction Processing technical specification for the *two-phase commit protocol* (as shown in Figure 7 on page 29). This specification is also known as the extended architecture (XA)² specification, and resource managers that support the two-phase commit (2PC) protocol defined by this specification are sometimes called XA resources.

² The XA standard is a specification by The Open Group for distributed transaction processing.

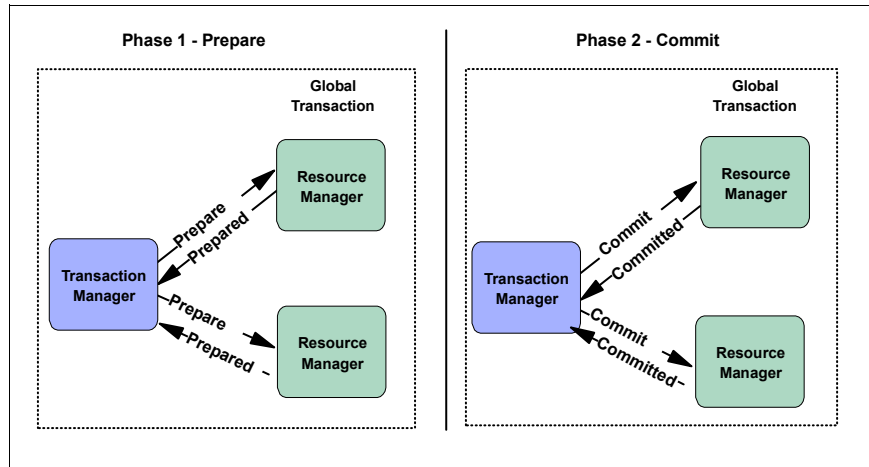


Figure 7 Two-phase commit protocol

The two-phase commit protocol works essentially in the following manner:

1. When you use an XA resource, its resource manager registers that use with a transaction manager. If this use will be part of a global transaction, the resource manager registers it with an external transaction manager.
2. The same step is repeated for each resource that you use within that transaction context.
3. When you are ready to complete the transaction and issue the commit command, the transaction manager then enters into the two-phase commit protocol with the resource managers:
 - In phase 1 of commit processing, the transaction manager asks all the resource managers to prepare to commit recoverable resources (prepare). Each resource manager can vote either positively (prepared) or negatively (rolled-back). If a resource manager is to reply positively, it records to a recoverable log the information that it needs and replies “prepared,” and is then obliged to follow the eventual outcome of the transaction as determined at the next phase. The resource manager is now described as *in doubt*, because it has delegated eventual transaction control to the transaction manager.
 - In phase 2, if all the resource managers voted positively, the transaction manager replies to each resource manager with a commit flow. Upon receipt of the commit flow, the resource manager finalizes updates to recoverable resources and releases any locks held on the resources. The resource manager then responds with a final committed flow, which indicates to the transaction manager that it is no longer in doubt.

For the most part, the IT industry has conquered the challenges of 2PC processing, in that, most middleware products include this support and also open standards exist that allow different middleware products to communicate using 2PC protocols. For example, the *Java EE Connector Architecture (JCA)* is part of the *Java Platform, Enterprise Edition (Java EE)* standard implemented by application servers, such as WebSphere Application Server. The JCA provides support for 2PC as part of its transaction management contract. It also provides support for connection and security management between an application server and an enterprise information system (alternative terminology for a TP system). Similarly, the web services standard *WS-Atomic Transaction (WS-AT)*³ implements the 2PC protocol so that heterogeneous resource managers can coordinate their updates using web services.

³ WS-AT is a web services specification that defines the atomic transaction coordination type for transactions of a short duration.

One area that remains challenging for some vendors is how to handle synchronization or resynchronization failures. Some TP middleware products will note the error but leave the records and resources locked and the database unrecovered until an administrator manually recovers it. When you are talking about high-volume processing and millions or billions of records, the potential for seriously impeded performance, or data integrity issues, becomes all too real.

However, even if the middleware efficiently manages distributed 2PC transactions, for many business transaction scenarios, especially those that span several systems, maintaining long-lived locks, and restricting data access for extended periods of time, is not an acceptable option. This reason is why most surveyed clients are careful to limit the scope of distributed transactions. They realize that distributed units of work (UOW) are possible across multiple platforms, but that managing a distributed UOW spanning multiple servers can cause operational difficulties when failures occur.

Asynchronous transactions

In a *synchronous* model, the expectation is that the service requester and the service provider are both available at the same time. An *asynchronous* model removes this requirement by adding an intermediary queue of work. The service requester puts a request message onto the queue. The service provider gets the request message from the queue. This pattern introduces a different transactional model, with two transactions typically used to ensure that the request message is delivered to the service provider.

"We try to keep the unit of work always as short as possible and we trigger non-critical business tasks in an asynchronous way." (Marcel Däppen, UBS WM&SB)

Imagine the case where the bank wants to print and send a letter of acknowledgement to Waldo for a really high-value deposit. The logic in the application looks at the deposit amount, and if it is over a certain amount, \$10,000 for example, it prints out a letter confirming that the deposit was received and credited to Waldo's account. The bank wants this task to occur transactionally, that is, to ensure that the letter is printed and sent if the deposit is made, but not to send the letter if the deposit fails for some reason.

However, printing a letter might take 5, 10, or even 30 seconds or more to print - depending on the speed of the printer. And if you had to print the letter within the transaction scope, you would in effect be holding locks on the account database for that entire time, which is an eternity in computing terms where most transaction times are measured in milliseconds, or even microseconds.

The bank can handle a situation like this through the use of a reliable messaging system, that is, the message can be put on a queue as part of a global transaction. Likewise, the message can be retrieved from the queue as part of a global transaction. In the meantime, because the queue is backed by persistent storage, it is highly reliable: the message will remain on the queue indefinitely. This way means that the printing process can be done asynchronously but with a high level of integrity.

The key aspect about asynchronous transactions is that they are really performed under *two* distinct and essentially independent transactions: one transaction to put the message on the queue, and a separate transaction to read and remove the message from the queue. The only thing holding the asynchronous transactions together is the message on the queue. The thing that ensures the integrity of the combined results is the assurance of the queue manager to hold messages on the queue indefinitely and across any potential failure or restarts of the system.

In that sense, we can claim the combined logical transaction is *atomic*, that is, either all of it happens or none of it happens. More literally, the first transaction either happens or does not happen. If the first transaction rolls back, the message is never committed to the queue and therefore the second transaction (associated with that message) never begins or is never completed. If the first transaction completes, a message is posted to the queue and the second transaction is started, and will either be completed successfully or will be retried until it does complete.

However, the combined logical transaction is only partially *isolated*, that is, while no one else is able to see intermediate results in the first transaction until it completes, you can certainly see the results of that first transaction even before the second transaction has even started, and before it has been completed. In this sense, we describe these two transactions as being loosely coupled, a part of a loosely coupled asynchronous application. The main benefit of asynchronous transactions is that they are very scalable.

Compensating transactions

The emergence of SOA-based and business process management (BPM)-based solutions means that business processes now span an increasing number of systems, and maintaining robust integration and consistency becomes ever more challenging. This challenge has placed a strain on the classical transaction model. As a result, we see the emergence of the concept of *eventual consistency*, which is based on *compensating transactions*.

Eventual consistency

Eventual consistency provides a guarantee that a transaction will be correctly recorded and that the data will eventually be made consistent even if there is a significant period of time during which the data is not consistent. It is based on the loose coupling of a set of transactions that make up a larger business transaction.

The idea is that, within certain parameters, an organization can guarantee customers that their transactions will be correctly recorded and that eventually the data will be consistent even if that consistency is not immediately evident. In practical terms, there will be periods between reconciliations when the system is not consistent and does not give expected results.

A *compensating transaction* is a group of operations that undoes the effects of a previously committed transaction. It is important to stress that an application that depends on compensating transactions must have extra logic to deal with failures, and the possibility that further updates are made to the resource in between the original committed transaction and the *undo* transaction. Otherwise, the data might be left in an inconsistent state.

Process integrity

As we have discussed previously in this paper, business agility is critical for excellence in managing change; however, business performance and business integrity are critical for excellence in business execution.

"Business agility without process integrity is, at best, a fragile value proposition." (Claus Jensen, IBM Chief Architect SOA-BPM-EA Technical Strategy)

Process integrity requires an automated and predictable compensation and recovery mechanism to manage process failures, whether those failures are in the process itself or in the information systems on which the process depends. In the IBM white paper, *BPM and SOA require robust and scalable information systems*⁴, the authors illustrate these dependencies as shown in Figure 8.

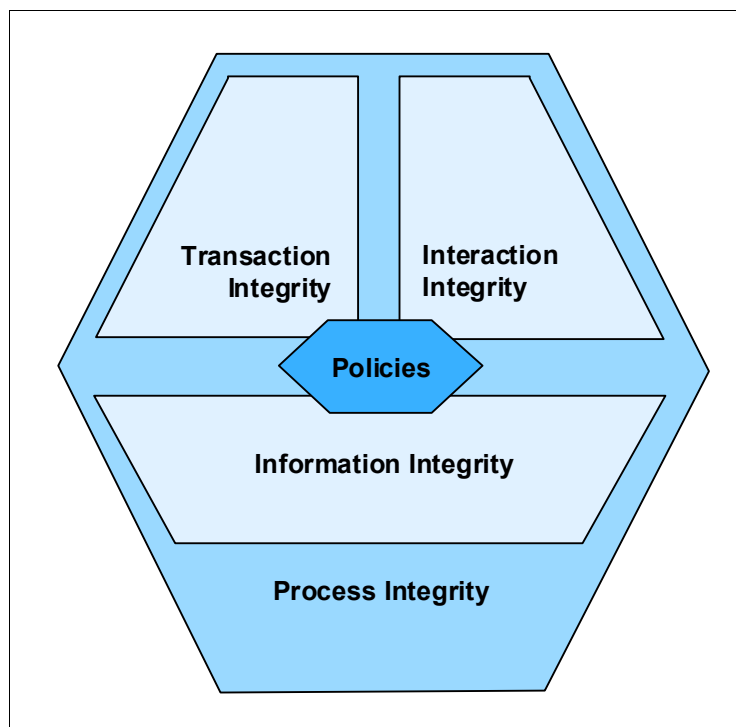


Figure 8 Process integrity

Process integrity depends on transaction integrity but it is also dependent on interaction and information integrity:

- *Interaction integrity* is the ability to ensure that elements of people's interactions with information systems are intact wherever and whenever those interactions occur, thereby making people an integral part of the process execution environment. This is somewhat similar to the notion of transaction integrity, only focused on the many distinct and independent user interactions within an activity rather than on the activity as a whole.

⁴ [ftp://ftp.software.ibm.com/common/ssi/sa/wh/n/wsw14104usen/WSW14104USEN.PDF](http://ftp.software.ibm.com/common/ssi/sa/wh/n/wsw14104usen/WSW14104USEN.PDF)

- *Information integrity* provides consistency of information consumed or produced by business processes. We need reliable, complete, and manageable information, regardless of the delivery channel. What is important to realize is that most horizontal business processes touch upon information with different semantic contexts, formats, and lineage; therefore, the information infrastructure must be able to deliver trusted information in a consistent fashion independently of source platform.

The process itself is generally defined as describing a sequence of activities along with the key metrics, policy conditions, and business assumptions that profile the process. One of the operational policies defined governs automated compensation and recovery.

Consider Waldo's account-opening scenario as an example (see "Business process management" on page 21). Perhaps the account administrator decides not to approve the request in step 6 on page 22. In this case, an operational policy can be defined so that the previously committed update to the Account Management System is automatically backed out with a compensating transaction. This is in contrast to step 7 on page 22 in which the core banking and CRM systems are updated within a global transaction.

When you build business processes that are automated in the WebSphere Process Server, along with the other middleware products offered by IBM that build on WebSphere Application Server, you can enable process integrity through the combined transactional strengths of these products.



How System z differs in transaction processing and integrity

In this chapter, we explore how the IBM mainframe is critical in contributing to the typical business transactions in the day-to-day life of Waldo and Waldo Jr. Transactional Integrity on IBM System z® was not only added to the middleware transactional managers, but “built in” as part of the base “DNA” in the system. To understand how middleware benefits from these base capabilities, we must first briefly look at its history.

The first shared mainframe

In 1964, Waldo’s father might not have even noticed, but IBM announced a new family of computer systems, the IBM System/360. It represented the first basic reorganization of the electronic computer by IBM since the development of the IBM 701 in 1952. More than any other computer development, it tied together the “loose ends” of electronic data processing. Specifically, the new system enabled companies to integrate all of their data processing applications into a single management information system. Virtually unlimited storage and instant retrieval capabilities provided management with up-to-the-minute decision-making information.

In 1970, IBM introduced System/370, the first of its architectures to use virtual storage and address spaces, and with it came a new operating system, the IBM Multiple Virtual Storage (MVS™) system. With multiple virtual address spaces, errors are confined to one address space, except for errors in commonly addressable storage, thus improving system reliability and making error recovery easier. Programs in separate address spaces are protected from each other. Isolating data in its own address space also protects the data. The use of address spaces allowed MVS to maintain the distinction between the programs and data belonging to each address space. The private areas in the address space of one user are also isolated from the private areas in other address spaces; this address space isolation provides much of the operating system’s security. With MVS came other subsystems and features that lay the foundation for future transaction processing, for example, storage keys, Resource Recovery Services (RRS), System Management Facility (SMF), Security Authorization Facility (SAF), and Workload Manager (WLM).

The mainframe environment supports a truly shared environment. Resources, such as CPUs, memory, storage, programs, and data, can be shared by multiple users (transactions); however, the integrity of each of those transactions is enabled by the underlying system services.

The implementation of security keys, SAF, and WLM at the operating system (OS) level creates a foundation designed for running mixed workloads in a secure and isolated environment. This way, one transaction cannot affect another transaction, and you can define different performance and execution priorities for each workload or transaction. In addition, SMF and RRS provide services for you to gather execution metrics, to recover in-flight or failed transactions, and to free up resources locked by failed transactions. With the addition of IBM Parallel Sysplex® technology, you can harness the power of up to 32 IBM z/OS® systems, yet make these systems behave like a single, logical computing facility.

The mainframe has evolved over the past 40 or more years. The system has changed in response to client requirements and technology advancements. As an example, instructions were added to the CPU to support Java workloads, and other middleware and applications. The internal memory, chip packaging, and I/O have changed to create a true workload-optimized system from the ground up.

Figure 9 shows how the CPU has changed over the past generations to grow holistically to the fastest CPU available. The four levels of cache or internal memory are designed to support mixed workloads and high-volume online transaction processing (OLTP), while ensuring transactional integrity.

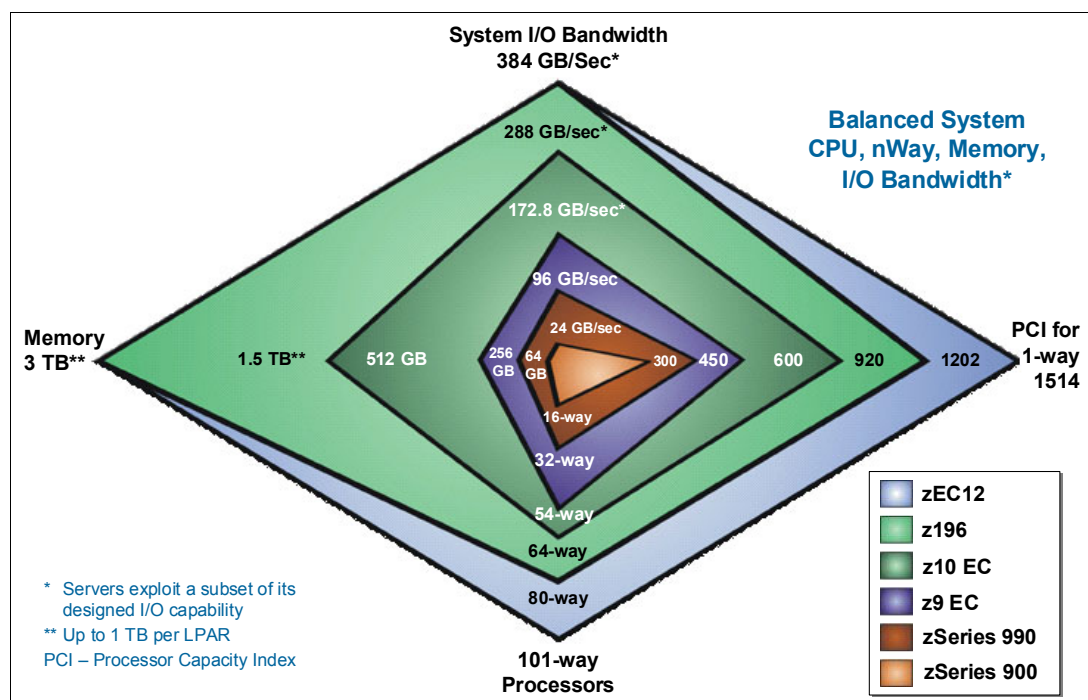


Figure 9 System z CPU evolution over the past generations

“Our new z196 is significantly faster.” (Andreas Wagner, head of DBA department, GAD)

The shared mainframe today

As explained in the previous section, the System z architecture was designed with the concept of *sharing*. Sharing starts in the hardware components and ends with the data that is being used by the platform. The ability to share everything is based on one of the major strengths of the System z mainframe: *virtualization*.

As the term is commonly used in computing systems, virtualization refers to the technique of hiding the physical characteristics of the computing resources from users of those resources. Virtualizing the System z environment involves creating virtual systems (logical partitions and virtual machines) and assigning virtual resources (such as processors, memory, and I/O channels) to them. Resources can be dynamically added or removed from these logical partitions through operator commands.

System z offers a management layer so you can divide your system into logical partitions (LPARs), each running the operating system of your choice and each with the ability to consume resources within a set of parameters that you define. A key design point is to exploit resources where possible to meet service-level agreements (SLAs). This statement means that a busy LPAR can “steal” resources from another LPAR if that other LPAR has idle resources available.

Figure 10 shows an example of a System z server sharing a variety of processor types across LPARs that are running various operating systems and workloads.

Note the following information about the figure:

- ▶ LPARs 1 - 4 all run dedicated z/OS operating systems for production and preproduction purposes.
- ▶ LPAR 5 runs z/TPF.
- ▶ LPARs 6 and 7 run a dedicated Linux system.
- ▶ LPAR 8 runs IBM z/VM®, with a number of Linux guests.

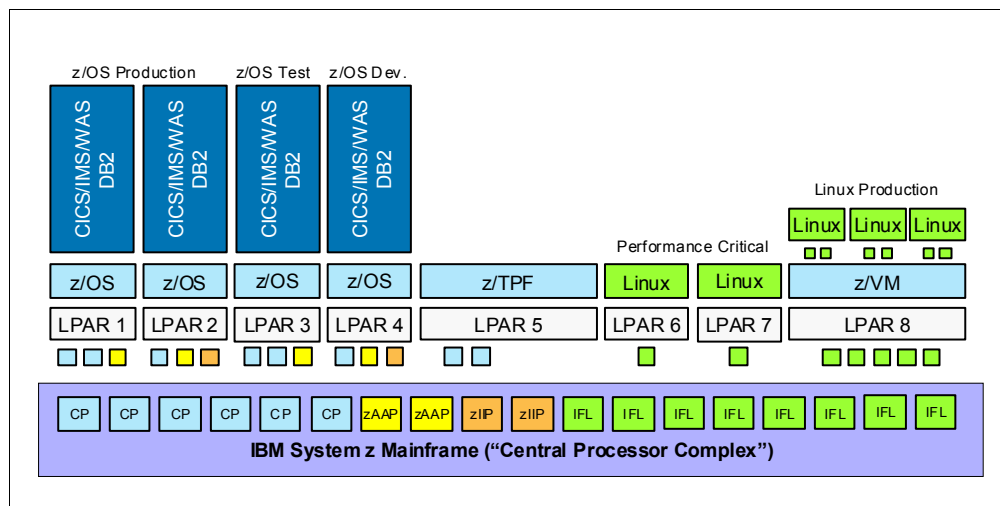


Figure 10 The IBM mainframe of today: Virtualized and sharing everything

The applications running in CICS, IMS, or WebSphere Application Server inherit all the qualities of service and transactional integrity of the middleware, operating system, and System z hardware described previously.

“The specific System z TP capabilities that distinguish it from other platforms are scalability, workload management, system monitoring, and data integrity.” (China Merchants Bank)

zEnterprise: A truly workload-optimized system

“The shared mainframe today” on page 37 demonstrates that the IBM mainframe is fully virtualized and shares resources across multiple system images running a variety of operating systems. The proportions of resources allocated to each of these images are managed in a dynamic manner using sophisticated software built into the IBM mainframe. The most recent generation of IBM mainframes, the IBM zEnterprise System, goes even further. It not only provides a choice of operating systems, but also a choice of processor types. The zEnterprise also provides management software to help you manage a heterogeneous application that consists of multiple components running on various operating systems and processor types as one logical workload.

Figure 11 on page 39 shows how zEnterprise adds a high-capacity and highly scalable blade architecture to the already existing “shared everything” IBM mainframe. An application, for example, SAP, might span both blades on an IBM zEnterprise BladeCenter® Extension (zBX) and LPARs on the z196, z114, or zEC12 central processor complex (CPC). Such an application is often referred to as a *heterogeneous* or *hybrid* application or workload. In this case, the IBM zEnterprise Unified Resource Manager is used to manage the application as one logical entity with one set of service levels. Also, the communication between the components residing on the z196, z114, or zEC12 CPC and the components residing on zBX occurs over a 10 GB high-speed internal private network.

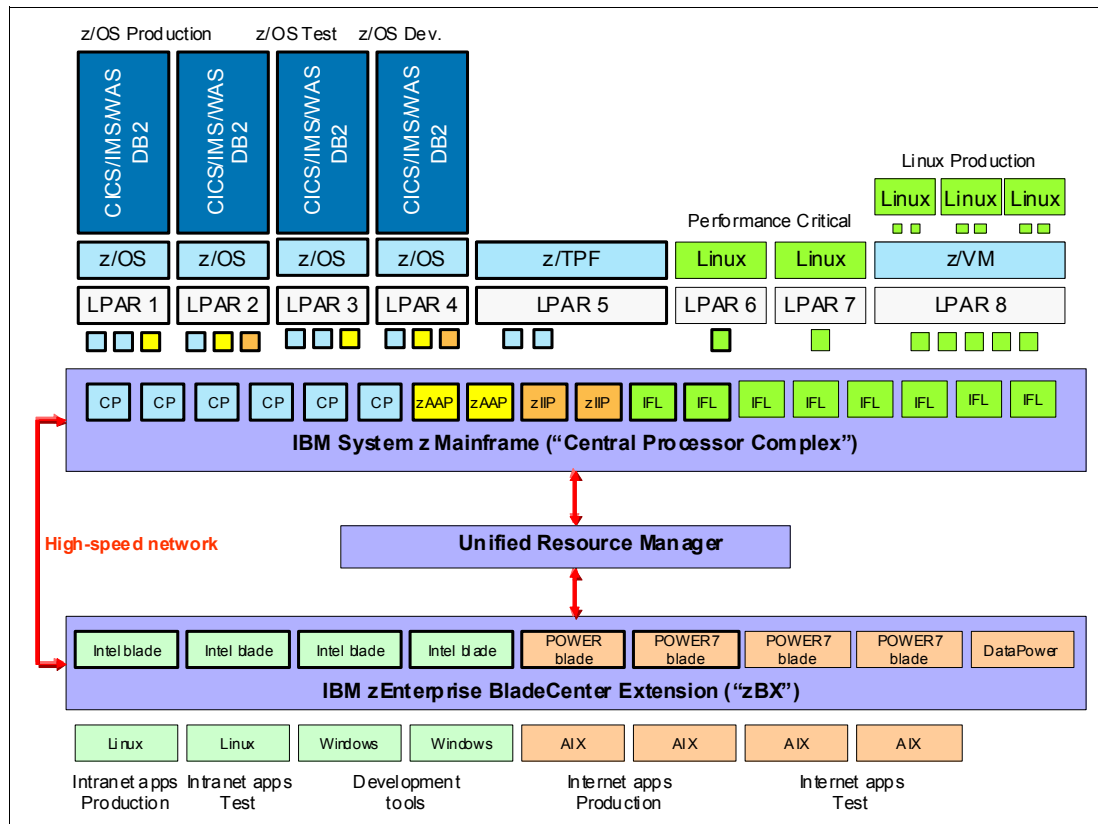


Figure 11 zEnterprise: Unifying the shared everything IBM mainframe with a blade architecture

zEnterprise scalability capabilities

zEnterprise enables both *vertical* and *horizontal* scaling:

- As a result of the “shared everything” architecture on the System z CPC, applications can scale up significantly within their LPARs and across LPARs, when implemented in a Parallel Sysplex configuration. This scale-up behavior does not require any manual intervention to add resources and happens dynamically. Applications that experience significant peaks can benefit from this type the most, and service levels can be easily maintained.
- Applications with an architecture that permits running many occurrences on physically separate servers can benefit from the “scale-out” behavior of the zBX. A good example includes Internet-facing front-end applications, using many resources for rendering user interfaces, but typically not running any transactions or performing mission-critical database updates.

Often, an application has both components that require scale-out behavior and scale-up behavior. In this case, the application components that require the scale-up behavior, for example, database-update transactions, can be implemented on the System z CPC; the application components that require scale-out behavior, for example, an HTTP Server that renders web pages, can be implemented on the zBX. As stated previously, these components are managed as one application by the Unified Resource Manager.

Figure 12 on page 40 shows a simplified view of scalability benefits.

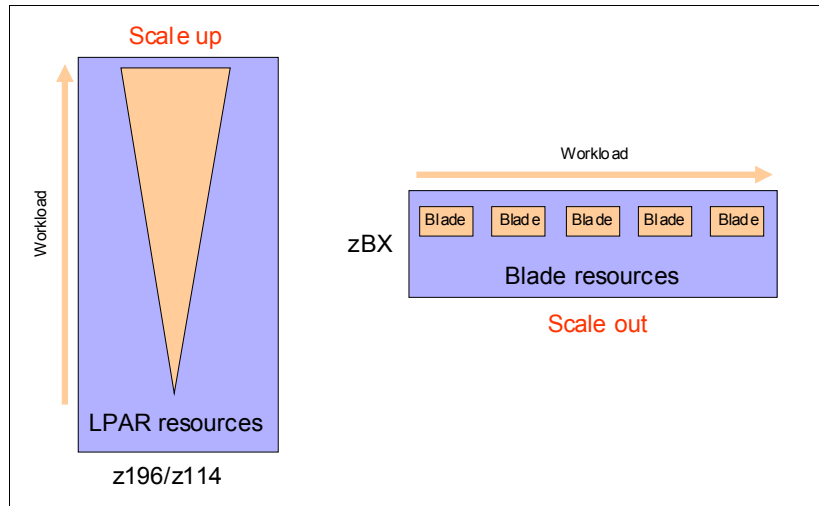


Figure 12 zEnterprise scalability

We have already discussed some important business and technology trends affecting transaction processing. One of the trends is that transactions are in fact becoming more complex and might span multiple application tiers and middleware products. With the zEnterprise, it is possible to keep heterogeneous transactions, even when running across multiple operating systems and on multiple processor types, in a single management scope and on one logical server platform.

Accelerators and optimizers

An increasing trend in IT infrastructure is to use appliances. An *appliance* is a hardware device with preloaded firmware built to perform specific tasks in a highly efficient manner. An appliance requires a minimal number of steps for installation and configuration and is therefore almost ready to run when it is plugged in. Because an appliance is built for limited specific tasks, another big advantage is that it is highly optimized and much faster in those tasks than a regular server that uses a generic operating system.

There are two appliance types: *accelerators* and *optimizers*:

- ▶ An accelerator performs the tasks the server could have performed, but much faster. In this case, the server basically “delegates” tasks to the accelerator with the main objective of performing these tasks much faster and freeing up processor cycles on the server. The IBM DB2 Analytics Accelerator for z/OS is an accelerator that can be connected to a zEnterprise system running DB2 for z/OS to speed up analytics and business intelligence queries, by an order of magnitude.
- ▶ An optimizer is similar to an accelerator, but usually adds functionality that is not available on the server. Therefore, besides performance reasons, an optimizer is also chosen for its specific functionality. The IBM WebSphere DataPower® Integration Appliance XI50 for zEnterprise (XI50z) is an optimizer that performs enterprise service bus (ESB) functionality and can be used as part of a service-oriented architecture (SOA) deployed on zEnterprise. The XI50z is managed as an integral part of zEnterprise with the same management software (Unified Resource Manager) that is used to manage server images.

Appliances are part of the IBM strategy in general and are an important component in hybrid and cloud infrastructures.

Transaction management

A key concept of IBM mainframes is that ensuring reliability with transaction processing is a responsibility of the middleware and operating system. Customer Information Control System (CICS), Information Management System (IMS), and z/Transaction Processing Facility Enterprise Edition (z/TPF) are all heavily used transaction management solutions in many of the Fortune 1000 companies around the world. These transaction managers are fully integrated with database managers, such as DB2 for z/OS and file systems and Virtual Storage Access Method (VSAM). In fact, DB2 for z/OS is also a transaction manager and you can run stored procedures under a two-phase commit scope, which adhere to the atomicity, consistency, isolation, and durability (ACID) properties described earlier.

The z/Transaction Processing Facility (z/TPF) operating system is a special-purpose system that is used by companies with very high transaction volumes. The CICS and IMS transaction managers require an underlying operating system, which is z/OS by default for IMS and either z/OS or IBM z/VSE® for CICS. Under the Linux for System z operating system, transaction management is available through WebSphere and a number of ISV solutions. Most of the WebSphere family of products on System z also provide transaction management. Java EE applications can be run transactionally inside the WebSphere Application Server.

Transaction managers on System z

This section describes the IBM transaction manager solutions available on System z. Generally, running a workload inside a transaction manager provides the best reliability, frees the developer from the burden of maintaining the transaction management logic, and provides a common framework and APIs.

CICS Transaction Server

CICS is one of the most popular IBM software products. The base of the CICS technology, however not yet under its current name, appeared for the first time in 1968 as the “Type II Application Program” that was designed to help companies with building online TP applications. The first version of CICS marked the beginning of the shift from batch processing to OLTP.

Through the years, CICS transformed into an “all-in-one” TP system, with functionalities, such as queuing, rollback, logging, authorization, and system services, such as storage management, tracing, statistics, messaging, operational services, and workload management, to name a few. CICS provides a programming model and run time that in combination provide high-volume, highly secure, and scalable processing of transactions with automatic two-phase commit, and recovery of “broken” transactions. CICS systems are normally run as part of a CICSplex, a group of CICS systems that you manage and manipulate as if they were a single entity. Modern tools are available for application development and testing, and also administration and configuration of the CICSplex.

CICS supports no less than nine programming languages, varying from traditional 3GL languages, such as Assembler, COBOL, and PL/I, to other languages that are popular today, such as Java, C/C++, PHP, and Groovy. With additional functionalities, such as Web 2.0, ATOM feeds, business event processing, and integration with other middleware products, CICS is an integral part of the mainframe application architecture of the future. If CICS did not manage transaction integrity, the loss of the predictability of behavior would create significant work. Our clients’ applications would look extremely different and would be much more complex.

“CICS is very important to us in guaranteeing transactional integrity.” (China Merchants Bank)

Figure 13 shows a high-level overview of CICS and the supporting ecosystem.

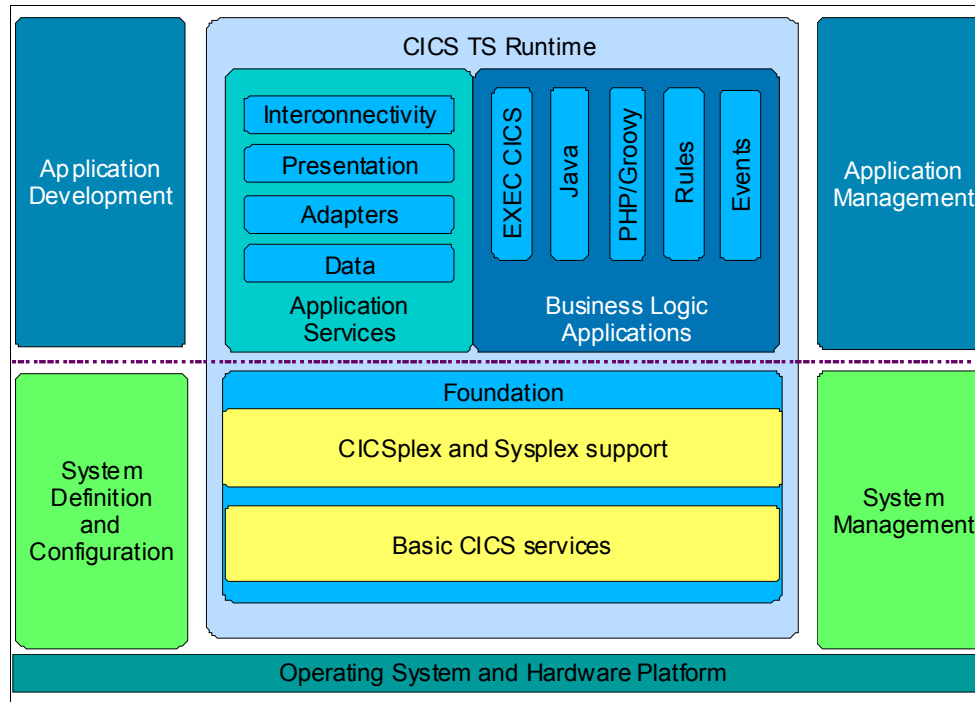


Figure 13 CICS overview

With the recent announcement of the CICS TS V5.1 open beta, cloud-style development, deployment, and operations are being added to the CICS platform.

“CICS has been the gold standard for transaction processing for more than 40 years, growing and evolving with the mainframe platform and adding unique capabilities in response to customer demand. Game changing levels of efficiency and agility are our aims for the CICS TS V5.1 open beta.” (Ian Mitchell, IBM Distinguished Engineer, CICS Transaction Server)

IMS

IMS is the IBM premier solution combining transaction and hierarchical database management, virtually unsurpassed in its kind in database and transaction processing availability and speed. IMS clients have trusted IMS with their most critical business asset, their operational data, for decades. Today’s IMS has only a superficial resemblance to the product that first shipped in 1969. However, an application program that ran on IMS in 1969 will still run today, unchanged, on the current release of IMS.

IMS consists of two major components: the Transaction Manager (TM) based on a message queuing paradigm and the Database Manager (DB). In addition, a set of system services provides common services to the TM and DB components:

- ▶ The IMS Transaction Manager provides high-volume, rapid response transaction management processing for application programs accessing both IMS and DB2 databases, as well as MQ queues. IMS TM supports numerous external connectivity mechanisms - both inbound and outbound - to access external resources, such as terminals and applications. The requests for processing can come from people at terminals or workstations, or other application programs either on the same z/OS system, on other z/OS systems, or on distributed platforms.
- ▶ The IMS Database Manager provides a central point of control and access for the IMS databases based on a hierarchical database model. This centralized environment provides access to these databases from other TP systems (IMS, CICS, DB2 stored procedures, and WebSphere Application Server), from batch jobs, and from any distributed environment using a set of universal drivers built on the Distributed Relational Database Architecture (DRDA®) standard. IMS data sharing can be implemented in a Parallel Sysplex environment to provide high availability.

Figure 14 provides a high-level overview of IMS and its supporting tooling environments.

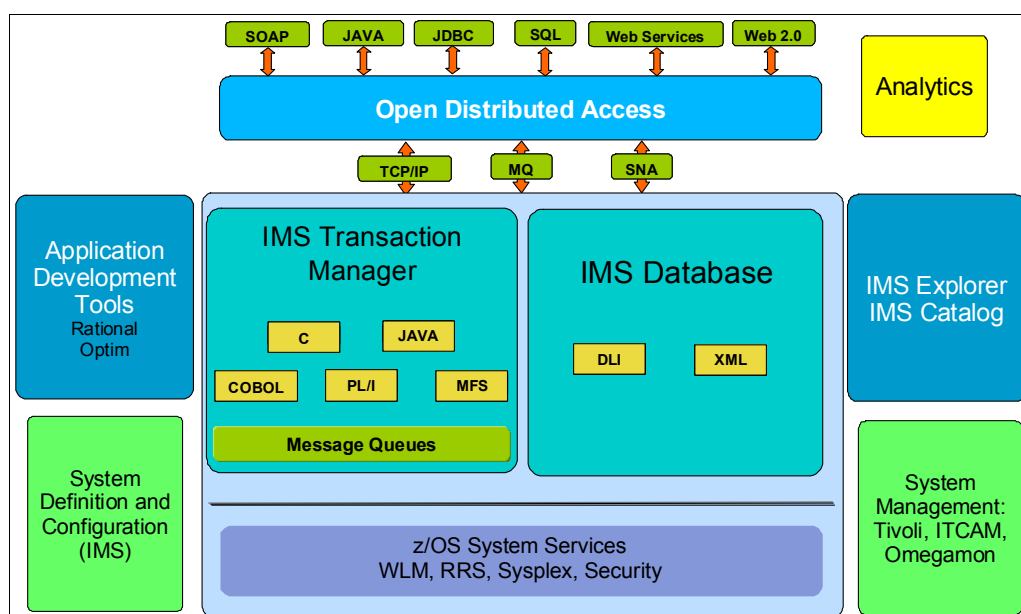


Figure 14 IMS overview

Cloud, big data, and whatever the future holds for the evolution of the IT industry will undoubtedly factor into how IMS TM will evolve. What does not change, however, is the critical importance of transaction integrity, and the IMS TM position as a transaction manager of choice in guaranteeing the integrity of its client transactions.

WebSphere Application Server

WebSphere Application Server is a modern transaction manager that provides all the capabilities of a traditional TP system but is based on the Java programming language and Java EE programming model. WebSphere Application Server has become the foundation for many other middleware solutions and packaged applications.

The first versions of WebSphere Application Server focused on “web serving” capabilities, such as providing web browser content and secure access, but soon “application serving” capabilities were being added. WebSphere Application Server, specifically the version for the z/OS operating system, soon started to provide sophisticated two-phase commit support for programs running in WebSphere Application Server, database updates in DB2, and programs running in CICS and IMS.

Compared to the traditional TP systems on System z, such as CICS and IMS, the behavior of transactions in WebSphere Application Server is determined by configuration attributes set by the application developer.

The combination of WebSphere Application Server and the z/OS operating system provides a unique solution for meeting service levels:

- ▶ Vertical scalability is achieved by the ability to run multiple application servers within a server image, and, depending on workload, each application server can scale up by dynamically adding resources.
- ▶ Horizontal scalability is achieved by the ability to cluster multiple application servers across multiple LPARs that reside on the same or different physical servers.

In both cases, all application servers can access the same image of the data when using DB2 Data Sharing and Parallel Sysplex.

As shown in Figure 15, WebSphere Application Server for z/OS distinguishes itself from WebSphere Application Server on other platforms in a number of ways.

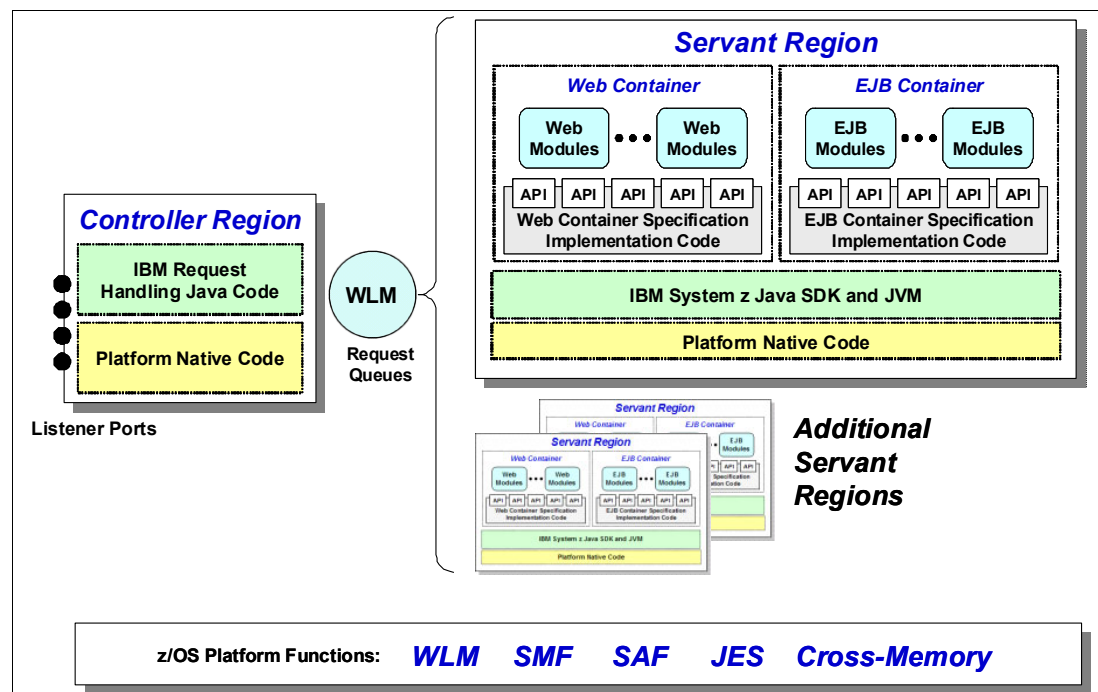


Figure 15 WebSphere Application Server overview

Note the following information about Figure 15 on page 44:

- ▶ There are two types of “regions”: the *controller* region and the *servant* region. Application code runs in the servant regions and is thus isolated from the authorized WebSphere Application Server code running in the controller region. This approach is definitely a benefit for reliability and availability.
- ▶ Multiple servant regions can exist and can be stopped and started dynamically based on workload. If multiple servant regions are running, WLM distributes the workload across the available servant regions, based on service-level policies.
- ▶ As mentioned earlier, if WebSphere Application Server servers are implemented and clustered in multiple system images (on the same or distinct physical servers), and these system images are part of a Parallel Sysplex, a highly resilient environment is achieved. In this resilient environment, any WebSphere Application Server server in any image can do the same work as the other WebSphere Application Server servers, accessing the same data, the same WebSphere MQ queues, and so on.
- ▶ WebSphere Application Server for z/OS exploits z/OS subsystems and System z hardware to perform specific tasks. Examples are in the following list:
 - Security functions are delegated to the security product that is installed on z/OS by invoking System Access Facility (SAF) interfaces.
 - For logging, the System Management Facility (SMF) is used to generate logging records.
 - Resource Recovery Services (RRS) is engaged to coordinate the transaction context between WebSphere Application Server and other Resource Managers (RMs).

WebSphere eXtreme Scale

WebSphere eXtreme Scale brings WebSphere Application Server to an even higher level of scalability. WebSphere eXtreme Scale is an optional add-on to WebSphere Application Server, and it provides an elastic, scalable, in-memory data grid. The data grid dynamically caches, partitions, replicates, and manages application data and business logic across multiple servers. WebSphere eXtreme Scale performs massive volumes of transaction processing with high efficiency and linear scalability. With WebSphere eXtreme Scale, you can also achieve qualities of service, such as transactional integrity, high availability, and predictable response times. WebSphere eXtreme Scale is supported under the Linux operating system on System z.

WebSphere eXtreme Scale can be used in various ways. You can use the product as a powerful cache, as an in-memory database processing space to manage the application state, or to build Extreme Transaction Processing (XTP) applications. These XTP capabilities include an application infrastructure to support your most demanding business-critical applications.

z/TPF

z/Transaction Processing Facility Enterprise Edition (z/TPF) is a high-performance operating system that is specifically designed to provide high availability for demanding, high-volume, real-time transaction processing for mission-critical applications. The origin of z/TPF is the first mainframe transaction application developed in 1962 as a joint project between IBM and American Airlines, called the Semi Automatic Business Research Environment (SABRE). Today, z/TPF continues to run the world's largest airline reservation systems, but has also expanded within the travel industry supporting railways and hotels.

“Our zEnterprise system uses a z/TPF operating system that we have optimized over the years for our reservations processing, and that we believe is a strategic competitive advantage.” (Misha Kravchenko, Marriott International)

z/TPF is also used in the financial industry to support retail authorizations, ATM clearing, and balance/deposit transactions. Positioned to handle the management of extreme transaction volumes, z/TPF can process tens of thousands of transactions per second from hundreds of thousands of users. Using up to 32 IBM System z servers that are loosely coupled together to access a single-image database, the system can handle over a million physical I/Os per second. As with all mission-critical transaction systems, high availability is necessary. With its unique architecture and error processing design, z/TPF provides more than 99.999% system availability.

Although z/TPF is a special-purpose system, it uses open system interfaces and packages for security (SSL), communications protocols (TCP/IP, HTTP, and SOAP), and data manipulation, such as XML. z/TPF is a runtime-only system and uses Linux for System z as a development and support platform. Programs can be developed in C/C++ and built by using the GNU tool chain. z/TPF also offers an Eclipse-based Integrated Development Environment (IDE) that provides productivity tooling, such as a context-based editor, code generation wizards for developing web services, and a source-level debugger. See Figure 16.

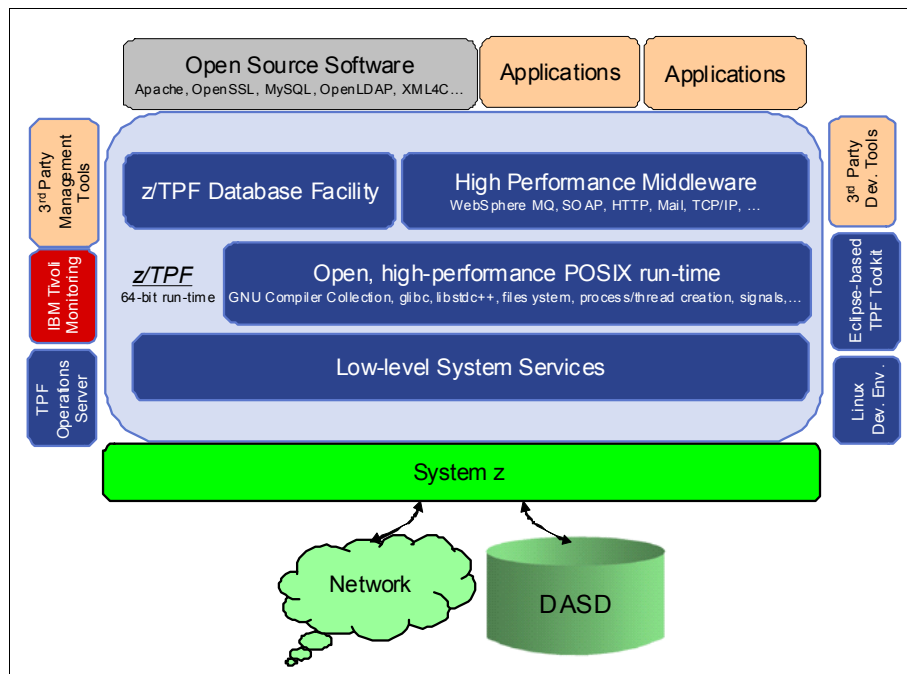


Figure 16 z/TPF overview

“Our z/TPF system is already a cloud by itself, because it scales very easily.” (Misha Kravchenko, Marriott International)

DB2 for z/OS

The primary Relational Database Management System (RDMS) on the System z platform is DB2 for z/OS. DB2 for z/OS achieves the highest levels of availability, maintainability, resiliency, and scalability through an optimized combination of hardware, operating system, APIs, and tools.

With respect to transaction processing, and especially transaction integrity, the role of the Database Manager is crucial. After all, it is the Database Manager that has the responsibility of persisting data updates and also taking all necessary actions to back out updates to an earlier commit-point after a failure.

The role of DB2 for z/OS in transaction processing and integrity

DB2 for z/OS in transaction processing and integrity has two roles:

- ▶ DB2 for z/OS participates as a Resource Manager in a “global” transaction spanning both DB2 for z/OS and other Resource Managers. In this case, DB2 for z/OS and the other Resource Manager (or multiple Resource Managers) work together to establish the two-phase commit. System z is unique in this respect because it provides highly reliable native technologies, built into the operating system, to achieve this role. Resource Recovery Services (RRS) and the CICS DB2 Attachment Facility are examples of these technologies. In this case, data integrity is fully guaranteed regardless of the type of failure.
- ▶ DB2 for z/OS performs the role of a transaction manager all by itself. DB2 for z/OS provides support for running stored procedures in a variety of programming languages and adhering fully to the ACID properties. When necessary, these stored procedures can be executed in fully isolated WLM-controlled z/OS address spaces, adding additional security and service-level commitment. Database updates initiated in a stored procedure are processed using the two-phase commit protocol, and data integrity is fully guaranteed regardless of the type of failure.

DB2 data sharing

One of the most important requirements for a highly available information system is the ability to continue to provide service without losing data integrity in case of an outage of any of the components. This requirement generally means that in the case of a system failure (in the image, server, or entire data center), updates in progress on the failing image are backed out to the latest commit-point and that the surviving image (or multiple images) continues providing the service, while the failing image is being repaired. To be able to fulfill this requirement, there are basically two underlying requirements:

- ▶ All images must have access to the same version of the data.
- ▶ The data cannot be hosted on only one physical component, which would cause a *single point of failure (SPOF)*.

DB2 for z/OS, in conjunction with the underlying z/OS operating system and the System z hardware, provides a unique solution that implements the requirements. DB2 *data sharing* is an excellent example of using an optimized combination of hardware, operating system, APIs, and tools. It takes advantage of the System z hardware and Parallel Sysplex technology so that multiple versions of an application can read from and write to the same DB2 data concurrently, even when these applications are on physically separate servers in physically separate data center locations.

DB2 for z/OS uses Parallel Sysplex technology to keep the application and subsystem state information in the *coupling facility (CF)*. This way enables a “rolling window” update approach so members of the data sharing group can work seamlessly if one node fails.

“DB2 data sharing is one of the specific System z transaction processing capabilities that distinguishes it most from other platforms.” (Andreas Wagner, GAD)

Resource recovery on System z

Recovering from failures is just as important as ensuring successful completion of work. Failures can occur for a variety of reasons. Hard drives can go bad. The power can fail. Someone can accidentally unplug the machine. A device driver can have a problem. A rogue can hack into the system. The operating system or middleware can have a defect. And, the application might have a bug. But for whatever the reason, if the system fails while you are in the middle of performing some work, you want the system to recover quickly so that you can resume that work without leaving your information in an inconsistent state.

When you are performing work, locks are generally created in transactional resources, such as the database. For example, these locks are essential to prevent Waldo and his spouse from debiting their account at the same time. However, if the system fails while it is performing work, these locks will remain on the data until someone or something releases those locks. Resource recovery is the process of releasing locks on previously in-flight work, and restoring data back to its original state of consistency.

Many middleware products on the market today assume that someone else will recover the system in the event of a failure, most often relying on the database administrator to detect that locks are being held, and to use database tools to release those locks. On the IBM mainframe, the design point is that data must be brought back to a consistent state by the middleware in case of a failure. Middleware uses the operating system functions to make this state happen. With IBM Parallel Sysplex, middleware and databases provide this consistency across multiple systems operated on one or multiple physical servers in one or multiple physical locations.

How System z helps you to ensure transactional integrity

The transaction manager has the following task responsibilities:

- ▶ Creating a transaction scope
- ▶ Registering resources into that scope as your application uses them
- ▶ Committing any changes you make to those resources within that transaction scope (or rolling back the changes if something goes wrong)
- ▶ Recovering those resources in the case of a systems failure

For half a century, IBM has been creating and deploying reliable and secure transaction managers. Evidence of that applied experience can be seen in the mainframe transaction middleware, including CICS, IMS, z/TPF, and the WebSphere Application Server family of products.

z/OS provides a common shared transaction manager that can be used among multiple subsystems, referred to as the Resource Recovery Services (RRS). WebSphere Application Server, WebSphere MQ, DB2, CICS, and IMS all defer to RRS for transaction management on z/OS. When used in conjunction with Parallel Sysplex, RRS maintains recovery state in the coupling facility (CF)¹, ensuring that recovery can be performed almost instantaneously on any remaining system if one of the systems fails.

Resource managers and protected resources

Applications need to access resources in a manner that guarantees integrity. An application must be sure that changes to data have been either made or backed out. To enable this process to happen, applications usually access resources through resource managers (RMs)².

A data resource manager, such as DB2, handles tasks, such as data access, locking and serialization, commit or rollback processing, and restart and recovery. DB2 data is referred to as a *protected* or *recoverable* resource. A protected resource is a resource that can be updated in a manner that conforms to the requirements of an ACID transaction. A protected resource does not have to be only a piece of data in a DB2 or IMS database, it can be any resource that can be changed during the process of a transaction.

Resource managers must provide the following functions to ensure that the resources they manage can be considered protected resources:

- ▶ *Locking* functions, to ensure that changes to protected resources that are not yet committed by a transaction can be seen only by that transaction
- ▶ *Logging* functions, to ensure that before and after images of changed resources are held so that the changes can be backed out, or for restart processing to reapply changes in the event of system failure

As previously noted, there must be a sync point manager that logs the application change request and decides whether the resource managers should commit or back out any changes to protected resources. This process is referred to as the *two-phase commit* process and was described previously in this guide.

If an application is updating DB2 data only, DB2 can act as both the resource manager and the sync point manager. This scenario is the simplest. When more than one resource manager is involved, things become more complex. There must be one sync point manager that talks to all involved resource managers and coordinates the commit process.

Over the years, resource managers on z/OS (such as IMS, CICS, and DB2) have developed to provide two-phase commit processing between them. This approach involves one of the resource managers taking the role of the sync point manager to coordinate the two-phase commit processing. It has required code that is written by the IMS, CICS, and DB2 developers to implement the process for every possible scenario.

¹ The coupling facility (CF) is a separate image used to store information needed for recovery if one or multiple system images fail. The CF is accessible from all images that are part of the sysplex.

² A *resource manager* (RM) is a subsystem or component, such as WebSphere Application Server, CICS, IMS, or DB2, which manages resources that can be involved in transactions. Resource managers can be categorized as *work managers*, *data resource managers*, and *communication resource managers*.

With the increasing number of resource managers now available on z/OS, there was clearly a need for a general sync point manager on z/OS that any resource manager could use. The Resource Recovery Services (RRS), a component of z/OS, provides a global sync point manager that any resource manager on z/OS can use. It enables transactions to update protected resources managed by many resource managers.

Of course, the closer the resource managers are located to each other, the less likely that transactional failures will occur. Or, in other words, the less network that is involved, the better. For this reason, our clients tell us that they limit the scope of a transaction when possible and try to stay away from distributed transactions that span different platforms. It is also why we suggest using WebSphere Application Server on z/OS if an external transaction manager is needed to coordinate System z resource managers.



The future of transaction processing

We have reviewed a number of business challenges and technology trends in the previous chapters. These challenges and trends will continue to have an impact on transaction processing systems in the coming years.

This chapter looks into the future and investigates what additional challenges and technology developments will affect our clients' TP systems. We also look at how these trends are shaping IBM solutions, including TP middleware and System z hardware.

"Similar to the way e-Business was formed by the advent of the Web, the engaging enterprise of the future is being formed by the convergence of mobile, SaaS, social networking and Big Data." **(Jerry Cuomo, IBM Fellow, Vice President, WebSphere Chief Technology Officer)**

Opportunities from new computing paradigms

Business trends and emerging technologies are opening up possibilities for significant, sometimes revolutionary changes to business models and the supporting information processing facilities. We continue to see a massive increase in computing "things," appliances, sensors, and devices, producing volumes of data. Organizations are seeking and finding new value in this data. Computing power will become increasingly accessible in the coming years.

Mobile devices will become the standard for everyday computing, replacing desktop and mobile computing in many aspects. Social networks and mobile technologies will change the way businesses create value. At the same time, new economies and massive new consumer markets are emerging.

This change opens many opportunities to do meaningful things with the increased computing power and new dimensions of information. A new *smarter computing* model is emerging that has the following traits:

- ▶ Designed for data

An infrastructure that is designed for data is able to process large volumes of information and do meaningful things with the results. Besides the traditional structured, well-governed information forms we find today in our relational databases, there is also a need to manage the increasing stream of unstructured data, informal in its content, such as tweets, video, speech, images, and so on.

- ▶ Tuned to the task

This trait means matching workloads to systems that are optimized for the characteristics of the workload.

- ▶ Managed elastically and reliably

As variability in demand increases and predictability decreases, the new computing model has to dependably cater to significant variances in data and transaction volumes. In the era of smarter computing, the infrastructure is managed with cloud technologies to improve service delivery and efficiency.

Within the context of smarter computing, we will see inevitable changes to today's TP systems. We explore these trends further in the next sections.

The growth of the mobile platform

The development of mobile solutions will continue to expand; for example, the mobile banking user base is projected to grow to over 50 million adults in the US in the next few years.

The mobile platform is leading to a reestablishment of the industry ecosystem and the emergence of new business models. New *branchless* financial services companies are emerging, providing a new set of services, sometimes combined with other commercial propositions, purely based on the mobile platform. Mobile devices are not just new interfaces to existing services, but they will facilitate new and enhanced products and services, extending the "electronic wallet," integrating diverse services, serving as a distribution channel and many other functions that we cannot even imagine today. Mobile network operators will diversify into financial services, media services, and citizen services, and will integrate with brick and mortar businesses.

The impact of the mobile channel will be most dominant in developing markets. In these growing economies, there is often no history of technology and the new mobile capabilities are embraced more quickly. This impact is accelerated by the increasing spending power of the large population. Our interviewed clients expect that globalization and emerging markets will significantly affect their TP systems in the future.

Waldo's relatives, and their friends around the world, will be online anytime anywhere, for conducting transactions, such as checking accounts, social networking, and making purchases, to conducting more complex business transactions.

New business integration models

Tighter cooperation and integration of organizations through the use of technology are expected to grow significantly in the coming years. The definition and application of standards-based business interfaces (business APIs) are emerging, enabling the technical integration of commercial services with network operator services, social networking, and other services through standardized contracts that describe the conditions for invoking the business transactions of each other.

The services developed over the past years will further mature and their application spreads quickly. For example, account aggregation services emerged, allowing clients to compile information from separate accounts across separate financial institutions. These aggregation services will be augmented with additional financial advisory capabilities, opening new business opportunities. The integration of these services is not limited to a single industry; a whole set of new capabilities will be developed, combining products from various industries into completely new offerings. The challenge for TP systems will be to support these enhanced interoperability patterns.

Cloud computing models continue to evolve and provide businesses with easy access to information processing functionality and infrastructure to realize the emerging business integration described in “Cloud” on page 23. Thus, business transactions increasingly become *hybrid* transactions, in terms of the businesses involved, and in terms of the supporting information processing capabilities involved. These business transactions consist of interactions with diverse TP systems: from self-managed, on-premises facilities to cloud-based software as a service (SaaS) services. Consequently, the elasticity and reliability of the information processing capability of the organization itself need to be assured; and equally important, potential disruptions in the service of dependent information processing facilities outside the control of the organization must be assured.

In this increasingly complex environment for completing business transactions, there is an increasing need to address what we call “multi-party transaction integrity.” Our clients indicate they see significant challenges with the current capabilities for managing distributed transactions across multiple systems, even if these transactions are all managed internally. The existing patterns and technologies for distributed transaction processing that are available for these needs require careful consideration and design and are perceived as complex, especially in error situations. Clients are telling us that currently when state must be handled in a business transaction, it is often implemented at the process management level. Handling errors during business transactions is accomplished through compensating transactions at the business level, eventually through manual activities. Compensating transactions are often too complex to automate.

With the increasing requirements for managing transactions across multiple systems, that can be geographically dispersed, hosted in the cloud, and managed outside the realm of the IT department, new solutions are required that provide global consistency capabilities for these types of transactions.

The big data challenge

The amount of data is growing ten-fold every five years, from the 800 billion gigabytes created in the last two years to petabytes of both structured and unstructured data from many sources. Data is being collected through automated business processes, web applications, user-generated content, and instrumentation of the physical infrastructure.

Sources vary from relational tables, to documents, log files, sensor readings, blogs, tweets, digital photos, webcam videos, and voice recordings. This data presents new opportunities to accomplish tasks with a better understanding of clients, partners, competitors, current state of the business, and the impact of past actions.

The data sets will be 10,000 times larger than the typical enterprise database today. For example, Twitter generates multiple terabytes of data per day. To extract business value from such a huge volume of data, businesses need high capacity and new systems that integrate traditional transaction processing capabilities with deep analytics. These new systems will allow businesses to take actions from real-time transactions, becoming more responsive to trends anywhere in the world as they develop, and not one week or one month after the event. By incorporating transactions into deep analytics, individual services to the long tail of customer segments can be provided, fulfilling a demand that would normally be ignored or recognized too late. TP systems will not only be critical for back-end reconciliations but their real-time transactions can transform how customers, employees, businesses, and partners influence each other in deciding what to offer and what they did not even know they needed.

The data types that TP systems need to cope with are changing:

- **Unstructured data**

The amount of data from image, sound, video, physical sensors, and written and spoken natural languages from multiple nationalities will continue to grow. It is at the peril of a company's future to ignore these diverse and growing sources of additional insight. TP systems were first created with data strictly categorized in hierarchical or relational schemas. In the future, TP systems will enable patterns in the data to self-emerge without first requiring a force-fit of data into predetermined, static, limited, and chosen dimensions. This enabling of patterns is the core of deep insight, so that the data can guide human beings to the unexpected, unforeseen, and rich dimensionality of relationships within data.

- **Uncertain data**

The benefit of using real-time data is better responsiveness to a changing landscape. A caveat to using data that has not been vetted is confidence in the accuracy of the data. Today, the world's data contains increasing uncertainties that arise from such sources as ambiguities associated with social media, imprecise data from sensors, imperfect object recognition in video streams, and model approximations. Industry analysts believe that by 2015, 80% of the world's data will be uncertain.

In order to make confident business decisions based on real-world data, analyses must necessarily account for many kinds of uncertainty present in very large amounts of data. Analyses based on uncertain data will have an effect on the quality of subsequent decisions, so the degree and types of inaccuracies in this uncertain data cannot be ignored.

An option is to have human beings scrutinize all data before it is used by processes and individuals. The growth and explosion in the volume of data makes this option impracticable. In the future, TP systems will be augmented to verify data using multiple sources where possible. For example, the location of vehicles, people, and supplies can be correlated with independent readings from GPS readings, traffic cams, traffic ticket citations, and checkpoint readings. Where data cannot be corroborated from independent, multiple sources, TP systems can provide confidence levels of analyses based on the pedigree of the data. Confidence levels augmenting transactional data will be used in downstream decision-making flows to decide either to get more data or to act with what degree of certitude, immediacy, and urgency.

► Streaming data

An advantage that data sources, such as Twitter, have over Google is that the data is more real-time; it is seconds old versus days old. For operations that benefit from immediate feedback, such as critical security situations or real-time mining or trading operations, the data is coming directly from the environment: mobile devices, line feeds from financial exchanges, or sensors in the physical infrastructure. Traditional TP systems that have been optimized to obtain data from vast banks of storage drums will be augmented to accept petabytes of data directly from the network.

To drive value from this mass of uncertain and unstructured data, TP systems will be augmented with functionality to dynamically create, in real time, self-organizing, self-corroborating, and confidence-rated data warehouses that can be queried and have self-describing temporal and geographical pedigrees of its data sources. TP systems will take advantage of advanced visualizations to present petabytes of data patterns to human beings that were not presupposed by data administrators.

Memory architectures will not stay passive. Included in the hierarchy of memory tiers supporting computation will be self-organizing associative memory. Current TP systems require explicit keywords, indexes, and addresses to retrieve a specific byte range of information. Advanced TP systems will take advantage of the storage and retrieval of data with self-associative memory. For example, in the future, when a TP system requests information about a “rose,” the TP system will recognize the context of the request and would provide: images of a rose, the scent of a rose, the stored temporal experiences of where the user may have encountered a rose, the price of purchasing a rose, where to purchase a rose, and how to use rose petals for culinary, medicinal, or sentimental occasions. Many possibilities arise from storing and providing data that is self-associated. The TP system can also be instructed to expand the range of the context to not only include an individual’s association with a particular datum, but the groups an individual is a part of; their family, co-workers, fellow municipal or national citizens, hobbyists the user is related to, all yielding a plethora of different patterns and associations with a particular datum, directing the final outcome of a transaction.

Applications are emerging that process streams of information in real time, analyze and correlate events, and generate actions. Streaming data and event processing applications aggregate events and store these aggregated events for further analysis or reporting, but also generate immediate actions for situations that need real-time responses. This dynamic has led to the development of new types of middleware for streaming data and event processing applications. Transaction processing middleware is enhanced with event processing capabilities. Further enhancements allow TP systems to process streams of data flowing in from the network. Additionally, facilities will be included to allow more complex event analysis and analytics in real time, to action events found in the data and interact with event correlation engine and action capabilities. Our surveyed clients tell us that they anticipate more integration based on events in the future, further enabling their TP systems to automatically send notifications when the state of a business transaction changes.

Addressing resilience in a globally connected enterprise

Global connectivity is decreasing the resilience of infrastructures at every level of business and government. As the world’s systems become more interconnected, they also are becoming interdependent and therefore more challenging to secure and insulate from threats and disruptions. This dynamic is leading directly to significant increases in large-scale and cascading failures of both digital and physical systems. The key catalysts for these failures include natural disasters, failures of communications or IT infrastructure, and human errors.

These problems can be addressed by building oversight systems that take into account the fact that any component, including human operations, might fail. These systems then isolate faults, migrate functions, and then resume operations. These principles are applicable to both IT systems and physical systems, such as energy and utilities infrastructures, transportation systems, and buildings. New approaches involve the building of robust systems that can conduct continuous sensing and model-based prediction, and then orchestrate suitable responses. Thus, TP systems will be built with self-prediction failure analysis. They will continuously monitor their transaction rate, accesses to the system, and how their resources are consumed. Their infrastructure will proactively predict when abnormalities are occurring, which might lead to cascading failures or security breaches. This development is reflected in the zAware feature of the zEnterprise EC12, which provides predictive analysis on streaming messages to give faster, more pinpointed recognition of problems before they occur.

These self-predictive monitoring components will then be coupled with model-based reasoning and the orchestration of proactive responses. Businesses need to identify and isolate failures early and avoid cascading disasters by applying these techniques in coordinated ways across IT systems and physical systems, and also business applications and services.

The globally connected nature of future transactions raises additional security and compliance concerns. To protect systems from unauthorized access, layered defense mechanisms and appliance technologies will be applied as hardened components acting as portals to the actual TP systems. Enhancements in the propagation of identity credentials across organizations facilitate the authorizations and physical and logical isolation of applications based on these identities. Furthermore, as services are increasingly provided by specialized partners, organizations that are under strict compliancy regulations will pass these compliance restrictions on to their providers. New capabilities for *federated compliance* will be put in place. The new approach is to build robust systems based on continuous monitoring and predictive analytics of the environment, model-based reasoning, advanced early warning, federated assurance, and orchestration of a proactive response.

Evolution of hybrid architectures

Hybrid system architectures are realized on various levels. On the application level, architectures have evolved to composite architectures, and are built upon loosely coupled components that are deployed in a geographically dispersed landscape, and that consist of cloud and on-premise IT services. In this landscape of composite applications, the supporting middleware solutions, including TP middleware, must support the business processes that span the hybrid middleware.

Server architectures are becoming workload-optimized and more heterogeneous. The hybrid nature of systems is realized through a tighter integration of scale-up and scale-out architectures. Such an integrated platform allows diverse workloads to be run on the hardware architecture that is the best fit for the task.

TP systems will take advantage of new hybrid server architectures in a number of important ways:

- ▶ Field Programmable Gate Arrays (FPGAs)

An FPGA is a hardware component that can be programmed for a specific task that can then be very quickly and efficiently executed. FPGAs are increasingly used to run specific computational functions, thus off-loading the central processor and enabling a higher level of parallelism.

- Direct memory addressable accelerators

Special processors that execute in parallel with the central processor to accelerate a special function, and have direct access to the memory shared with the central processor. Such processors will help to manage the big data challenge by extracting patterns from raw multi-modal data (image, sound, video, speech, and so on).

- Transactional memory

Transactional memory is a hardware technique to support parallel programming; it is a generally more efficient alternative to the use of locks in managing access to shared resources by parallel threads. Instead of using locks, with transactional memory, the state of the shared resource before and after an atomic operation is checked. If there is no change to the resource, the operation is committed; if there is a change to the resource, the atomic operation is backed out and retried.

- Compiler enhancements

Compilers will not only generate machine code for a specific platform, but take into account new and improved hardware capabilities, such as transactional memory, simultaneous multithreading, and multi-modal accelerators, that can be usefully employed to support the habitat of a transaction.

These hardware advances and the evolution of hybrid systems will allow TP systems to accelerate a wider range of functions, which will help to address the more complex processing requirements and higher transaction throughput expected by our clients.

"In our enterprise architecture, the mainframe is our transaction processing box; it is optimized for transaction processing, and we expect it to be further optimized for this in the future." **(ABN AMRO Bank)**



Summary

In this paper, we reviewed the past, present, and future of transaction processing and transaction integrity. We described how we arrived at where we are and where the future of transaction processing might lead us. Most of the challenges and requirements that led to the development and evolution of transaction processing systems are still applicable today, and recently, we have seen some intriguing developments. Predictions for future business and technology trends point at the increased importance of transaction processing systems.

The impact of innovations and trends in various areas was evaluated in this paper. Some of the most thought-provoking innovations and trends are mobile platforms, new business integration models, and big data. The next generation of transaction processing systems will address new challenges and needs as these trends emerge.

The world's most important solutions for transaction processing run on IBM systems. The distinguishing capabilities and the leadership position of System z were also highlighted in this paper. System z has been the flagship platform for transaction processing for almost half a century and runs key IBM software technologies for transaction processing, such as CICS, IMS, z/TPF, WebSphere Application Server, and DB2.

The traditional strengths of System z for transaction processing systems is its scale-up architecture that facilitates a mix of high-volume and time-critical workloads with high-integrity requirements on the accessed shared data. The business and technology trends highlighted in this paper indicate that this type of transaction processing capability continues to be key in many organizations. Moreover, new capabilities that address the magnitude and the functional needs of tomorrow's transaction processing requirements will extend the current strengths of the System z platform.

There is more to System z in this context. Technology that became available on System z in 2010 provided for a new paradigm, hybrid computing. Since then, in combination with the zEnterprise BladeCenter Extension and in combination with IBM PureSystems™ systems, the System z environment also addresses the capability for a scale-out processing model of transaction processing. With this capability, transaction processing workloads can be combined on systems that are geared toward the scale-out application model. This combination of capabilities assures that System z is the unique platform that enables organizations to run transaction processing workloads on technology that is tuned to the task, and is perfectly positioned to address today's and future transaction processing needs.

The team who wrote this guide

This guide was produced by a team of specialists from around the world working at the International Technical Support Organization (ITSO).

Alex Louwe Kooijmans is a Senior Architect at the Financial Services Center of Excellence at IBM Systems and Technology Group. Prior to this position, he spent almost 10 years in the International Technical Support Organization leading IBM Redbooks® projects, teaching workshops, and running technical events with a focus on using the IBM mainframe in new ways. Alex also worked as the Client Technical Advisor to various banks in The Netherlands and performed various job roles in application development. His current focus is on modernizing core banking systems and the role of the current IBM mainframe technology.

Elsie Ramos is a Project Leader at the International Technical Support Organization, Poughkeepsie Center. She has over 30 years of experience in IT supporting various platforms, including System z servers.

Niek De Greef is an Executive IT Architect working for IBM in The Netherlands. Niek has more than 20 years of experience in IT. His areas of expertise include application integration, software engineering, and infrastructure architecture.

Dominique Delhumeau is a Distinguished Engineer working for IBM in France. Dominique is a Certified Architect and a member of the IBM Academy of Technology. He has more than 30 years of experience in IT. His areas of expertise include enterprise architecture and application integration in banking and insurance domains.

Donna Eng Dillenberger is a Distinguished Engineer at IBM Research. She joined IBM in 1988 and has worked on future hardware simulations, mainframe operating systems, workload management, scalable JVMs, scalable web servers, stream servers, machine learning algorithms, WebSphere, and was an IBM Chief Technology Officer of IT Optimization. She is a Master Inventor, a member of the IBM Academy, and an Adjunct Professor at the Columbia University Graduate School of Engineering. She is currently working on systemic risk, hybrid server architectures, and mainframes.

Hilon Potter is a Chief IT Architect working at the IBM Design Center in Poughkeepsie. He has worked in the IBM Systems and Technology Group for 29 years and for the last 13 years at the Design Center where he supports System z.

Nigel Williams is a Certified IT Specialist working in the IBM Design Centre, Montpellier, France. He specializes in enterprise application integration, security, and SOA. He is the author of many papers and IBM Redbooks publications, and he speaks frequently about CICS and WebSphere topics.

Special thanks to **Marcia Harelik**, WebSphere on z Market Management, for her leadership in sponsoring and enabling this IBM Redguide project.

Thanks to the following people for their contributions to this project:

Marcel Däppen
CTO for zEnterprise Systems, UBS WM&SB, Switzerland

Misha Kravchenko
Vice President Information Resources, Marriott International

Johan Meijer
Infrastructure Architect, ABN AMRO Bank, The Netherlands

Ben Noordzij
Enterprise Architect, ABN AMRO Bank, The Netherlands

Arjen van Verseveld
Technology Manager, ABN AMRO Bank, The Netherlands

Andreas Wagner
Head of DBA department, GAD, Germany

Hans Baken
Infrastructure IT Architect, IBM GTS SO Delivery, ABN AMRO account, The Netherlands

Chris Barwise
System z IT Architect (zITA) and zChampion, IBM UK

Ella Buslovich
International Technical Support Organization, Poughkeepsie Center

Martin Braun
Senior IT Architect; Industry Architect for GAD, IBM Germany

Mark Cocker
CICS Technical Strategy and Planning, Hursley laboratory, IBM UK

Jonathan Collins
Product Line Manager - z/TPF, WTCF & ALCS

Lian Cheng Deng
Client Technical Architect, IBM China

SangSoo Han
CICS Business Development Manager, IBM China

Rob High
IBM Fellow, Vice President and CTO for Watson

Claus Jensen
STSM and Chief Architect SOA-BPM-EA Technical Strategy

Yannick Le Floch
IT Architect, European Design Center, Montpellier, IBM France

Helene Lyon
IBM Distinguished Engineer and European IMS Architecture Team Technical Executive

Colette Manoni
STSM, TPF Design and Development

Shyh-Mei Ho
IBM Distinguished Engineer; IMS On Demand SOA Chief Architect

Ian J Mitchell
IBM Distinguished Engineer; CICS Transaction Server

David Rhoderick
IBM System z Competitive Project Office

Sandy Sherrill
IMS Worldwide Market Manager

Matthew Sykes
WebSphere Application Server Development

Darren Taylor
Software IT Architect, IBM UK

Mauro Tibolla
Client IT Architect for UBS, IBM Switzerland

Anna Wang
Solution Architecture - Banking and Financial Services Center of Excellence

Charles Webb
IBM Fellow, System z Processor Design

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Stay connected to IBM Redbooks

- ▶ Find us on Facebook:
<http://www.facebook.com/IBMRedbooks>
- ▶ Follow us on Twitter:
<http://twitter.com/ibmredbooks>
- ▶ Look for us on LinkedIn:
<http://www.linkedin.com/groups?home=&gid=2130806>
- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:
<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>
- ▶ Stay current on recent Redbooks publications with RSS Feeds:
<http://www.redbooks.ibm.com/rss.html>

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.




Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>



The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

BladeCenter®	MVST™	WebSphere®
CICS®	Parallel Sysplex®	z/OS®
DataPower®	PureSystems™	z/VM®
DB2®	Redbooks®	z/VSE®
DRDA®	Redguide™	zEnterprise®
IBM®	Redbooks (logo)  ®	
IMS™	System z®	

The following terms are trademarks of other companies:

Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.