codit
integrating your business

white paper

CONSIDERATIONS FOR MAINTAINING CHAIN INTEGRITY AFTER A SYSTEM OR DISASTER RECOVERY

codit.eu
linkedin.com/company/codit
twitter.com/CoditCompany
facebook.com/CoditCompany

codit
info@codit.eu

codit

TABLE OF CONTENTS

1. ABOUT CODIT

Large, international companies often struggle to easily exchange data with their subsidiaries, customers, suppliers and other business partners. Many also face challenges with the upcoming trends such as Cloud, SaaS apps, Mobile, Internet of Things, Big Data...
Companies not only have to be aware of them, yet have to adopt these technology changes strategically to gain competitive advantage. That is exactly what our expert teams do: we integrate business applications with the newest Microsoft technologies.

AUTHOR

Jasper Defesche

REVIEWERS

My thanks goes to the whole Axon Olympus team for reviewing this white paper.

2. PROBLEM DEFINITION

Microsoft BizTalk Server (BizTalk) is the most appropriate tool for connecting applications. It offers a solution to a wide variety of integration challenges, such as connec-ting systems and guiding processes (orchestration).

One aspect that is often not taken into account is a restoration following a system failure (disaster recovery). Aside from the necessary effort needed to restore the failed system, the surrounding systems also need to be considered. A backup always concerns a situation from the past that has been saved. By performing a recovery of a single system (restoring the last backup) the recovered system is running behind in comparison to the surrounding systems (i.e. no longer running synchronously). A system restore, therefore, cannot be executed in isolation but must be considered within the context of the chain of systems involved.

By taking the possible necessity of a chain recovery into account, one is able to make well-considered choices beforehand, rather than having to make these choices under pressure when the system failure occurs. Often, the best choice is no longer possible at that stage. The purpose of this document is to provide a guide for functional recovery, which will allow companies to determine the correct strategy to be followed. It is intended for persons involved in the design and implementation of integration projects. In addition, project leaders are given an insight into the problems that occur with chain administration and error recovery. As each situation is unique, it is not possible to describe all the chain recovery possibilities in this white paper. Therefore, by way of a number of examples, the key considerations will be described.

Using this as a basis, a summary can be made of the questions that need to be asked in order to decide what the correct solution is for a functional chain recovery for a specific organisation. Finally, the implications for BizTalk in the areas of development and administration will be considered.

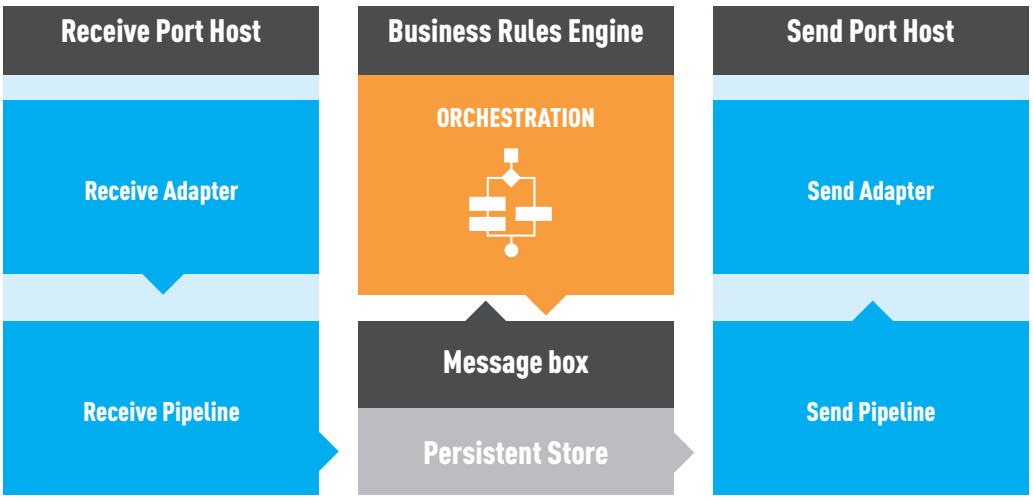
Although many subjects are not platform-specific, the deployment of BizTalk forms the starting point of this white paper.

3. CONTEXT

Before proceeding further, it is important to reflect on the significance and the role of Microsoft BizTalk Server.

3.1. MICROSOFT BIZTALK SERVER

Microsoft BizTalk Server is a server product for enterprise application integration that can be used as a Business Process Management Server or Enterprise Service Bus. This enables companies to integrate, automate and manage business processes. Powerful tooling makes it possible to solve a wide range of integration issues in a standard way.



The figure above shows a schematic overview of Microsoft BizTalk Server (valid for all versions). Every message that enters Microsoft BizTalk Server is first received by a Receive Adapter. After transforming the message, a (xml) message is created that is stored in the BizTalk database (MessageBox). The MessageBox is responsible for storing incoming messages. Messages are routed by means of the subscriptions mechanism.

This can be seen as a subscription of interested systems to a message. This can be an orchestration to organize and supervise a (complicated) business process, but also an outgoing stream to a third party or another system. When exiting Microsoft BizTalk Server the message can be translated once again by the Send Adapter before being delivered to the recipient.

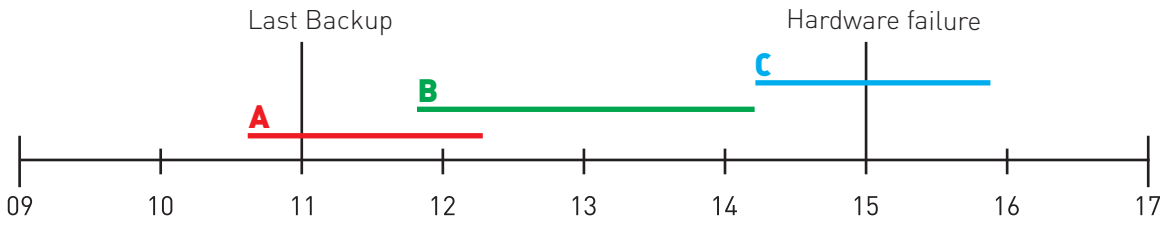
An overview of standard Microsoft BizTalk Server functionalities:

- Message transformations
- Business Rule Engine
- Business Process Management
- Trading Partner Management
- Business Activity Monitoring
- RFID

Source: Understanding BizTalk Server 2006 R2, David Chappel, August 2007

BizTalk is built entirely on the .NET framework and thereby makes extensive use of WCF technology, which gives it the flexibility for creating extensions.

The MessageBox is the heart of BizTalk. This component consists of a Microsoft SQL Server database. Microsoft SQL Server is a proven product which, in the context of BizTalk, ensures that message processing occurs transactionally. This also makes BizTalk reliable, robust and scalable. In the event that the SQL Server database server fails, a restore from a previously made backup must be executed. Through this action there is a chance that messages could be lost. With regard to the recovery chain, as an example, a scenario is described below in which the BizTalk environment fails due to a hardware failure.



Here, it can be seen that the hardware failure occurred at 3 PM. The last backup was performed at 11 AM, which means that after recovery (restore) BizTalk returns to the situation that existed at 11 AM. This may mean that a previously executed process will be picked up again (A). It may also be possible that a successfully processed message handling operation is no longer visible in BizTalk (B). But the situation may also occur that a process was started but not completed, and now no longer exists and will never start (C). It is clear, in the first place, that situations A and C are undesirable because in such cases the systems and processes within the chain can become out of sync. Moreover, situation B is not acceptable with a view to governance and non-repudiation.

There are several conceivable solutions by which this scenario can be partially offset. These include the deployment of a cluster or the use of a virtualization solution in combination with redundant hardware. This enhances availability and error tolerance, but can never completely ensure that such a scenario does not occur.

The use of BizTalk makes it possible to create multiple chains from diverse systems. When creating and expanding such chains it is important to take into consideration which actions must be executed when a component of the chain fails and needs to be restored.

This can include the following situations:

- Hardware failure
- Fire in the data centre
- Human error
- Software failure

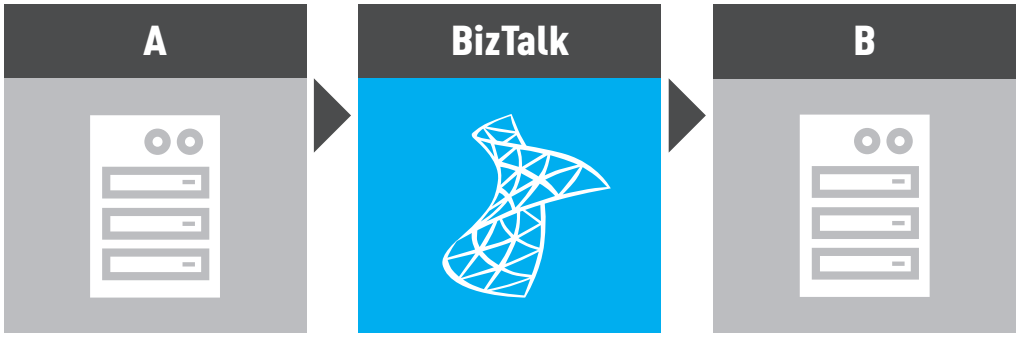
Merely reinstating a component within the chain via a backup (by means of a standard restore) can have far-reaching, undesirable and adverse consequences. It is therefore important to consider the possibilities in advance so that deliberate choices can be made instead of adopting ad-hoc solutions at the time of the emergency. In the following chapter we will further discuss the possible scenarios and the choices that need to be made to facilitate a functional chain recovery.

4. SCENARIOS

Functional chain recovery involves restoring the chain in the event that one or more systems within this chain fail. There are certain important aspects of this recovery to consider. In the following chapter we will elaborate further on these aspects by way of two examples. The two examples used are:

- Preserving the integrity of the chain
- Determining the impact on the business process (both when system failure occurs as well as the ability to contribute to a recovery)
- Administration issues
- Technical characteristics within the chain that contribute to a recovery

These examples are positioned at the extremity of either simplicity or complexity. They thereby demonstrate that more than one strategy is possible. Before considering these examples it must be assumed that the company has a good working backup and restore strategy; (see also: http://msdn.microsoft.com/en-us/library/aa577989.aspx



4.1. SIMPLE INTERFACE

A simple interface is taken as the first example. The figure above shows Server A sending messages to Server B via BizTalk. Should Server A fail and be restored, it is possible that it will uncontrollably resend messages that have already been sent. We must pause and consider the possible consequences of this behaviour. The key here is to determine whether the message handling in server B is idempotent. After all, if this is not the case, data integrity is endangered and remedial measures should be taken. Idempotence means that repeatedly sending an identical message leads to the same result. A simple example of an idempotent service is a calculator: two plus two equals four, and always will.

If server B fails, it will lag behind after recovery. Both server A and BizTalk no longer have the original message. If this does not matter to the business then nothing needs to be done. Otherwise action must be taken in server A in order for the messages to be resent. If this is not possible the alternative remains of entering the information manually in server B or to read in the data manually from the database of server A.

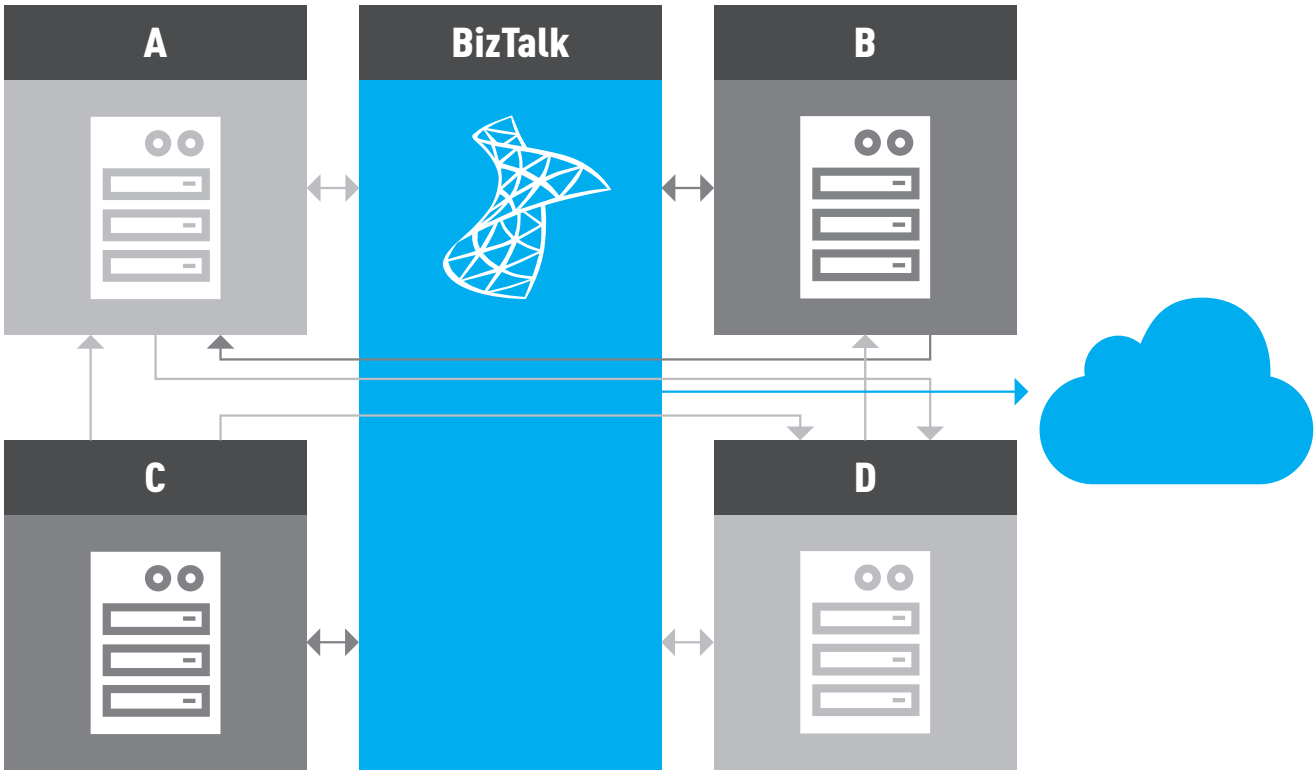
In the case of a simple chain, which can easily be restored, the business process is not usually instantly blocked as soon as one of the systems involved fails. It is of course important for the administration department to understand the chain, but in-depth knowledge and experience of the workings of a simple chain are not directly necessary. Often work can be executed in ad hoc fashion as the risk of data inconsistency is relatively small.

4.2. COMPLEX CHAIN

A complex chain consists of several systems that exchange messages amongst each other, expect a response, and are, for instance, dependent on the same master data. See figure above for an impression of this:

There are different message streams within the chain:

- Server A sends messages to Server C and expects a synchronous answer
- Server C sends messages to Server D and expects an asynchronous answer
- Server D sends messages to Server B but does not expect an answer
- Server D sends messages to Server C but does not expect an answer
- Server B sends messages to Server A but does not expect an answer
- Server A sends messages to Server D but does not expect an answer
- There are messages that are sent outside the organization



It is clear that in such a situation the failure of one (or more) parts of the chain may have a greater impact than in the case of a 1-to-1 interface. It is therefore much more complicated to identify and repair the damage. All the interdependent systems will have to be checked. This can also include systems that are not technically linked but are process-linked (manually). It is important to note that BizTalk, in the figure above, has a much more prominent role and is therefore a critical part of the chain.

The administration department must have thorough knowledge and experience of the chain, not only to repair system failures but also to ensure correct handling of functional errors that can occur anywhere in the chain.

Now that a picture of the problem has been outlined, it is important to answer the question of how to determine the best strategy for a functional chain recovery.

5. ANALYSIS

The previous chapter showed that there are a range of possibilities and/or situations which must be taken into account. Each situation is unique and requires a specific solution. In order to make a correct choice, an analysis must be made. By answering the following questions in this analysis it is possible to determine the best methodology for an organisation:

- What is the overlying business process executed by the chain?
- How critical is the chain of software systems to the business process?
- What are the human activities within the chain?
- Is each chain component of equal importance?
- How complex is the chain in technical terms?
- To what extent is it possible for systems within the chain to resend messages or receive duplicate messages?
- To what extent are external systems part of the chain?

In this chapter, each question will be addressed in a separate section.

5.1. WHAT IS THE OVERLYING BUSINESS PROCESS EXECUTED BY THE CHAIN?

The starting point of the analysis is to gain an understanding of the business process. The business process determines to a great extent how systems and people work together. This could mean that by changing a step in the business process, technical problems that may have arisen can be avoided. Take for example assumptions made from a business viewpoint concerning the technical solution. Based on the current customs and or processes, requirements and requests can arise that appear logical but have a substantial technical impact. Therefore, there is also a risk that the functional chain recovery could become more complex. A critical examination of the proposals made from a business viewpoint can therefore do no harm.

5.2. HOW CRITICAL IS THE CHAIN OF SOFTWARE SYSTEMS TO THE BUSINESS PROCESS?

With this question it becomes apparent how important the chain actually is to the business process. The term 'critical' can be broken down into a number of underlying questions:

- What is the business impact if a chain (or part thereof) is not available? A direct financial impact, for instance, or damage to the company's reputation? Departments could run out of work or customers may not be able to place orders.
- Are there any possible workarounds and are these scenarios known and practiced by the appropriate persons/ departments? Technical interfaces can often also be replaced by manual operations. This is not usually desirable, but may be unavoidable in certain cases.
- What are the consequences for the business process when messages are lost? There are situations when the loss of messages is not damaging. For instance a batch process whereby all the item data is refreshed and forwarded to a website at regular intervals. It is usually not serious if a number of runs have been lost: the process begins anew as soon as the chain is restored. Contrary to this is the case of a bank that processes financial transactions: the loss of messages is unacceptable here.
- Is the late arrival of messages inconvenient for the business process? For some interfaces within a chain, reliability is far more important than speed.

5.3. WHAT ARE THE HUMAN ACTIVITIES WITHIN THE CHAIN?

Functional recovery is not just about systems but also involves people. There are situations when human actions form an essential part of the chain. Here, it is also important to consider the possible impact of restoring a system within a chain after a failure. For example, a department or person may have to undo a filing or take measures to avoid a filing request being executed twice.

5.4. IS EACH CHAIN COMPONENT OF EQUAL IMPORTANCE?

A chain of systems typically consists of several stand-alone connections (interfaces). Not every interface has to be of equal importance. The interface that brings orders into the ERP system may be of far greater importance than the interface that transfers certain master data. An interface that transfers customer data is not directly critical: when a new client is entered in the ERP system but does not automatically arrive in the destination system, this process can often also be performed manually. In this case it is important to verify that the client is not created again (automatically) when the system is restored.

5.5. HOW COMPLEX IS THE CHAIN IN TECHNICAL TERMS?

This question deals with the technical side of the story. There may be interface components for which highly complex logic is required to integrate with a system (i.e. make the system available to other systems). At the same time, these components will have little functional value. There is a basic difference, for example, between a system that supports web services and a closed system, i.e. a system where no connection from outside can be made. The latter situation often involves complex logic to achieve connectivity anyway. Recovering from error situations and system failures is generally more difficult and therefore more costly than normal.

5.6. TO WHAT EXTENT IS IT POSSIBLE FOR SYSTEMS WITHIN THE CHAIN TO RESEND OR RECEIVE MESSAGES AGAIN?

Linked to the previous question is the question whether systems within the chain can store sent and received messages and whether these can then be sent out and received again. If a system has this functionality, restoring will be much easier, since it can be determined exactly which messages have already been received and which messages should be sent out again. Care must be taken when resending messages, as the other systems within the chain must be able to cope with this. BizTalk can facilitate in this by parking message streams and letting them run in a controlled manner. When using this alternative it is important that the (technical) administration department responsible for BizTalk is well informed and up to date on the processes. When such a message request arrives via the help desk, for instance, it can be acted upon more quickly.

5.7. TO WHAT EXTENT ARE EXTERNAL SYSTEMS PART OF THE CHAIN?

Another important aspect is the extent to which external systems of customers or suppliers are linked with the internal systems. Every business process is, to a greater or lesser degree, linked to external parties. This may be by telephone, paper, or mail through to fully automated systems (e.g. XML or EDI exchange messages). When an internal system is restored, the impact on external parties/systems for each variant must be examined. It may be necessary to stop messages to a supplier (to prevent duplicate orders) or a supplier may have to resend some invoices.

5.8. CONCLUSION

By answering the above questions, it becomes clear where the risks lie, and thereby where the most attention must be given. In order to limit the impact on the production environment as much as possible, it is very important to take error situations and their subsequent recovery into account during the implementation process. This is one of the issues which the following chapter is concerned with.

6. IMPLICATIONS DEVELOPMENT AND ADMINISTRATION

The maxim that prevention is better than cure is also applicable to functional recovery within the chain. Even though certain elements of the chain may be neither important nor technically complicated, it is still a good idea to look at certain areas within the design and development process that may make chain recovery easier. This chapter provides an overview of a number of methods that can help to simplify a functional chain recovery.

6.1. INCREASE THE AVAILABILITY AND RELIABILITY OF CHAIN COMPONENTS

By increasing the availability of chain components, the chain as a whole will be more reliable and less prone to failure. Therefore, consider virtualization, load balancing or clustering.

6.2. CONSIDER MESSAGE FORMATS

When services are idempotent, it does not matter if the same message is presented several times. As a result, when recovering a message-sending system it is not necessary to take into account the systems that receive these messages. Because of this the message formats need to be considered. As an example we can take the message to a bank: "transfer 100 euro to account X". A money transfer will be executed each time this message is sent. By incorporating a field in the message definition for a unique transaction key, it is possible for the receiving systems to ignore duplicated messages since they will have identical keys.

6.3. REFLECT ON SERVICE COMPOSITION

By considering the services (composition) it is possible to simplify the interaction between systems. One way of achieving this is the application of chunky interfaces. With chunky interfaces a functional objective is covered as far as possible by a single message.

This contrasts with chatty interfaces whereby a number of messages are sent back and forth to reach this functional goal. A simple example is a service that comprises two operations: one to enter the order and one to provide each order line. It is much more practical to send the order with the order lines in one operation, as this will ease error recovery.

6.4. CONSIDER EXTRA DECOUPLING POINTS

When certain parts within the chain are unreliable or when recovery is time consuming, it is worth considering the creation of extra decoupling. Consider, for instance, extra archiving of sent or received messages. BizTalk can facilitate this by initiating message tracking.

6.5. PROVIDE A SUPPORT MANUAL WITH THE SOFTWARE

When providing support manuals developers are obligated to document all known errors and supply these along with other documentation. Problems and error situations encountered during the development process are described in the support manual, along with the actions executed in solving the problem or the error situation. This provides the administration department with a basis for understanding error situations and possible recovery actions they can take after the solution has been deployed. It is common that an error message in a system can be traced back to an incorrect action much earlier in the chain or in another system. Lastly, describing these situations avoids developers having to be involved immediately when production problems occur. After acceptance the support manual will be maintained by the administration department.

6.6. INVOLVE THE ADMINISTRATION DEPARTMENT FROM THE START

Involving the administration department from the beginning of the project ensures that attention is given to all matters involving the administration. As a general rule, organizations have not always defined their demands and wishes concerning administration beforehand. Consider, for instance, scalability, extensibility of the software and performance of the chain. A (horizontal) scalable solution contributes to better reliability and availability, reducing the likelihood that an error recovery action will be required. A comprehensive hand-over to the administration department assures that the organization is prepared for error situations. Make and record clear agreements.

A backup and restore strategy must be set up in collaboration with the project team and the administration department. This should be tested and delivered with the project. This way, the procedure to be followed in the event of a system error will have been set-up before actual events occur.

6.7. AIM TO INCLUDE ESB PROPERTIES IN THE INTEGRATION LANDSCAPE

An ESB (Enterprise Service Bus) is a method for integrating systems and processes. The most important objective of this method is the maximum decoupling of systems and processes. In this way, integration chains are made simpler and more predictable. Because the systems depend less on each other, both implicitly and explicitely, error recovery will also be easier. Consider, for example, a system that publishes master data that is restored after a system failure. When an ESB service is used for sending revised master data it is possible to stop previously sent messages being sent again to certain subscribed systems.

6.8. THE USE OF BIZTALK DURING SYSTEM RECOVERY

There are many standard possibilities within BizTalk for archiving and holding properly processed messages (parking). It is therefore important to bear these possibilities in mind when an error recovery is required. This requires that the BizTalk administrator has a clear picture of the context of the chain. Because these possibilities often concern exceptional situations it is important to prevent standard procedures getting in the way here - a management procedure for instance, whereby parked messages are offered again on a daily basis: this can thwart an error recovery completely.

6.9. COMMUNICATION WITH EXTERNAL PARTIES

Communication with external parties/systems runs preferably via one route so that all the entry checks, message integrity checks, etc. can be carried out in one location. This also reduces the dependency on the suppliers. Aside from the fact that BizTalk is very capable of fulfilling this role it also provides the possibility of better preparation towards managing error recoveries. As this particular scenario deals with an external party, it will demand more preparation to

avoid possible reputation damage. BizTalk can also be of assistance here by temporarily parking messages and then allowing them through or deleting them in a controlled manner.

6.10. APPLICATION AND / OR SUPPLIER SELECTION

When selecting an application or supplier for a system that has to be linked to BizTalk it is important to examine:

- The technical features of the application with regards to chain recovery. Make this a part of the selection criteria to ensure that the necessary tooling and methods for chain repair are provided.
- The technical and functional capabilities of the supplier in the areas of interfacing, error handling and chain recovery. Many suppliers only have limited knowledge of integration and there is a risk that sub optimisation may occur (unnecessary customization for instance)

6.11. SYSTEM RESPONSIBILITIES

It is very important that appropriate responsibilities are assigned to each component in the chain, especially the integration layer. Processes that run across multiple systems must not themselves be implemented in source or target systems, because this would entail the undesirable introduction of dependencies, and the risk of lock-in and more complex chain recovery. By properly defining responsibilities there will be far fewer connections between systems, making an eventual chain recovery less complex.

6.12. TRANSACTION MANAGEMENT WITHIN BIZTALK

Transaction Management within BizTalk includes the controlled handling of transactions, subtransactions and the ability to restart. A BizTalk transaction is the collection of actions that must be executed as a whole. For instance, invoking a webservice after sending an e-mail. Critical steps within a transaction are acknowledged. These are steps where errors can occur, and in practice they are often invoked from external systems via BizTalk. Critical steps are handled as subtransactions. When a subtransaction goes wrong, the transaction can be restarted but account must be taken of subtransactions that have already been completed successfully. Reversing (compensating) previous subtransaction(s) is also a possibility.

Central to transaction management is the idea that technical error processing is seen as a functional process.

Without the application of transaction management each error will result in an error message in BizTalk, causing a running process to stall without being able to be restarted easily. By using transaction management, the reliability of the chain as well as the ability for the chain to be restarted, are enhanced. As thus transaction management is useful for catching errors and taking corrective measures.

6.13. DEVELOPMENT STANDARDS

By recording and applying development standards and guidelines, interfaces become more uniform. Consider here the nomenclature of interface components, uniform error handling, error codes and logging. This allows for a smaller set of procedures and agreements to work with, thereby reducing administration effort. The predictability of the error recovery is thus increased.

7. CONCLUSION

Challenges exist in the area of functional recovery when using chains to implement business processes. Often this fact is forgotten or is taken into account too late. By considering it from the start of the realisation of a chain, an informed choice can be made about how to proceed when a chain must be functionally restored.

As no two chains are alike (nor are interfaces within a chain), it is important to make an analysis based on the correct questions so that the optimal choice can be made.

BizTalk can play an important role during the functional recovery of a chain component. Here as well, one needs to make advance considerations regarding the installation and configuration in order for BizTalk to facilitate a functional recovery.

8. CONTACT

Would you like more information or need advice?

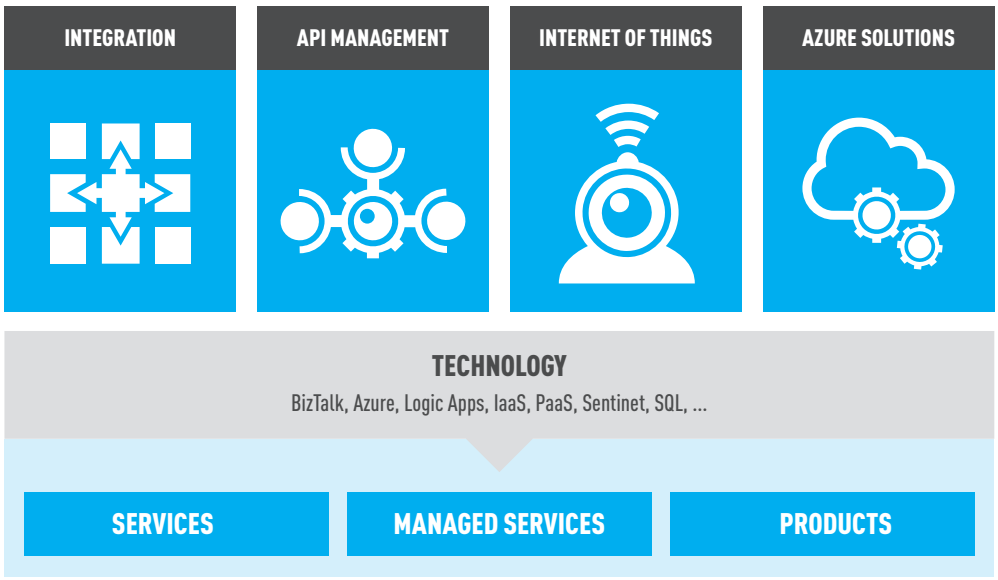
Contact us to plan a meeting: sales@codit.eu | www.codit.eu | www.codit.eu/blog

9. GLOSSARY

Term	Definition
Disaster recovery	Restoring a system failure using a predetermined and rehearsed protocol
Idempotence	A property of a service whereby the same result occurs when resending a previously received message. See also: http://en.wikipedia.org/wiki/Idempotence
Message Tracking	Storing data about a BizTalk process. For instance, data such as date of receipt and type of message. The message content can also be stored with the help of tracking.
Enterprise Application Integration	Tools and techniques to allow business applications to work together. See also: http://en.wikipedia.org/wiki/Enterprise_application_integration
Business Process Management	An approach to streamlining all aspects of an organization to meet the customer's needs and wishes. See also: http://en.wikipedia.org/wiki/Business_process_management
Enterprise Service Bus	An architectural pattern for providing, maintaining, and monitoring services. See also: http://en.wikipedia.org/wiki/Enterprise_service_bus
Message Transformation	Converting messages from one format to another.
Business Rule Engine	Responsible for managing and executing business rules.
Trading Partner Management	Component within Microsoft BizTalk Server by means of which B2B communication between companies can be set up and managed. See also: http://msdn.microsoft.com/en-us/library/ee920490(v=bts.70).aspx
Business Activity Monitoring	Tooling which enables monitoring of business processes in real-time.
RFID	Radio Frequency Identification
WCF	Windows Communication Foundation. A part of the Microsoft .NET framework that provides a uniform programming model for communication between systems. See also: http://msdn.microsoft.com/en-us/netframework/aa663324
EDI	Electronic Data Interchange. A standard for the exchange of electronic messages between parties. See also: http://en.wikipedia.org/wiki/Electronic_data_interchange
Scalability	The degree to which a system can grow to meet increasing demand. Vertical scaling concerns the increase of server capacity (more memory, additional processors). Horizontal scaling means deploying additional servers to perform the same job.
Non-repudiation	Mechanism to create the necessary security concerning the reception of a message so that the sender cannot deny having sent it.

codit

connecting (is) everything



codit.eu

linkedin.com/company/codit

twitter.com/CoditCompany

facebook.com/CoditCompany