Redbooks

ibm.com/redbooks

Getting Started with IBM Z Resiliency

Bill White

Marc Coq

Pabitra Mukhopadhyay

Carlos Alberto Pistillo

David Raften

Mai Zeng



IBM

Redbooks



IBM Redbooks

Getting Started with IBM Z Resiliency

March 2020

SG24-8446-00

Note: Before using this information and the product it supports, read the information in “Notices” on page vii.

First Edition (March 2020)

© Copyright International Business Machines Corporation 2020. All rights reserved.
Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices vii
Trademarks viii

Preface ix
Authors ix
Now you can become a published author, too! x
Comments welcome x
Stay connected to IBM Redbooks xi

Chapter 1. Resiliency is key to the survival of a digital business 1
1.1 What's the cost of downtime 2
1.1.1 Fixed costs 2
1.1.2 Lost revenue 3
1.1.3 Intangible costs 3
1.1.4 Balancing IT risk and costs of mitigation 4
1.2 Measuring availability 4
1.2.1 Availability objectives 5
1.2.2 Recovery objectives: RPO and RTO 6
1.2.3 Aspects of availability 7
1.3 A path to higher availability 9
1.3.1 Model 1: Starting with a resilient and reliable base 10
1.3.2 Model 2: Reducing the duration of outages with failover capability 11
1.3.3 Model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS 12
1.3.4 Model 4: Adding GDPS Continuous Availability for maximal resilience 13

Chapter 2. Resiliency is built into the IBM Z platform 15
2.1 Why redundancy and RAS 16
2.2 Infrastructure layer 17
2.2.1 Compute 17
2.2.2 Storage 22
2.2.3 Network 24
2.2.4 Virtualization 25
2.2.5 Range of resiliency for IBM Z environments 27
2.3 Operating system layer 31
2.3.1 z/OS 32
2.4 Middleware layer 39
2.4.1 IBM MQ for z/OS 39
2.4.2 IBM Db2 for z/OS 42
2.5 Application layer 45
2.5.1 Customer Information Control System 46
2.5.2 Information Management System 62
2.5.3 IBM WebSphere Application Server for z/OS 73
2.5.4 Batch processing 78
2.6 Management layer 81
2.6.1 IBM System Automation for z/OS 82
2.6.2 IBM Z Operations Analytics (IZOA) 84
2.6.3 IBM OMEGAMON product family 85
2.6.4 CICS tools 85

© Copyright IBM Corp. 2020. All rights reserved.

2.6.5 IMS tools 86

Chapter 3. IBM Z resiliency models 93
3.1 Model 1: Starting with a resilient and reliable base 94
3.1.1 Description 94
3.1.2 Considerations 94
3.1.3 Use case: planned outage 95
3.1.4 Use case: unplanned / unexpected outages 96
3.1.5 Resiliency optimization and next steps 96
3.2 Model 2: Reducing the duration of outages with failover capability 97
3.2.1 Parallel Sysplex Resource Sharing 97
3.2.2 Description 97
3.2.3 Considerations 98
3.2.4 Use case: planned outage 98
3.2.5 Use case: unplanned outage 99
3.2.6 Resiliency optimization and next steps 100
3.3 Model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS 101
3.3.1 Description 101
3.3.2 Considerations 101
3.3.3 Multi-site workload configuration 103
3.3.4 Use case: planned outage 104
3.3.5 Use case: unplanned outage 104
3.3.6 Resiliency optimization and next steps 105
3.4 Model 4: Adding GDPS Continuous Availability for maximal resilience 106
3.4.1 Description 106
3.4.2 Considerations 108
3.4.3 Use case: planned outage 110
3.4.4 Use case: unplanned outage 112
3.4.5 Resilience optimization and next steps 114

Chapter 4. Planning for IBM Z resiliency 115
4.1 Considerations for the infrastructure layer 116
4.1.1 Conduct a component failure impact analysis 116
4.1.2 IBM Z platform 117
4.1.3 Coupling facility 118
4.1.4 Connecting IBM Z to storage servers 120
4.1.5 Connecting IBM Z to the network 120
4.1.6 Cryptography features/RoCE features/zEDC features 120
4.2 Considerations for the operating system layer 121
4.2.1 General considerations for Parallel Sysplex environments 121
4.2.2 Sysplex Failure Management (SFM) Policy 122
4.2.3 Automatic Restart Manager (ARM) Policy 123
4.2.4 IBM Health Checker for z/OS 124
4.2.5 Placement of data sets 124
4.2.6 JES2 emergency subsystem 124
4.2.7 Predictive Failure Analysis 125
4.2.8 Runtime Diagnostics 126
4.2.9 Asynchronous Duplexed CF Lock Structures 126
4.2.10 z/VM Single System Image (SSI) and Live Guest Relocation (LGR) 127
4.3 Considerations for the middleware layer 127
4.3.1 IBM MQ for z/OS 127
4.3.2 IBM Db2 for z/OS 128

4.4 Considerations for the applications layer 131
4.4.1 CICS Transaction Server 131
4.4.2 IMS 134
4.4.3 WebSphere Application Server for z/OS. 136
4.4.4 User-written application. 136
4.4.5 Batch processing 137
4.5 Considerations for the management layer 147
4.5.1 IBM Z Operations Analytics. 148
4.5.2 Data backup and replication 148
4.5.3 GDPS removes SPOF of people. 149
4.5.4 A word about cryptographic keys 150

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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 grant you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, MD-NC119, Armonk, NY 10504-1785, US

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 jurisdictions 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 provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

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.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

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 actual people or business enterprises 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. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

© Copyright IBM Corp. 2020. All rights reserved.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

CICS®
CICSPlex®
DB2®
Db2®
DS8000®
FICON®
FlashCopy®
GDPS®
HyperSwap®
IBM®
IBM Z®
IBM z15™
InfoSphere®
Interconnect®
NetView®
OMEGAMON®
Parallel Sysplex®
RACF®
Redbooks®
Redbooks (logo) ®
Resilient®
Resource Link®
System Storage™
System z®
Tivoli®
VTAM®
WebSphere®
z/Architecture®
z/OS®
z/VM®
z/VSE®
z15™

The following terms are trademarks of other companies:

ITIL is a Registered Trade Mark of AXELOS Limited.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Windows, and the Windows logo are trademarks of Microsoft Corporation 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.

Red Hat, are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

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.

Preface

Resiliency is the ability to provide the required capability in the face of adversity, without significant impact. This is not something that just happens, but rather must be thoroughly planned for and tested, including:

- Keeping hardware, the operating system, middleware, and applications up and running throughout planned and unplanned outages
- Recovering a site from an unplanned event without data loss

This IBM Redbooks publication gives a broad understanding of resiliency on the IBM Z platform and explains how it works and why it is important. It describes the technology and built-in resiliency capabilities of the IBM Z platform at various hardware and software levels, and discusses what to consider when pursuing higher availability.

This publication is intended for IT managers, IT architects, system programmers, and system operations professionals.

Authors

This book was produced by a team of specialists from around the world working at IBM Redbooks, Poughkeepsie Center, New York.

Bill White is a Project Leader and Senior Infrastructure Specialist at IBM® Redbooks®, Poughkeepsie Center.

Marc Coq is a Senior Electrical Engineer at IBM in IBM Z® RAS Design team. He joined IBM in 1990 in Systems Test. Marc holds an MSISE degree from Polytechnic University of New York. His expertise lies in Reliability Engineering, RAS Design, Resiliency, Procurement Engineering, Supplier Development, Power Supply Engineering, New Product Development & Introduction, Process Engineering, Product End of Life Engineering Management, Quality Engineering, Technology Qualification, Electronics Packaging Qualification, and Qualification Engineering. Marc is an IBM Master Inventor, a member of the IBM Academy of Technology, and an IEEE Senior Member.

Pabitra Mukhopadhyay is an Infrastructure Architect and IBM Certified Specialist in IBM India. He has 13 years of experience in the field of IBM Z. He holds a B.Tech degree in Electronics and Communication Engineering from West Bengal University of Technology, WB, India. He has worked at IBM for 7 years. His areas of expertise include IBM z/OS®, IBM Z hardware, Parallel Sysplex®, Middleware, Disaster Recovery, and process automation. He has written extensively on z/OS and middleware products in *IBM Systems* magazine and IBM Developer.

Carlos Alberto Pistillo is a z/OS System Programmer in Brazil. He has 13 years of experience in Operations, Scheduling, and z/OS System Support fields. He holds a degree in System Analysis and Development from IBTA/Metrocamp. His areas of expertise include z/OS Hardware Configuration, IBM Z Cloud migrations, Disaster Recovery planning, and execution with resilience, Parallel Sysplex, as well as IBM and ISV products support.

David Raften is an Advisory Programmer in Poughkeepsie, NY, USA. He graduated from the University at Albany with a degree in Computer Science. He has 37 years in IBM and has

© Copyright IBM Corp. 2020. All rights reserved.

worked with IBM Z hardware, z/OS, Parallel Sysplex, GDPS, and business resiliency. He has written extensively on Parallel Sysplex.

Mai Zeng is a Senior Software Engineer in IBM Systems Laboratory, China. He graduated from University of Electronic Science and Technology of China (UESTC) with a Masters degree in Computer Science. Mai has been working in the field with IBM mainframes since 2008 in the areas of z/OS, IBM Db2® for z/OS, Qrep, IBM GDPS®, and business resiliency. For the last several years, Mai has successfully helped major enterprises worldwide to achieve Continuous Availability as a solution consultant.

A special thanks to the following people for their contributions to this project:

Robert Abrams, Luiz Alves, Rich Brown, Betty Patterson Bucci, Anthony Ciabattoni, Tracy Dean, Desmond Fitzpatrick, David Follis, Mike Gonzales, Steve Hobson, Matthew Leming, David Petersen, Jacob Snyder, Matthew Sunley, Dave Surman, Gregory Vance, Phil Wakelin, Marna Walle, Stephan Wiedemer

Matt Lesher
Content Specialist, IBM Redbooks

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 IBM Redbooks 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`

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  `ibm.com/redbooks`

- Send your comments in an email to:

  `redbooks@us.ibm.com`

- Mail your comments to:

  IBM Corporation, IBM Redbooks
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

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

1

Resiliency is key to the survival of a digital business

As companies become digital businesses, they quickly realize that IT resiliency is no longer a “nice to have,” but rather a fundamental business requirement.

The cost of a service outage, resulting in missed business opportunities and stalled productivity, can be enormous. Even a short outage can cost millions of dollars.

The impact of an outage varies greatly depending on the business. Some might be able to survive an outage of a few hours or a day, while for others, even getting poor performance for several seconds can impact the bottom line.

IT resiliency is not just to ensure that the system is up and running. Instead resiliency applies to the entire IT infrastructure stack (compute, operating system, middleware, application, network, and storage layers) should consistently be available to make certain that services and data are accessible. In addition, transactions ought to be completed successfully and on time, with good performance. And all regulatory compliance and security requirements should be met.

Failures and cyberattacks[1] do occur and maintenance will eventually have to be applied to the IT infrastructure. The only question is; “What will occur and when?” Having immediately available backup through redundant IT components and automated recovery can greatly reduce the duration of outages. Without these measures, you must first identify and fix the problem, and then manually restart the affected hardware and software components.

Redundancy and the technologies that exploit it are perhaps the most prevalent mechanism to ensure resiliency, which is one of many key IBM Z[2] strengths. IBM designers have worked through the years to remove single points of failure in the Z platform. They have developed recovery routines in the firmware, operating systems, middleware, and applications to cope with slowdowns or to minimize outages and rapidly recover[3] from them.

[1] For IBM Z Cyber Resiliency see this web site: https://www.ibm.com/it-infrastructure/z/capabilities/enterprise-security

[2] The following IBM Z for IT Infrastructure Resiliency video provides an overview of how IBM Z protects infrastructures against all types of failures: https://www.youtube.com/watch?v=ZGZav2wL-Tk

© Copyright IBM Corp. 2020. All rights reserved.

1.1 What's the cost of downtime

Outages, whether *planned* [4] or *unplanned*[5], can have a significant impact on the business, resulting in lost revenue, reputation, and competitive position to name a few (see Figure 1-1).

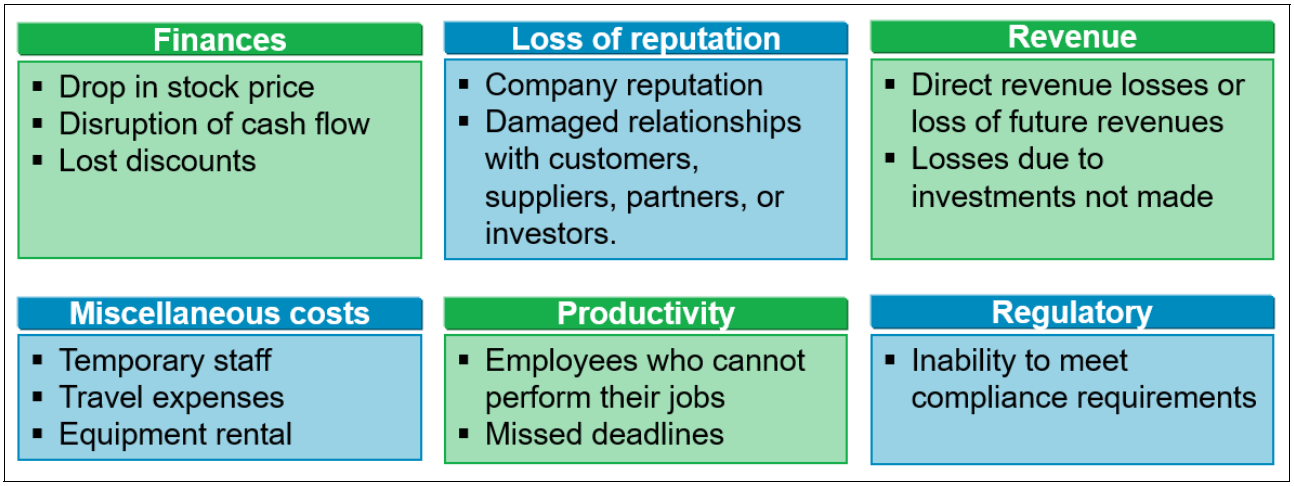


Figure 1-1 Cost of an outage

There are several variables that can reveal the impact of an outage for a given business. This involves evaluating:

- Fixed costs
- Lost revenue
- Intangible costs
- Balancing IT risk and costs of mitigation

Every Second Counts: Rapid Recovery for Finance video provides an example of the importance of rapid recovery:
https://www.youtube.com/watch?v=l1VtCf2mGMk&index=2&list=PLRg1AqGIzYLDhaVxbujKhKoNGVWTYo-xO

1.1.1 Fixed costs

Fixed costs are easiest to determine, but have the least long-term impact to a company. They can include the following factors:

- The costs of resources that are unproductive, such as the salary of the employees who are idle during the outage.
- The costs of catching up, making up for the downtime.
- The cost of managing and running the data center during the outage.
- The impact of missed internal deadlines.
- Financial penalties for missing external deadlines.

[3] IBM Z Instant Recovery is an offering on the IBM z15™. It minimizes the duration and impact of downtime and accelerates the recovery (see https://www.ibm.com/it-infrastructure/z/technologies/system-recovery).

[4] A system or its operating system, middleware, or application might need to be brought down for maintenance or an upgrade. This can require a partial or a full outage. In a high resiliency environment, this can be planned for and the application (service) availability is usually unaffected if enough spare capacity is readily available.

[5] Unplanned outages are by their nature unexpected and disruptive to business. The goal is to minimize this disruption. The best approach is to have duplicated resources and seamlessly switch to the backup resource during a disruption.

1.1.2 Lost revenue

While some transactions can be deferred, others cannot. Someone attempting an online purchase might wait until the system is up again or might just go to a competitor. There is also potential loss of new business, a new customer who might be a significant revenue source in the future.

The cost of a service outage can vary with duration of the outage, times of day or day of week. For example, just before a major public holiday versus during that public holiday.

The lost revenue will vary depending on which applications are affected. Calculating it can be done by looking at:

- Number of users affected by the outage
- Duration of the outage, from the end user's perspective
- Average number of transactions executed per hour, per user
- Estimated revenue per transaction

Table 1-1 shows examples of lost revenue based on cost per hour with percentage of availability per year. The calculation that is used here works as follows:

```
One year = 8766 hours
99% availability = 1% unavailable
8766 * 0.01 (1%) = 87.66 hours of outages per year
With lost revenue of $300,000 per hour (for example)
87.66 * $300,000 = $26,298,000 lost revenue per year with 99% availability
```

Table 1-1 Lost revenue example

Cost per hour	99%	99.9%	99.99%	99.999%	99.99999%
$100,000	$8,766,000	$876,600	$87,660	$8,766	$87
$500,000	$43,830,000	$4,383,000	$438,300	$43,830	$438
$1,000,000	$87,660,000	$8,766,000	$876,600	$87,660	877

1.1.3 Intangible costs

It is difficult to quantify the cost of losing brand image, customer confidence, and loyalty. These intangible costs become more important as more people use social media. If a customer has a bad experience, they might note this on line. And all their contacts are now aware of the outage, even persons who are not directly affected. Such issues can affect customer retention and in extreme cases, have an impact on the stock price.

To make things more complicated, the impact of an outage per minute changes with the length of the outage. A potential customer might try to run a transaction. If the system is down, the customer might just try again 10 minutes later. But if the system is still down the potential customer might go to a competitor. The next time that the customer becomes affected by the outage, you might lose that customer permanently.

Examples of intangible costs include:

- Lost business opportunities
- Loss of customer brand loyalty
- Loss of reputation
- Loss of market share
- Lack of confidence in the IT department

1.1.4 Balancing IT risk and costs of mitigation

Balancing IT risk and costs of mitigation is essential to achieving business goals. Achieving this balance continues to be a critical challenge in today's digital economy.

By balancing risk of a type of failure and cost of a solution, you can determine how much redundancy and availability management you can design into your IT infrastructure. By understanding the real cost of service *unavailability*, you can calculate the costs that you can sustain to provide this availability. These factors also determine when you can do maintenance and upgrades to the IT infrastructure.

You must evaluate the risk and impact of the outage and decide whether the design needs to be changed to meet the cost implications of service outages. Or maybe these cost implications should be adjusted.

Under-investment can result in "spending money by not-spending money," increasing risks to the business. Should an outage occur, you might be able to bear the cost of recovery at the time of the outage, but the cost of recovery can be greater than the cost of prevention.

Over-investment can result in excess solution costs. By over spending on unnecessarily high levels of protection, you can incur expenses and risks associated with lost opportunity.

Figure 1-2 provides an illustration of resilience optimization. Levels of resilience are shown on the bottom of the graph, with lower levels on the left (Costs from risk events) and higher levels on the right (Costs of mitigation solutions). The arrow in the center of the graph shows the optimum resilience/risk balance.

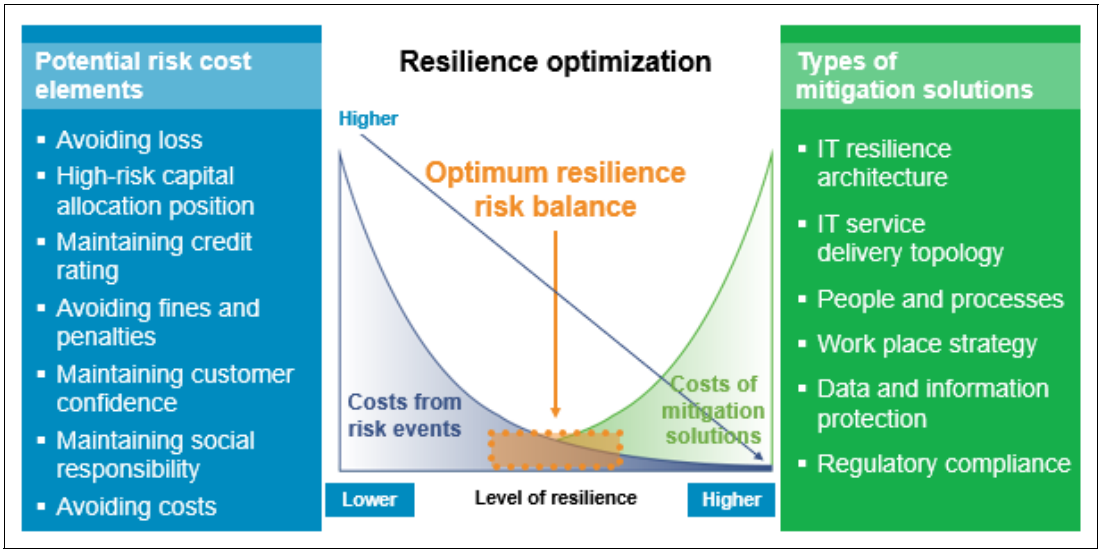


Figure 1-2 Requirement to balance risk and costs

1.2 Measuring availability

The first step in determining the impact of unavailability is to define what an outage means. What if transactions are running, but going slowly? What if a single application is getting error messages? What if this application is not a customer-facing application? What about a planned outage? Is a 1-minute downtime an outage? What if one system is down, but transactions can still run in a data sharing partner? What if the server is running smoothly, but there are problems in the network getting work to and from the server?

A simple way of looking at availability is to take a consumer point of view. Look at the number of transactions per second that successfully completed for each application. If for some reason this number unexpectedly drops, this condition can be considered an outage. The impact can be determined as a percentage of transactions that did not run. This metric has several advantages: it focuses on monitoring each application, it is objective (one can get a “percent available” number), and it allows you to analyze overall trends in availability.

Transaction slowdowns due to poor performance affect the total number of transactions that can run. Such issues can also be defined as an outage.

In addition to monitoring the availability percentage, you can also do a system outage analysis. Review each outage, identify the cause, and note the impact. The effect of each cause can then be plotted to identify where action is needed to improve availability. Some initial causes might arise in these areas:

- IT infrastructure design
- IT infrastructure component failures
- Poor documentation
- Human error

For selective outages, Root Cause Analysis will give better insight. Insights can include these reasons for outage:

- Better training needed.
- Need to apply known fixes more often.
- Better test / Quality Assurance processes needed.

After the analysis is complete, corrective action can be taken and monitored to ensure the effectiveness of that action.

1.2.1 Availability objectives

Service Level Agreements (SLAs) specify commitments to levels of service that are agreed on by the provider and the consumer. SLA commitments can be measured in a qualitative or quantitative way. SLA commitments can be associated with one or more escalations that specify the actions that are needed if the commitment is not met.

The main benefit of the SLA is that it documents the consumers' expectations of service quality, thus setting targets that the service provider must achieve. By monitoring the service delivery, the service provider can measure how well they are doing in the eyes of the consumer. Conversely, the consumer can also measure if the service provider is delivering on their commitments.

On the other hand, the SLA helps raise consumer awareness of cost of service and clearly defines the expected service quality. This clarity has both short-term and long-term benefits. The service provider can validate consumer complaints regarding poor service quality against the SLA, and new consumer requirements can be incorporated and priced.

SLAs can be applied to many types of records — including problem tickets and work orders — and can include target start, response, and delivery dates.

An SLA can specify the following kinds of information:

- Effective start dates, end dates, and review dates.
- Commitments that are related to the type of record that the SLA applies to. For example, for tickets you can specify target dates and times for response, resolution, delivery,

availability, and other values. For work orders, you typically specify target times for start, finish, and delivery time.

- The conditions under which a service level agreement applies, such as classification, services, and other criteria.
- Key performance indicators that track performance over time.

You must secure the consumer's formal agreement to any of the standards that are described in this document. Standards can include service requirements for application availability and response times. An agreement document might state the following, for example:

- Application ABC must be able to run at least 10,000 transactions/second with 95% finishing under 0.1-seconds response time as measured by application records.
- Application ABC must be available 99.999% of the time other than an 8-hour planned outage, one time per quarter. Warning of this planned outage must be given no later than 2 weeks before the outage. Availability level is based on the value that is indicated in the outage record and agreed to by all parties.

The SLA requirements determine how the IT infrastructure should be designed.

1.2.2 Recovery objectives: RPO and RTO

Recovery events are typically measured by the targets that are defined for Recovery Point Objective (RPO) and Recovery Time Objective (RTO) targets, as explained in Table 1-2.

Table 1-2 RPO and RTO targets

Recovery Point Objective (RPO)	Looking backward from the event, the RPO represents how much data is lost due to the site failure. This value can vary from zero with synchronous remote copy, to 24 hours or more with recovery from tape. Looking forward from the event, the RTO is the goal for how much time can pass before service is recovered after a site failure. It includes bringing up the operating system at the recovery site, the middleware, enabling the network, and bringing the applications (services) on line. This time span can be anywhere from minutes to days, depending on the recovery solution. To repeat, the RPO defines the data loss that can be incurred. If the requirement is 0, you need synchronous data replication. Keep in mind that you can combine a synchronous replication with asynchronous replication for three-site or four-site solutions. This approach can support a recovery site that is located an extended distance away. If the RPO is greater than 0, only asynchronous replication is needed. There are different ways to configure asynchronous replication to support different objectives. Keep in mind the cost of the data loss. For example, the difference between three seconds and 30 seconds of data loss is 27 seconds. How much would these 27 seconds of data loss cost the business? The answer to that question reveals how you must implement data replication.
Recovery Time Objective (RTO)	The RTO is not just the time that is needed to bring up a backup server. It also includes the time to recover the data, prepare the physical facilities, switch the network, bring up the operating system, databases, management tools, and finally the applications. In case of problems, there are dependencies on the infrastructure staff: Operations, Network staff, and System Programmers, and application programmer availability. Automation can help in greatly reducing the time that is needed to recover the environment by reducing the "Single Points of Failure" of support staff, while also optimizing the recovery process.

It seems obvious to say that you must choose the replication techniques that achieve the recovery objectives. However, business realities often complicate your choices, as in these examples:

- An existing building is owned at a remote location so it must be used, even though it is too far away to support synchronous replication to get RPO=0.
- Tape-tape replication is chosen because it is cheaper, even though the amount of potential data loss might cause business issues, in the event of a disaster.
- The SLA, RPO, and RTO have been defined when the running environment was different and are now outdated. Regular reviews and updates of existing SLAs, RPO, and RTO are needed to capture changes to the business environment.

SLAs (RPO and RTO objectives) should reflect business requirements and drive the technology that is needed to meet them, not the technology driving the business requirements.

1.2.3 Aspects of availability

IT resiliency is the ability to rapidly adapt and respond to any internal or external disruption, demand, or threat, and continue business operations without significant impact. IT resiliency is related to, but broader in scope, than disaster recovery. Disaster recovery concentrates solely on recovering from an unplanned event.

Business Continuity (BC) is the ability of an organization to ensure continuity of service and support for its customers and to maintain its viability before, after, and during an event. This means that a company must be able to maintain business processes, technology, and business functions.

BC has evolved to an all-encompassing term for most companies. The term can be applied to business functions and processes, and to storage and database capabilities. BC is a discipline that leverages people, process, and technology.

- People who understand their roles-and-responsibilities during an outage or disruption. They must design the strategies and solutions to enable the continuity of business operations. Operators must practice *recovery procedures* to ensure a smooth recovery after an outage or disruption.
- Processes include crisis management and business continuity, but can be expanded to include problem, change-and-configuration management (such as ITIL), and business controls.
- Technologies are often emphasized too much. However, they include backup, replication, mirroring, and failover solutions.

One method to identify IT components that have backup capability and the procedure to switch to them is to perform a component failure impact analysis (CFIA). A CFIA can shorten the time from when an incident is recognized to when services are restored. Through CFIA, you anticipate problems, modify or develop procedures to fix problems, and identify procedures for each type of failure.

For more information about CFIA, see “Conduct a component failure impact analysis” on page 116.

Business resiliency challenges

The challenges for achieving business resiliency include:

- Evaluating, planning, and mitigating the business impact of various types of risks.
- Responding to regulatory pressures and compliance associated with business resiliency.
- Ensuring data is protected, available, and accessible as needed by the business.
- Achieving availability objectives by reducing frequency and duration of infrastructure applications and data outages.
- Recovering from and responding to disruptive events.

Several terms are used in the discussion of availability. The key terms that are used in this document are high availability, continuous availability, and disaster recovery.

High availability

A high availability (HA) environment has a fault-tolerant, failure-resistant infrastructure that supports continuous application processing. The goal is to provide service during defined periods, at acceptable or agreed upon levels, and mask *unplanned outages* from end users. Many hardware and software products have redundant components and recovery routines to mask possible events. The extent of this support greatly varies.

HA refers to systems and applications that typically operate 24x7 and usually require built-in redundancy or rapid failover capabilities. These capabilities minimize the risk of downtime due to failures in the IT infrastructure.

HA is distinguished from disaster recovery (DR) primarily through the duration and scope of outage that the technology is designed to mitigate. Table 1-3 shows outage types with typical industry practices.

Table 1-3 Outages and mitigation

Outage examples	Approach to mitigation
Less than a day: Potentially less than 24 hours in length or which do not impact the entire business operation.	Handled through a company's high-availability architecture.
More than a day: Can exceed 24 hours and/or might impact the entire business operation.	DR declaration, formally defined as the activities and programs that are designed to return the entity to an acceptable condition following an event.

Continuous operations

A continuous operations (CO) environment enables non-disruptive backups and system maintenance that is coupled with continuous availability of applications. The goal is to mask *planned outages* from end-users. It employs non-disruptive hardware and software changes, non-disruptive configuration, and software coexistence. Live Guest Relocation, for example, enables CO because you can move work off one server before you bring that server down for maintenance.

Continuous availability

Continuous availability (CA) is a combination of CO and HA. It is designed to deliver non-disruptive service to the end user 7 days a week, 24 hours a day. There are no planned or unplanned outages. A Parallel Sysplex[6] is an example of a CA solution. It enables HA by removing single points of failure (SPOFs) within the hardware and software environment. It enables CO by allowing “rolling IPLs” to mask planned events.

[6] Parallel Sysplex is a clustering technology used on IBM Z platforms (with z/OS). Parallel Sysplex acts like a single system image by sharing data, computing, and connectivity resources to allow for continuous service delivery.

Disaster recovery

Disaster recovery (DR) is protection against unplanned outages through a reliable, predictable recovery process. Examples include inability to access storage or a site disaster such as a fire or loss-of-power event.

DR is the ability to respond to an interruption in services by implementing a disaster recovery plan to restore an organization's critical business functions. DR is considered to be more reactive than HA solutions. DR solutions typically enable recovery within 24 to 72 hours, while HA solutions enable recovery within 1 to 2 hours of an event. As with any event, the duration and scope of the event will dictate when use of the DR plan is necessary. A DR process might be an appropriate response, even with events that will last less than the 24-to-72-hour standard that was previously mentioned.

1.3 A path to higher availability

To assist you with improving the resiliency of your IBM Z environment, IBM has developed a series of deployment models for resiliency. Those models map to different levels of resiliency based on business requirements. The models can guide you through assessment of your current resiliency investment and selection of new targets for business workloads. The models then help you to identify practical steps to achieve availability goals for your IT infrastructure. In addition, the models suggest disaster recovery considerations between data centers as part of your assessment. Table 1-4 provides an overview of the four IBM Z resiliency deployment models.

Table 1-4 The four IBM Z deployment models for resiliency

Model #	Model name	Risk/impact to business	Hours to recovery	Typical RTO*
1	IBM Resilient® / Reliable base	High	8+	3+ days
2	Failover	Medium	2+	8 hours
3	Fault Tolerant plus GDPS Metro or GDPS Global	Low	Minutes	< 1 hour
4	Fault Tolerant plus GDPS Continuous Availability	Minimum	Minutes	< 2 minutes
* Recovery Time Objective (RTO) is the target time for system restoration. Not to be confused with Recovery Point Objective (RPO), the data loss that the enterprise can accept. (RPO could be ZDL, zero data loss.)				

Each of the four models supports different capabilities of the hardware and software technologies that are represented by the layers that are listed in Table 1-5. The table also emphasizes that the layers relate to each of the four resiliency deployment models.

Table 1-5 Each of the four models relates to all layers

Layer	Components examples of each layer	Relationship to the four models *
Infrastructure	Compute, storage, network, virtualization	► Model 1: Starting with a resilient and reliable base ► Model 2: Reducing the duration of outages with failover capability ► Model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS ► Model 4: Adding GDPS Continuous Availability for maximal resilience
Operating System	z/OS, z/VM, Linux on Z	
Middleware	IBM Db2, IMS/DB, IBM MQ	
Application	IBM CICS®, IMS/TM, WebSphere® Application Server	
Management	Automation, monitoring, and reporting tools; data replication	
* The question is not *whether* each of the four deployment models relates to each infrastructure layer. *All models **do** have a relationship to aspects of all the layers*.		

In the relationship of the layers to the four resiliency deployment models, the question is this: *What are the essential features of the layers for each of the four models?*

Here are some example issues regarding layers in relation to the four models:

- **Model 1 to Model 2:** In the transition from resiliency model 1 to 2, the infrastructure layer of your system typically grows from a single system to a multi-system environment.
- **Reference architectures:** Various IBM solutions can be a part of your implementation of a model. Each solution has a 'reference architecture,' which can include hardware components and software, such as middleware and applications.

For descriptions of the capabilities for each layer, see Chapter 2, "Resiliency is built into the IBM Z platform" on page 15.

1.3.1 Model 1: Starting with a resilient and reliable base

It would be best to never have failures. You can get closer to this ideal by starting with quality base technology together with regular maintenance. Application code that you write on top of that base should have extensive problem detection and near-instantaneous correction. You take these steps with IBM Z hardware and the operating system.

By design, the IBM Z platform is a highly resilient system. The architecture of IBM Z platforms has built-in self-detection, error correction, and redundancy. This architecture reduces single points of failure, to deliver the best reliability of any enterprise system in the industry. Transparent processor sparing and dynamic memory sparing enable concurrent maintenance and seamless scaling without downtime. Business workloads that fail can be restarted in place. The operating system has code to seamlessly recover from failures and even predict failures before they happen.

If the operating system or logical partition (LPAR)[7] is down for a planned or unplanned outage, another instance of the workload (in another LPAR) can absorb the work. An automatic recovery process can reduce the workload impact by avoiding the delays associated with a manual intervention.

[7] A logical partition (LPAR) is a subset of the IBM Z hardware that is defined to support an operating system. An LPAR contains resources (such as processing units, memory, and input/output interfaces) and operates as an independent system. Multiple LPARs can exist in an IBM Z platform. See "Virtualization" on page 25 for more information.

Resiliency for a single Z platform can be enhanced by creating a Parallel Sysplex environment with data sharing across two or more z/OS images. Although this does not provide for additional hardware redundancy, it can keep middleware and application workloads available and accessible across software upgrades by allowing rolling restarts.

The following configuration of architectural layers has built-in redundancy with fault tolerance within the base infrastructure components that also allow you to ramp up resiliency as required.

- **Infrastructure** (i.e., compute, storage, network, virtualization)
  A single IBM Z platform, one copy of the data, one path for the network to access the applications.
- **Operating System** (i.e., z/OS, IBM z/VM®, Linux on Z)
  Each application runs on a single copy of the operating system.
- **Middleware** (i.e., Db2, IMS/DB, IBM MQ)
  Each application runs on a single copy of a database server, messaging server, or both.
- **Application** (i.e., CICS, IMS/TM, IBM WebSphere Application Server)
  Although there is a single LPAR, often there are multiple application server instances. In a simple model, each is dedicated to a particular application. There can also be redundancy where multiple application servers are available for a particular application.
- **Management** (i.e., monitoring and reporting tools)
  Linear effort as the environment grows.

For more information on this model, see 3.1, “Model 1: Starting with a resilient and reliable base” on page 94.

1.3.2 Model 2: Reducing the duration of outages with failover capability

Despite the best base technology, failures can still occur. You should anticipate failures with the right technology implementation. *Failover*, the ability to restart work quickly on a backup component, can significantly reduce the impact of an outage.

With the installation of a second IBM Z platform in the IT infrastructure, your component resource sharing begins the road to full system redundancy. Additionally, you can mitigate single points of failure for data by setting up “Metro” (synchronous) data mirror, and Global (asynchronous) data mirroring for longer distances. And with IBM Copy Services Manager (CSM), you enable fast data replication with little or no data loss.

A *failover configuration* eliminates the SPOFs present in a single IBM Z platform. However, you must configure the system to use this to fail over seamlessly.

- **Infrastructure** (i.e., compute, storage, network, virtualization)
  Multiple copies of the base infrastructure exist, but failover is a manual process with a cold standby. If there is a failure, recovery is operations intensive, but can be done. Full storage replication. Best RPO with synchronous remote copy.
- **Operating System** (i.e., z/OS, z/VM, Linux on z)
  Each application runs on a single copy of the operating system.
- **Middleware** (i.e., Db2, IMS/DB, IBM MQ)
  Each application runs on a single copy of a database server, messaging server, or both.
- **Application** (i.e., CICS, IMS/TM, WebSphere Application Server)
  Redundancy for application servers for a particular application. An example can be multiple CICS AORs or IMS MPRs within a single LPAR.

- **Management** (i.e., automation, monitoring and reporting tools, data replication)
  Ability to manage details of failover, management of the central processor complex (CPC), site recover actions, and more.

For more information on this model, see 3.2, “Model 2: Reducing the duration of outages with failover capability” on page 97.

1.3.3 Model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS

Even the best fault tolerant designs can fail. When failure happens and service is impacted, be prepared to restore service quickly.

One method to reduce the impact of an outage is to avoid outages whenever possible. Anticipate failures with the right technology implementation. Designing a robust Fault Tolerance architecture can help prevent failures from causing a disruption of service by removing SPOFs. For example, you can use an IBM Parallel Sysplex with z/OS. Most hardware and software components in the IBM Z environment can be cloned on to a second CPC with dynamic workload balancing that spreads the work. For planned or unplanned events on one LPAR, the workload flows to another LPAR seamlessly, without affecting availability.

Storage is a single point of failure that is not covered by a z/OS Parallel Sysplex. By adding GDPS to the disk replication, you can automate management of CPC and site recovery actions. In failover situations, automation can reduce the business impact by minutes or even hours.

You can combine Metro Mirror and Global Mirror solutions into 3- and 4-site solutions to provide IBM HyperSwap® for very rapid recovery for disk outages. You also enable zero data loss across long distances, without affecting user response time.

A *fault-tolerant configuration* with z/OS Parallel Sysplex data sharing allows seamless failover. You can configure disk remote copy so that almost all SPOFs are removed within a data center. This failover configuration includes the LPAR, operating system, communication server, middleware, and the application server. IBM GDPS for disaster recovery addresses events that could bring down an entire site. GDPS functionality covers these two areas:

- (Synchronous) GDPS Metro provides the highest level of availability possible. It enables HyperSwap, for near-continuous availability for disk problems, and a multi-site workload to allow near-continuous availability for many types of site disasters.
- (Asynchronous) GDPS Global provides down to 3 seconds of RPO for long-distance DR.

Here are additional benefits that GDPS brings to the basic configuration layers of a system:

- **Infrastructure** (i.e., compute, storage, network, virtualization)
  Multiple copies of the base infrastructure exist. Failover is seamless. Disaster recovery can be down to RTO=0 with GDPS Metro, or <1 hr with GDPS Global.
- **Operating System** (i.e., z/OS, z/VM, Linux on z)
  Each application can run on multiple copies of the operating system simultaneously. Failover is seamless.
- **Middleware** (i.e., IBM Db2, IMS/DB, IBM MQ)
  Multiple database and messaging servers with full read/write access to the data. Each backs up the other. Failover is seamless.
- **Application** (i.e., IBM CICS, IMS/TM, WebSphere Application Server)
  Redundancy for application servers for a particular application. An example can be

multiple CICS AORs or IMS MPRs within a single LPAR as well as across LPARs. Failover is seamless.

- **Management** (i.e., automation, monitoring and reporting tools, data replication.)
  Ability to manage the entire configuration from a single point of control. Dynamic workload balancing between LPARs. End-to-end automation for disaster recovery to manage the remote copy environment, recovery actions, operating system environment, and protection from making operation errors.

For more information on this model, see 3.3, “Model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS” on page 101.

1.3.4 Model 4: Adding GDPS Continuous Availability for maximal resilience

The last of the four IBM Z resiliency deployment models is as follows:
Fault Tolerant within a data center plus GDPS Continuous Availability

When you have Fault Tolerance within the Primary Data Center, plus GDPS Continuous Availability for key workloads, Disaster Recovery looks like this architecturally:

- GDPS Continuous Availability consists of two sites, which are separated by virtually unlimited distances.
- The sites run the same applications with the same data sources, to provide cross-site workload balancing, continuous availability, and Disaster Recovery (DR).
- As a result, workloads to fail over to another sysplex for planned or unplanned workload outages within seconds.

GDPS Continuous Availability allows recovery time in 1 - 2 minutes with 3 - 5 seconds of data loss, also with full end-to-end automation.

- **Infrastructure** (i.e., compute, storage, network, virtualization)
  Multiple Parallel Sysplexes across two sites. Supports RTO of 1 - 2 minutes, and RPO of down to 3 seconds. Data replication done by the IBM InfoSphere® Data Replicator software.
- **Operating System** (i.e., z/OS, IBM z/VM, Linux on z)
  Multiple Parallel Sysplexes across two sites.
- **Middleware** (i.e., Db2, IMS/DB, IBM MQ)
  Multiple database servers with full read/write access to the data. Each backs up the other. The message servers would be in a queue sharing group in the primary data center, using asynchronously replication to the DR or secondary data center.
- **Application** (i.e., CICS, IMS/TM, IBM WebSphere Application Server)
  All update transactions for an application run on the Active site, although read-only transactions can run on the Standby site. This approach helps with workload balancing.
- **Management** (i.e., automation, monitoring and reporting tools, data replication.)
  Ability to manage the entire configuration from a single point of control. End-to-end automation for disaster recovery to manage the remote copy environment, recovery actions, operating system environment, and protection from making operation errors. Actions are performed at the application level or for the entire workload.

For more information on this model, see 3.4, “Model 4: Adding GDPS Continuous Availability for maximal resilience” on page 106.

2

Resiliency is built into the IBM Z platform

High availability cannot be achieved without a resilient IT infrastructure. Every critical component of the IT infrastructure must have resiliency that is built in to avoid single points of failure. IBM Z leads the industry in IT resiliency by providing technologies that avoid downtime, ensure access to critical applications, middleware, and data, and maintain productivity of users.

The IBM Z platform is known for its reliability, availability, and serviceability (RAS) capabilities. RAS is built into the hardware and software stacks of the IBM z/Architecture, where mean time between failures is measured in decades, making application availability of 99.999% possible.

A Parallel Sysplex expands on the Z platform's RAS design. It is a clustering technology for logical and physical systems that allows highly reliable, redundant, and robust Z environments. A properly configured Parallel Sysplex can achieve near-continuous availability with tight cooperation of the hardware and software stacks.

Geographically Dispersed Parallel Sysplex (GDPS) offerings use a combination of Parallel Sysplex technologies, data replication technologies, and automation to reach continuous availability of up to 99.99999%[1], while also providing disaster-recovery capabilities.

Availability can be improved by establishing base requirements, such as:

- Building restartability into system components
- Keeping restarts to a minimum
- Freeing up serialization when recovering from a failure
- Checking that all parameter changes can be performed dynamically
- Avoiding operator intervention
- Verifying component availability
- Avoiding single points of failure

[1] IBM internal data based on measurements and projections was used in calculating the expected value. Necessary components include z15 platforms with z/OS V2R3 or later that run in a Parallel Sysplex with resiliency technology enabled, such as System Managed CF Structure Duplexing, Sysplex failure management, and Capacity Provisioning Manager. Other configurations might provide different availability characteristics.

© Copyright IBM Corp. 2020. All rights reserved.

In this chapter, we provide an overview of the Z resiliency technologies and capabilities using the layers introduced in “A path to higher availability” on page 9.

IBM Z RAS and the layers that represent the different Z resiliency technologies and capabilities are described in the following sections:

- 2.1, “Why redundancy and RAS” on page 16
- 2.2, “Infrastructure layer” on page 17
- 2.3, “Operating system layer” on page 31
- 2.4, “Middleware layer” on page 39
- 2.5, “Application layer” on page 45
- 2.6, “Management layer” on page 81

The layers described in this chapter apply to all aspects of the Z resiliency deployment models described in Chapter 3, “IBM Z resiliency models” on page 93.

2.1 Why redundancy and RAS

One of the most basic requirements for continuity of service is that the components (hardware and software) in the IT infrastructure are operating in a resilient environment. This means, if you want to attain the highest levels of availability for your business-critical applications and data, you must start with sound fundamentals. These fundamentals include ensuring that critical IT components have available backup (redundant) capacity, redundant power sources, and redundant connections across critical paths to storage, networks, and other systems, as well as multiple instances of software (operating systems, middleware, and applications).

Note: Redundancy, by itself, does not necessarily provide higher availability. It is essential to design and implement your IT infrastructure by using technologies such as system automation, Parallel Sysplex, and GDPS. These technologies can take advantage of the redundancy and respond to failures with minimal impact on application availability.

From a redundancy and resiliency perspective, the IBM Z platform (hardware and software) design includes reliability, availability, and serviceability (RAS) principles that are driven by a set of high-level program objectives that move toward continuous reliable operation (CRO) at the system level. The key objectives of IBM Z RAS are to ensure data integrity, computational integrity, reduce or eliminate planned and unplanned outages, and reduce the number of repair actions.

The RAS strategy is to manage change by learning from previous generations of IBM Z and investing in new RAS functions to eliminate or minimize all sources of outages. The RAS strategy employs a building-block approach that is designed to meet stringent requirements for achieving CRO. These are the RAS building blocks:

- Error prevention
- Error detection
- Recovery
- Problem determination
- Service structure
- Change management
- Measurement
- Analysis

2.2 Infrastructure layer

Enhancements to IBM Z current RAS designs are implemented in the next Z platform through the introduction of new technology, structure, and requirements. Continuous improvements in RAS are associated with new features and functions to ensure that the IBM Z platforms deliver exceptional resiliency.

IBM Z RAS is accomplished with concurrent replace, repair, and upgrade functions for processing units, memory, CPC and I/O drawers, as well as I/O features for storage, network, and clustering connectivity. IBM Z RAS also extends to the non-disruptive capability for installing firmware (known as LIC, Licensed Internal Code), updates. In most cases, a capacity upgrade can be concurrent, without a system outage.

The IBM Z hardware and firmware are a physical implementation of the z/Architecture. The key capabilities that are featured in the infrastructure layer include compute, storage, network, and virtualization, as shown Figure 2-1 on page 17.

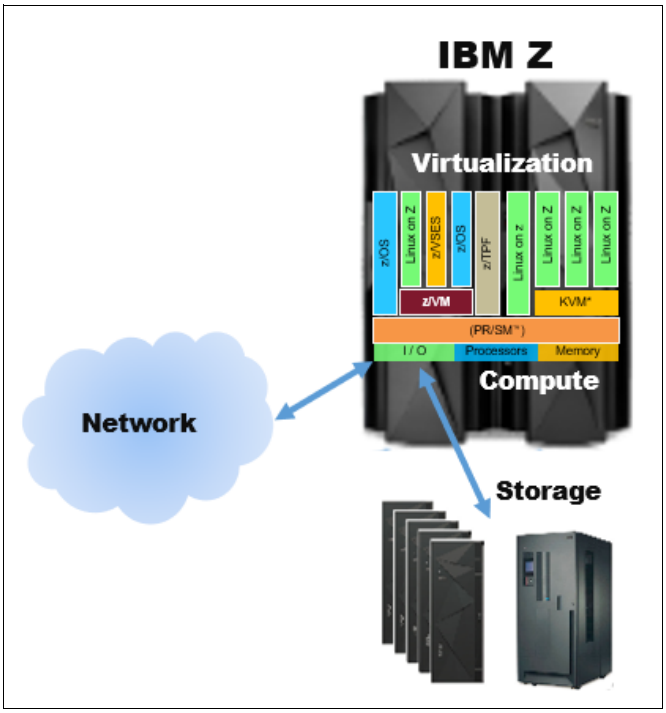


Figure 2-1 Key elements of the infrastructure layer

2.2.1 Compute

The IBM Z hardware platform can have one or more frames based on the model. The frames contain the following components:

- CPC drawers (processing units (PUs), memory, and connectivity to I/O drawers)
- I/O drawers for I/O features and special purpose features, such as encryption
- Cooling units for either air or water cooling
- Power supplies
- Oscillator cards for system clocking

The IBM Z model will depend on the number of processing units (PUs), the amount of memory, and how much I/O bandwidth you require to run your workloads. The PUs, memory,

and I/O features have built in resiliency and the power, cooling, and system clocking have redundant components.

CPC drawer

Each CPC drawer in an IBM Z houses the processing units, memory, and I/O interconnects. The CPC drawer design aims to reduce, or in some cases even eliminate, planned, and unplanned outages. The design does so by offering concurrent repair, replace, and upgrade functions for the CPC drawer.

The process through which a CPC drawer can take over for a failed CPC drawer is called Enhanced (CPC) drawer availability (EDA). EDA allows a single CPC drawer in a multi-drawer configuration to be removed and reinstalled concurrently for an upgrade or a repair.

Processing units

All PUs in the IBM Z are physically the same, but differentiated by their characteristics. The PUs can be characterized in advance or dynamically. Certain PU characterizations are better-suited to specific types of tasks than others. The PU characterization types are as follows:

- Central processor (CP) is used for standard processor with any IBM Z operating system and user applications.
- Integrated Facility for Linux (IFL) processor is used with Linux on Z and for running the IBM z/VM hypervisor in support of Linux. z/VM is often used to host multiple Linux virtual machines (known as guests).
- z Integrated Information Processor (zIIP) is used under IBM z/OS for designated workloads, which include IBM Java virtual machine (JVM), various XML System Services, IPSec offload, certain parts of IBM Db2 DRDA, star schema, IBM HiperSockets for large messages, and the IBM GBS Scalable Architecture for Financial Reporting.
- Internal Coupling Facility (ICF) is used for z/OS clustering. They are dedicated to this function and exclusively run the Coupling Facility Control Code (CFCC).
- System assist processor (SAP) is used for offload and manage I/O operations. Several SAPs are standard with the Z platform. More SAPs can be configured if increased I/O processing capacity is needed.
- Integrated Firmware Processor (IFP) is used as single PU that is dedicated to supporting native PCIe features (for example, RoCE Express, zEDC Express, HyperLink Express, and Coupling Express).

In the unlikely event of a permanent core failure, each core can be individually replaced by one of the available spares. Core sparing is transparent to the operating system and applications. The resiliency capabilities for the PUs include:

- Transparent PU sparing
- Concurrent processor drawer repair/add including Processor/Cache Chips and other internal components
- Various dynamic “capacity on demand” details
- Non-disruptive capacity backup
- Transparent SAP sparing
- Dynamic SAP reassignment

Memory

The IBM Z redundant array of independent memory (RAIM) design detects and recovers from failures of dynamic random access memory (DRAM), sockets, memory channels, or dual inline memory module (DIMM). IBM Z memory includes these resiliency capabilities:

- DIMM-level failure protection based on RAIM technology
- Memory channel and bus protection based on CRC and RAIM technology
- Concurrent memory repair/add through the concurrent drawer repair process
- Concurrent memory upgrade

PCIe fanout

The PCIe fanout in the CPC drawer provides the redundant paths for data between memory and the I/O drawers, which house the I/O features. The PCIe fanout is hot-pluggable. If a PCIe fanout fails, a redundant I/O interconnect allows a PCIe fanout to be concurrently repaired without loss of access to its associated I/O domains within the I/O drawer.

I/O drawer

The I/O drawer supports multiple features that are organized in four hardware domains per I/O drawer. Each domain is driven through an IBM Interconnect® PCIe switch adapter. The two PCIe switch cards provide a backup path for each other through the passive connection in the I/O drawer backplane. During a PCIe fanout or cable failure, all I/O features in the two domains can be driven through a single PCIe switch card.

I/O features are supported in any combination and can be concurrently added and removed. The resiliency capabilities for I/O include:

- Multiple channel path support
- Concurrent repair/add of all features in an I/O drawer
- Concurrent repair/add of I/O drawer
- Concurrent upgrade of any I/O feature type
- Domain failover based on Redundant I/O interconnect
- Dynamic activation of I/O configuration changes

System clocking

IBM Z has two oscillator cards (OSCs) for system clocking purposes: one primary and one secondary. If the primary OSC fails, the secondary detects the failure, takes over transparently, and continues to provide the clock signal to the system.

Power

The resiliency capabilities for power include transparent failover and concurrent repair of all power parts and redundant AC inputs. The power supplies for IBM Z are also based on the N+1 design. The additional power supply can maintain operations and avoid an unplanned outage of the system.

Cooling

IBM Z can provide N+1 cooling function for the radiator-based, air cooled model, which suits the needs of typical business computing. The N+1 (redundant) cooling function for the water-cooled model suits the needs of enterprise computing. The resiliency capabilities for cooling include transparent failover and concurrent repair of cooling pumps, blowers, fans, and so on.

System control structure

The system control structure includes redundant sideband control access to all units in the Z platform, and redundant network switches. The Support Elements (SEs) are connected to support processors in the CPC drawer, I/O drawers, power supplies, and cooling units. A Hardware Management Console (HMC) connects to the SEs.

The HMC and SE are stand-alone computers that run a set of Z platform management applications. The HMC and SEs are closed systems, which means that no other applications

can be installed on them. When tasks are performed at the HMC, the commands are routed to the primary SE of the Z platform. The primary SE then issues those commands to the appropriate support processor(s).

Two rack-mounted SEs (one is the primary and the other is the alternate), and at least one HMC are needed to manage the Z platform. With two HMCs, you can enable the automatic switchover function and perform the alternate HMC preload function from the primary HMC.

The HMC and SEs include N+1 redundant power supplies. Information is mirrored once per day between the primary and the alternative SEs.

IBM Z firmware

The IBM Z firmware provides the flexibility to update dynamically the configuration. You can perform the following tasks dynamically:

- Add a logical partition (LPAR)
- Add a logical channel subsystem (LCSS)
- Add a subchannel set
- Add a logical CP to an LPAR
- Add a cryptographic coprocessor
- Remove a cryptographic coprocessor
- Enable I/O connections
- Swap processor types
- Add memory
- Add a physical processor

To help minimize planned outages, the following tasks also are possible:

- Concurrent driver upgrades
- Concurrent and flexible customer-initiated upgrades
- Concurrent firmware (patches)

Capacity on Demand offerings

The Capacity on Demand offerings provide permanent and temporary upgrades by activating one or more IBM Z Licensed Internal Code Configuration Control (LICCC) records. These upgrades occur without disruption to the operation of the server. Depending on the type of upgrade, you can order upgrades yourself by using the customer-initiated upgrade (CIU) application on IBM Resource Link®. Or you can call your IBM sales representative to order the upgrades.

- **Permanent upgrades:** You can order permanent upgrades through your IBM sales representative or by using the CIU application Resource Link. However, what you can upgrade differs between the two methods.
  - **Through your IBM sales representative:** Ordering a permanent upgrade through your IBM sales representative (also referred to as Capacity Upgrade on Demand (CUoD)) allows you to:
    - Add model capacity
    - Add specialty engines
    - Add memory
    - Activate unassigned model capacity or IFLs
    - Deactivate activated model capacity or IFLs
    - Activate channels
    - Activate crypto
    - Change specialty engine (recharacterization)

- **Using the CIU application through Resource Link:** Ordering a permanent upgrade by using the CIU application through Resource Link allows you to add capacity that will fit within your existing hardware:
  - Add model capacity
  - Add specialty engines
  - Add memory
  - Activate unassigned model capacity or IFLs
  - Deactivate activated model capacity or IFLs.
  - Temporary upgrades

- **Other types of upgrades:** Three types of upgrades are provided through the following offerings:
  - **On/Off Capacity on Demand (On/Off CoD):** This offering allows you to temporarily add additional capacity or specialty engines due to seasonal activities, period-end requirements, peaks in workload, or application testing.
  - **Capacity Backup (CBU):** This offering allows you to replace model capacity or specialty engines to a backup server in the event of an unforeseen loss of server capacity because of an emergency
  - **Capacity for Planned Events (CPE):** This offering allows you to replace model capacity or specialty engines due to a relocation of workload during system migrations or a data center move.

You can order CPE or CBU temporary upgrade records or related entitlements, such as CBU tests, by using the CIU application through Resource Link or by calling your IBM sales representative.

You can order an On/Off CoD temporary upgrade only by using the CIU application through ResourceLink.

IBM Z System Recovery Boost

System Recovery Boost[2] is designed to reduce downtime from both planned and unplanned events with no additional IBM software licensing costs. System Recovery Boost provides additional Central Processor (CP) capacity during a temporary performance increase, which is known as the boost period. For planned events, the boost period accelerates system shutdown processing. For any event, planned or unplanned, it accelerates the system initial program load (IPL), the middleware and workload restart, the system recovery, and the workload execution that follows. System Recovery Boost can be used twice per IPL on z/OS; once for shut-down and once for start-up. This temporary performance increase results from three primary functions:

- **Speed Boost:** Enables CPs on subcapacity systems to run at full-capacity speed in the image(s) being boosted. This function is supported by z/OS, z/TPF, and z/VM,
- **zIIP Boost:** Enables general-purpose workloads to run on zIIP processors in the image(s) being boosted. This function is supported by z/OS only.
- **GDPS Enhancements:** Increases the speed at which GDPS drives hardware actions, along with enhancing the speed of the underlying hardware services. This function is supported by z/OS, z/TPF, and z/VM.

System Recovery Boost is operating-system dependent. At the time of writing this publication, the main System Recovery Boost exploiters are z/OS (running in an LPAR) and z/TPF. z/VM will make use of System Recovery Boost if it runs on CPs only (IFLs are always at their full clock speed). Second-level z/VM guest operating systems are able to inherit the

[2] IBM Z System Recovery Boost is supported on IBM z15 and later platforms.

boost if they are running on CPs. z/OS configured as a guest system under z/VM management will not exploit the boost.

System Recovery Boost can be used either during LPAR IPL or LPAR shutdown to make the running operating system and services available in a shorter period of time.

For more information, see *Introducing IBM Z System Recovery Boost*, REDP-5563 here:

http://www.redbooks.ibm.com/abstracts/redp5563.html?Open

2.2.2 Storage

IBM has a wide range of storage product offerings that are based on open standards and share a common set of tools, interfaces, and innovative features. The IBM System Storage™ (DS8000® series) is designed as a high performance, high capacity, and resilient series of disk storage systems. It offers high availability through redundant components, multi-platform support, and simplified management tools to help provide a cost-effective path to exceptional resilience.

With z/OS, data is stored on storage device in a standard format called count-key data (CKD) or extended count-key data (ECKD). z/VM and Linux on Z also can make use of CKD or ECKD.

IBM Z also supports Small Computer System Interface (SCSI), an ANSI standard electronic interface to communicate with storage devices. z/VM and Linux on Z can make use of SCSI.

The IBM DS8000 series of storage systems support both ECKD and SCSI and provide the following services for resiliency:

- z-synergy Services includes z/OS functions that are supported on the storage system. The functions include zHyperLink, transparent cloud tiering, IBM High Performance FICON®, PAV, HyperPAV, SuperPAV, and z/OS Distributed Data Backup.
- IBM HyperSwap is a high-availability feature that provides dual-site, active-active access to a volume. Data that is written to the volume is automatically sent to copies at each site. If one site is no longer available, the other site can provide access to the volume.
- Copy Services helps you implement storage solutions to keep your business running 24 hours a day, 7 days a week by providing data duplication, data migration, and disaster-recovery functions. The functions include Global Mirror, Metro Mirror, Metro/Global Mirror, Point-in-Time Copy/IBM FlashCopy®, z/OS Global Mirror, Safeguarded Copy, and z/OS Metro/Global Mirror Incremental Resync (RMZ).
- Copy Services Manager on Hardware Management Console enables IBM Copy Services Manager to run on the Hardware Management Console, which eliminates the need to maintain a separate server for Copy Services functions.

In addition, IBM Z can use tape devices to store and access data sequentially. Data is typically stored on tape for the following reasons:

- Backup, where copies of volumes or data sets are written to tape and remain onsite or are moved to an off-site vaulting facility.
- Archive, where the data might be kept for a time because of mandatory regulations. The data is no longer part of the day-to-day cycles of the business applications, so the access requirements to it are minimal or non-existent.
- Migration, where infrequently accessed data can be stored on tape instead of occupying disks. Therefore, the data might be accessed only monthly and does not require a high-performance medium.

There is also the concept of Virtual Tape Server (VTS) which means storing data in tape format on disk. The use of VTS over real tapes can be the preferred method for the following reasons:

- Optimize resource consumption for certain types of data sets
- Reduce contention on real tape devices
- Reduce wastage of real tape capacity

The TS7700 Virtualization Engine is a modular, scalable, and high performing architecture for IBM Z tape virtualization. It incorporates extensive self-management capabilities consistent with IBM Information Infrastructure initiatives. These capabilities can improve performance and capacity. Better performance and capacity help lower the total cost of ownership for tape processing and avoid human error. A TS7700 Virtualization Engine can improve the efficiency of Z tape operations by efficiently using disk storage, tape capacity, and tape speed. It can also improve efficiency by providing many tape addresses.

- A TS7700 Grid refers to two or more physically separate TS7700 Clusters connected to one another through a customer-supplied Internet Protocol network. A TS7700 Grid can contain disk-only clusters that do not attach to a physical tape library, clusters that do attach to a physical tape library, or a combination of both.
- The TCP/IP infrastructure that connects a TS7700 Grid is known as the Grid Network. The grid configuration is used to form a disaster-recovery solution and provide remote virtual volume replication. The term *grid* refers to the code and functionality that provides replication and management of virtual volumes and their attributes in cluster configurations. A Grid can be used to form disaster-recovery and high-availability solutions. A disaster-recovery solution is achieved when multiple clusters are geographically distant from one another. A high-availability solution is achieved when multiple clusters are in close proximity to one another.

Storage connectivity on the IBM Z platform is provided by IBM Fibre Connection (FICON) and IBM zHyperLink Express features.

FICON Express

FICON Express features follow the established Fibre Channel (FC) standards to support data storage and access requirements, along with the latest FC technology in storage and access devices. FICON Express features support the following protocols:

FICON

FICON provides for communication across channels; channel-to-channel (CTC) connectivity and with FICON devices, such as storage and tape devices, using extended count-key data (ECKD). FICON is used in z/OS, IBM z/VM, IBM z/VSE®, z/TPF, and Linux on Z environments.

Fibre Channel Protocol

Fibre Channel Protocol (FCP) is a standard protocol for communicating with storage disk and tape devices. The FCP channel can connect to FC SAN fabrics and access SCSI devices. FCP is used by z/VM, KVM, z/VSE, and Linux on Z environments.

High Performance FICON

High Performance FICON (zHPF) is a channel I/O architecture that is designed to improve the execution of small block I/O requests. zHPF enables multiple channel commands to be sent to the control unit as a single entity. The FICON channel forwards a chain of commands and does not need to keep track of each single channel command word (CCW)[3]. This leads

[3] A CCW is the I/O operation used for communications with the channel subsystem. The CCW contains a channel command, such as read, write, or control, along with the data address of the data area involved.

to cost reduction and increases the maximum I/O rate on a channel. The performance improvement depends on your workload.

zHyperLink Express

zHyperLink Express provides a low-latency connection from IBM Z to storage subsystems for faster data retrieval. zHyperLink Express is a direct connect, short distance IBM Z I/O feature designed to work with FICON or High-Performance FICON SAN infrastructures. The zHyperLink Express feature allows you to make synchronous requests for data that is in the storage cache of the IBM DS8800, which reduces the latency and improves the workload response time.

FICON Express and zHyperLink Express features can be concurrently added and removed. The resiliency capabilities include:

- Multiple channel path support
- Concurrent repair/add of all features

For more information about the available FICON Express and zHyperLink Express features, see *IBM Z Connectivity Handbook*, SG24-5444:
http://www.redbooks.ibm.com/abstracts/sg245444.html?Open

2.2.3 Network

The IBM Z is a fully virtualized platform that can support many system images at once. Therefore, network connectivity covers the connections between the platform and external networks with Open Systems Adapter-Express (OSA-Express) and RoCE Express features. And it also supports specialized internal connections for intra-system communication through IBM HiperSockets and Internal Shared Memory (ISM).

- The IBM Open Systems Adapter-Express (OSA-Express) is an I/O feature that can be installed into the PCIe drawers. It provides LAN connectivity. In addition, the OSA-Express can assume some of the PUs workload for several functions of the TCP/IP stack, which results in significant performance benefits by offloading processing from the operating system.
- IBM HiperSockets is an integrated function of IBM Z that supplies attachments to virtual local area networks with minimal system and network overhead. HiperSockets provides LAN connectivity across multiple system images on the same Z platform by performing memory-to-memory data transfers in a secure way. The HiperSockets function eliminates the use of I/O subsystem operations and the use of an external network connection.
- IBM RoCE Express feature uses Remote Direct Memory Access (RDMA) over Converged Ethernet (RoCE) to provide fast memory-to-memory communications between two Z platforms. The feature is designed to help reduce consumption of CPU resources for applications that use the TCP/IP stack. It can also help reduce network latency with memory-to-memory transfers by using Shared Memory Communications - Remote Direct Memory Access (SMC-R). With SMC-R, you can transfer huge amounts of data quickly, at low latency. SMC-R is not apparent to the application and requires no code changes, which enables rapid time to value.
- Internal Shared Memory is a virtual PCI network adapter that enables direct access to shared virtual memory, providing a highly optimized network interconnect for Z platform intra-communications. Shared Memory Communications-Direct Memory Access (SMC-D) uses ISM. SMC-D optimizes operating systems communications in a way that is transparent to socket applications. It also reduces the CPU cost of TCP/IP processing in the data path, which enables highly efficient and application-transparent communications. SMC-D requires no extra physical resources (such as RoCE Express features, PCIe

bandwidth, ports, I/O slots, network resources, or Ethernet switches). Instead, SMC-D uses LPAR-to-LPAR communication through HiperSockets or an OSA-Express feature for establishing the initial connection.

OSA Express and RoCE Express features can be concurrently added and removed. The resiliency capabilities include:

- Multiple channel path support
- Concurrent repair/add of all features

For more information about the available OSA Express and RoCE Express features, see *IBM Z Connectivity Handbook*, SG24-5444:
http://www.redbooks.ibm.com/abstracts/sg245444.html?Open

I/O channels are components of the IBM z/Architecture. They provide a pipeline through which data is exchanged between systems or between a system and external devices (in storage or on the network). z/Architecture channel connections, referred to as *channel paths*, have been a standard attribute of all IBM Z platforms that date back to the IBM S/360. Over the years, numerous extensions have been made to the z/Architecture to improve I/O throughput, reliability, availability, and scalability. One of the many key strengths of the Z platform is the ability to deal with large volumes of simultaneous I/O operations. The channel subsystem (CSS) provides the function for Z platforms to communicate with external I/O and network devices and manage the flow of data between those external devices and system memory. This objective is achieved by using a system assist processor (SAP) that connects the CSS to the external devices.

The mechanisms that are used to control I/O operations are collectively called the channel subsystem (CSS). The CSS directs the flow of information between I/O devices and main storage. A CSS relieves the PUs of the tasks of communicating directly with I/O devices. And it permits data processing to proceed on the CPUs while other data is transferred concurrently to and from the I/O devices.

The CSS uses one or more channel paths by using a Channel Path ID (CHPID) as the communication link to manage the flow of information to and from I/O devices.

2.2.4 Virtualization

Virtualization is a principle strength of the IBM Z platform. It is embedded in the architecture and built into the hardware, firmware, and operating systems. For decades, Z platforms have been designed based on the concept of partitioning resources (such as CPU, memory, storage, and network resources). So, each set of features can be used independently with its own operating environment.

Every Z platform is highly virtualized, with the goal of maximizing utilization of computing resources, while lowering the total number of resources and cost needed to run critical workloads and solutions.

Virtualization can help secure and isolate application workloads and data within virtual servers and storage devices for easier replication and restoration. This added resiliency can provide you with greater flexibility to maintain a highly available infrastructure, while performing planned maintenance, and to configure low-cost disaster-recovery solutions. Virtualization technologies solve many traditional backup issues, because they decouple the bindings between the operating system (with the application and data) and the underlying hardware.

Virtualization requires a hypervisor, which is the control code that manages resources that are required for multiple independent operating system images. Hypervisors can be implemented as software or hardware (firmware). Z platforms have both. The hardware hypervisor is IBM PR/SM. PR/SM is implemented in firmware as part of the base system that fully virtualizes the system resources and runs without any additional software. A software hypervisor is implemented with z/VM and KVM for IBM Z, both of which use PR/SM functions.

The hypervisors are designed to enable simultaneous execution of multiple operating systems, providing operating systems with virtual resources. The various virtualization options in IBM Z allow you to build flexible virtualized environments to take advantage of open source software as well.

Multiple hypervisors and operating systems can exist in different logical partitions (LPARs) within the same IBM Z platform, as shown in Figure 2-2.

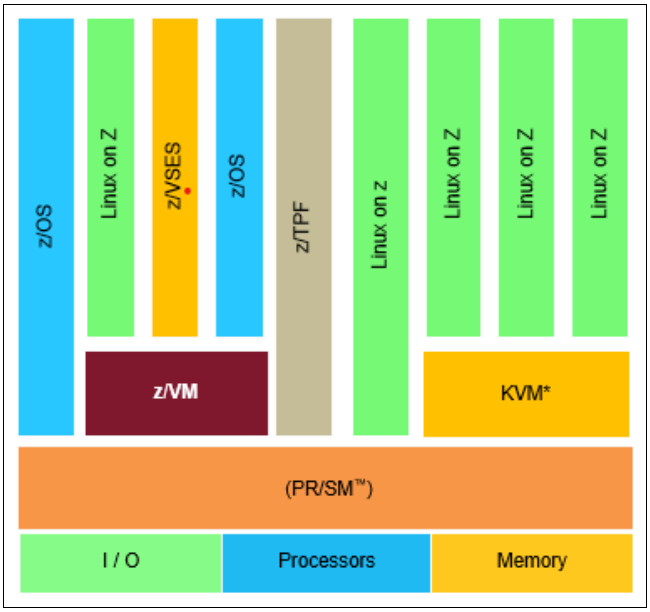


Figure 2-2 Supported hypervisors and operating systems

PR/SM

Unique to IBM Z, PR/SM is a Type-1 hypervisor that runs directly on bare metal, allowing you to create multiple LPARs on the same physical server. PR/SM is a highly stable, proven, and secure, firmware-encapsulated virtualization technology that allows multiple operating systems to run on the same physical platform. Each operating system runs in its own logical partition.

PR/SM logically partitions the platform across the various LPARs to share resources, such as processor units, memory, and I/O (for networks and storage), allowing for a high degree of virtualization.

z/VM

z/VM is a Type-2 hypervisor that allows sharing the Z platform's physical resources, such as disk, memory, network adapters, and CPUs (called CPs and IFLs). These resources are managed by the z/VM hypervisor, which typically runs on an LPAR, and other virtual machines (VMs) that run on top of the hypervisor. Typically, the z/VM hypervisor is used to run Linux virtual servers, but other operating systems (such as z/OS) can also run on z/VM. z/VM is a proven and well-established virtualization platform. It provides industry-leading capabilities to efficiently scale both horizontally and vertically.

z/VM provides each user with an individual working environment known as a virtual machine, which appears to a guest operating system as hardware, but is simulated by z/VM and the Z hardware. The virtual machines under z/VM share the total system resources. Processor and memory capacity is allocated to the servers that need it, when they need it. The virtual machine simulates the existence of a dedicated real machine, including processor functions, storage, and I/O resources.

KVM for IBM Z

KVM for IBM Z is an open source hypervisor solution that provides simple, cost-effective server virtualization for Linux workloads that run on the Z platform. It is a Type-2 hypervisor that allows you to share real CPUs (called IFLs), memory, and I/O resources through platform virtualization. It can coexist with z/VM environments: Linux on IBM Z, z/OS, z/VSE, and z/TPF. KVM for IBM Z is optimized for scalability, performance, security, and resiliency, and provides standard Linux and KVM interfaces for simplified operational control.

The KVM for Z is offered by the following Linux distributions:

- Red Hat Enterprise Server
- SUSE Linux Enterprise Server
- Canonical Ubuntu

2.2.5 Range of resiliency for IBM Z environments

The hardware, firmware, operating systems, middleware, and application components of the IBM Z platform have been designed to work together closely and provide high levels of resiliency. IBM Z design approaches application and data availability with an integrated and cohesive strategy that encompasses single-system, multi-system, and multi-site environments (see Figure 2-3).

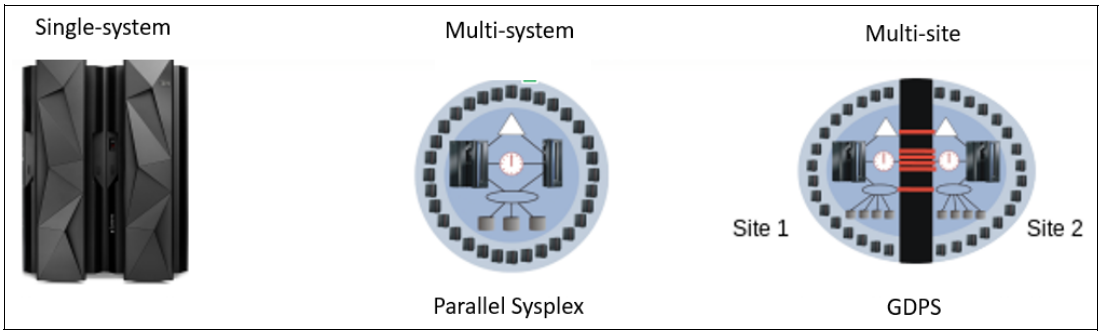


Figure 2-3 Range of IBM Z resiliency

Single-system environment

By design, the IBM Z platform is a highly resilient system. Through RAS design principles, the z/Architecture has built-in self-detection, error correction, and redundancy. The Z platform reduces single points of failure to deliver the best reliability of any enterprise system in the industry. Transparent processor sparing, and dynamic memory sparing enable concurrent maintenance and seamless scaling without downtime. The operating system has code to seamlessly recover from failures and even predict failures before they happen. Workloads, applications, and middleware that have failed can be restarted in place.

Businesses that have a single Z platform also see value in a Parallel Sysplex environment for availability purposes. Sometimes called a "*Sysplex-in-a-box*", it can keep applications available across software upgrades by allowing rolling IPLs. A single-system Parallel Sysplex can be configured with data sharing across two or more z/OS images, connecting to one or

two internal CFs. Although this does not provide for additional hardware redundancy, the hardware mean time to failure is so good that this might not be perceived as high risk for certain SLA objectives.

Multi-system environment

Beyond the built-in redundancy and resiliency of the IBM Z platform, IBM has created technologies that span multi-system environments to achieve application availability, these include:

- A sysplex, which is a cluster of independent instances of the z/OS operating system. Systems within a sysplex can communicate by using channel-to-channel (CTC) or OSA-Express connections between logical partitions (LPARs) with a cross-system coupling (XCF) service. Communication between LPARs in the same Z platform or across Z platforms are handled in the same way. Communication between these LPARs is effectively (from a network perspective) instantaneous. With exploitation by z/OS console services, it provides a single system image to operations staff.
- The Parallel Sysplex adds to the sysplex with the addition of the Coupling Facility (CF) and coupling links for high-speed cross-system communication. The primary goal of a Parallel Sysplex is to provide data sharing capabilities, allowing multiple databases for direct reads and writes to shared data. A Parallel Sysplex can:
  - Remove single points of failure within the Z platform, LPAR, or subsystems
  - Improve application availability
  - Give a single system image to users and applications with:
    - Dynamic Session Balancing
    - Dynamic Transaction Routing
  - Provide scalable capacity

  Inter-image (z/OS-to-z/OS) communication and the managing of data structures on behalf of applications are handled by the Coupling Facility (CF). The CF can run either as a separate LPAR (internal CF or ICF) or in dedicated Z hardware (external CF). An ICF can be used as a backup for an external CF.

  Through XCF, systems in the sysplex can also determine whether a sysplex member is communicating, and if not, XCF can trigger multi-system recovery.

  Figure 2-4 on page 29 represents a simple Parallel Sysplex configuration.

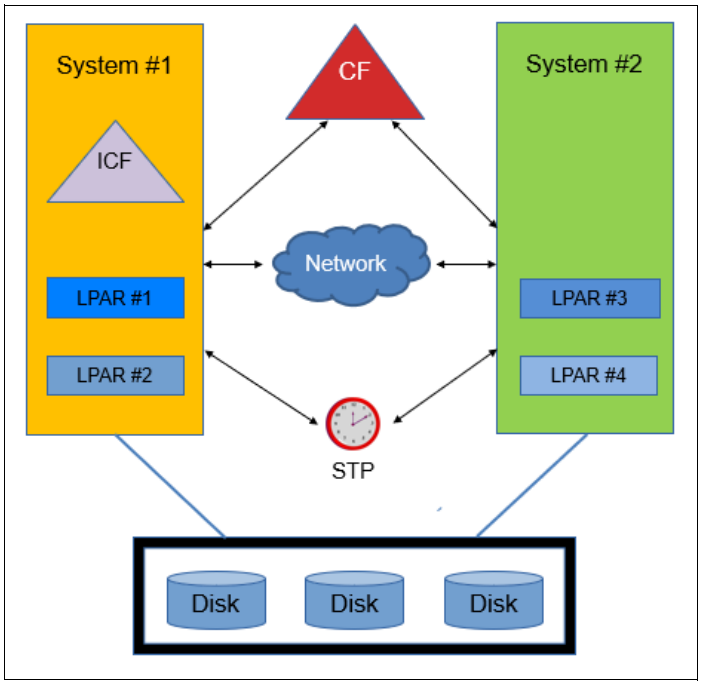


Figure 2-4 Parallel Sysplex with ICF and CF

In a Parallel Sysplex, the objective is non-disruptive planned or unplanned outages. From a hardware perspective, connectivity is a primary area of focus, as are other basic hardware components, such as the CF, Server Time Protocol (STP)[4], and coupling links. All CF links and connections to FICON directors and network switches are included in the resiliency design. This approach ensures cross-connectivity to all devices in the cluster and to potentially eliminate the effects of any outages. For availability purposes, the following convention is common practice in a Parallel Sysplex environment: at least two links to connect each system to each CF, to all storage, and to the network.

Each member of the Parallel Sysplex can concurrently cache shared data in local processor memory through hardware-assisted cluster-wide serialization and coherency controls. As a result, work requests that are associated with a single workload — such as business transactions or database queries — can be dynamically distributed for parallel execution on members in the sysplex cluster, based on available processor capacity. This provides the ability to perform non-disruptive hardware and software maintenance and installations. Through data sharing and dynamic workload management, members can be dynamically removed from or added to the cluster. As a result, installation and maintenance activities can be performed while the remaining systems continue to process work. Furthermore, software and hardware upgrades can be introduced one system at a time. This rolling IPL capability allows you to roll changes through systems at a pace that makes sense for your business.

The Parallel Sysplex can be viewed as a single logical resource to end users and applications. Just as work can be dynamically distributed across the individual processors within a single SMP system, so too can work be directed to any member in a Parallel Sysplex that has available capacity. There is no need to manually partition data or applications among individual members or to replicate databases across multiple systems.

[4] STP is implemented in LIC as a system-wide facility of the Z platform. The Z platform is enabled for STP by installing the STP feature code 1021.

Multi-site environment

IBM GDPS technology provides a total business continuity solution for z/OS multi-site environments. GDPS is a collection of end-to-end automated disaster-recovery solutions on the IBM Z platform, each addressing a different set of IT resiliency goals that can be tailored to meet the recovery objectives for your business. Depending on the solution, GDPS provides support for:

- Managing and monitoring the remote copy environment
- Managing and monitoring the production environment
- Data that spans more than one platform including z/OS, z/VM, and Linux on Z
- Multiple disk subsystem vendors
- Transparent and near-continuous application availability from disk failures
- Automated and transparent failover and failback[5]

GDPS extends the resource sharing, workload balancing, and continuous availability benefits of a Parallel Sysplex environment. It also significantly enhances the capability of an enterprise to recover from disasters and other failures, and to manage planned exception conditions. As a result, you can achieve your own continuous-availability and disaster-recovery goals.

GDPS is a collection of several offerings, each addressing a different set of IT resiliency goals that can be tailored to meet the RPO and RTO for your business.

Each offering uses a combination of system and storage hardware or software-based replication and automation and clustering software technologies:

- GDPS Metro
- GDPS Metro HyperSwap Manager
- IBM GDPS Virtual Appliance
- GDPS Global - XRC (also known as GDPS XRC)
- GDPS Global - GM (also known as GDPS GM)
- GDPS Metro Global - GM (also known as GDPS MGM)
- GDPS Metro Global - XRC (also known as GDPS MzGM)
- GDPS Continuous Availability

GDPS Metro provides continuous availability, disaster recovery, and production system / sysplex resource management capabilities. Based on Metro Mirror synchronous disk mirroring technology, it can achieve RPO = 0. Typically the recovery time is less than one hour (RTO < 1 hour) following a complete site failure. GDPS Metro also supports HyperSwap to provide near-continuous disk availability after a disk failure.

GDPS Global - GM provides disaster recovery and production system / sysplex resource management capabilities. Based on Global Mirror synchronous disk mirroring technology, it can achieve data loss down to 3 seconds (RPO = 3 seconds) and RTO < 1 hour. GDPS Global supports two sites separated by virtually unlimited distances with minimal impact to end-user response times.

Three- and four-site configurations are available such as combining GDPS Metro with GDPS Global - GM to form GDPS MGM. GDPS MGM is capable of:

- Down to zero data loss
- HyperSwap for disk availability
- Protection from regional events
- Minimal impact to end-user response time

[5] The ability to move production systems back and forth between production and recovery sites with minimal or no manual intervention in a planned or unplanned manor.

For more information about GDPS offerings, see *IBM GDPS: An Introduction to Concepts and Capabilities*, SG24-6374 here:
http://www.redbooks.ibm.com/abstracts/sg246374.html?Open

2.3 Operating system layer

Advanced capabilities are introduced with each new Z platform that cause it to move ahead of its predecessor platforms in terms of efficiency, flexibility, security, reliability, and much more. Whenever new capabilities are implemented, the z/Architecture[6] is extended rather than replaced, which helps sustain the compatibility, integrity, and longevity of the Z platform. Thus, protection and backward compatibility of existing workloads and solutions are key for the new capabilities introduced in the z/Architecture.

To handle new and different workloads, the scope of software and application options must be accommodated by the operating system. Also, the hardware and firmware components of the system must provide a viable option to integrate functionality into the architecture's capabilities. Hence, the Z platforms and operating systems always conform to the z/Architecture to ensure support of current and future workloads and solutions.

The Z platform supports multiple operating systems that have different characteristics and purposes, including:

- z/OS, a widely used operating system, is designed to offer a stable, secure, and continuously available environment for applications. See 2.3.1, "z/OS" on page 32 for more information.
- As a control program, z/Virtual Machine (z/VM) is a hypervisor, because it runs other operating systems in the virtual machines that it creates. For more information see:
  https://www.ibm.com/support/knowledgecenter/zosbasics/com.ibm.zos.zmainframe/zconc_opsyszvmintro.htm

  In addition, z/VM supports a single system image (SSI) cluster, which is a multi-system z/VM environment managed as a single resource pool, where guest virtual machines can be moved from one system to another while they are running.

  This mobility function is called live guest relocation (LGR). Relocating guests can be useful for load balancing and for moving workload off a physical server or member system that requires maintenance. For example, LGR can be used to allow critical Linux images to continue to run their applications during planned system outages.

  For more information, see *z/VM: CP Planning and Administration*, SC24-6271.
- z/Virtual Storage Extended (z/VSE) is popular with users of smaller IBM Z platforms. Some of these customers eventually migrate to z/OS when they grow beyond the capabilities of z/VSE. For more information see:
  https://www.ibm.com/support/knowledgecenter/zosbasics/com.ibm.zos.zmainframe/zconc_opsyszvseintro.htm
- There are several Linux distributions that can run on IBM Z: Red Hat Enterprise Server, SUSE Linux Enterprise Server, and Canonical Ubuntu.

  Linux on Z automatically inherits the strengths and reliability features of Z hardware. Linux on Z is used with z/VM or KVM to provide an efficient and cost-effective solution. For more information see:
  https://www.ibm.com/it-infrastructure/z/os/linux-tested-platforms

[6] IBM z/Achitecture is the conceptual structure of the Z platform that determines its basic behavior.

- The z/Transaction Processing Facility (z/TPF) operating system is a special-purpose system, used in environments with very high transaction volume. For more information see: https://www.ibm.com/support/knowledgecenter/zosbasics/com.ibm.zos.zmainframe/zconc_opsysztpfintro.htm

2.3.1 z/OS

z/OS is the result of decades of technological advancement. It evolved from an operating system that could process a single program at a time to an operating system that can handle many thousands of programs and interactive users concurrently.

z/OS is designed to deliver the highest quality of service for enterprise transactions and data, and extends these qualities to new applications by using the latest software technologies. It provides a highly secure, scalable, high-performance base on which to deploy modern and traditional applications.

z/OS RAS is designed around z/Architecture principles with the premise that problems will occur and the Z stack must be able to handle them. Design goals include:

- Detecting and identifying errors, and isolating problems
- Establishing recovery environments to handle First Failure Data Capture (FFDC)
- Enabling processing to continue or recover
- Releasing resources to avoid impacting other processing
- Labeling code and control blocks for easy identification
- Implementing robust system tracing
- Capturing diagnostic data
- Analyzing error data for problem analysis
- Ensuring reliable program operation via the verification of parameter inputs

The ability for z/OS to recover from a failure is based on the capabilities of each individual subsystem, application, middleware, and program to determine:

- When a failure occurs: 1) How a failure can be handled. 2) How a failure can be avoided.
- The severity of a failure to its operation and whether processing must be terminated.
- What data is required to expedite diagnosis of a failure.

Failures can be classified into three categories:

- Masked failures are when software or hardware detects a failure and corrects the problem with no impact to the application or workload. A hardware example is when a power supply fails, the system immediately switches to an alternate source and alerts IBM to replace the faulty unit.
- Hard failures are when the hardware or software detects a failure and immediately terminates the component to avoid impact to other programs. Automation can restart the failing component or subsystem, recovering from the problem with minimal impact.
- Soft failures are when the user detects a failure, but the cause of the failure might not be clear and often difficult to identify, much less recover from. Examples include:
  - A component holds resources like locks or enqueues, preventing other work from continuing.
  - Slow, eventual exhaustion of a resource, such as common storage, private virtual storage, or JES spool.
  - A build-up of serialization-related contention.
  - Normal or abnormal events that increase substantially compared to *steady-state*.

Detection of failures leads to the requirement that all z/OS subsystems, applications, middleware, and programs provide recovery routines to be given control when a failure or problem interrupts their processing. In general, recovery routines have three functions:

1. Analyze the failure and decide whether the subsystem, application, or program that experienced it can continue its processing or must be terminated.
2. Attempt recovery by repairing or limiting the effects of the failure.
3. Collect and record data related to the failure so that it can be subsequently analyzed to determine the cause of the problem.

z/OS provides services to allow its subsystems, applications, middleware, and programs to more easily build recovery characteristics into their basic design. These services are not limited to defining recovery routines. They also include functions and techniques for a subsystem, application, and program to provide — as part of its normal processing — the data that will be needed for problem analysis if a failure does occur. The data that should be kept for possible future failure analysis includes:

- The description of the error, identification of the subsystem, application, or program, and the location and contents of its data areas.
- Which functions were completed. Which function was being performed at the time.
- Which resources were allocated at the time.

The z/OS services that preserve this data include dumping, tracing, error log recording, and message logging. The data can also be analyzed by the recovery routine and is almost always an essential part of the data that the recovery routine records on some external medium for later failure analysis.

IBM has solutions to analyze operating system and middleware components to help meet *best practices*, predict problems that might lead to soft failures, and diagnose the current system if a soft failure is suspected. Other examples include the detection of a sysplex member that is inoperable and removal of that system from the sysplex before it causes data integrity issues. These solutions are delivered in the base z/OS operating system, related to preventing, identifying, and diagnosing soft failures. Other solutions include:

- Runtime Diagnostics is a base component of z/OS designed to help you analyze a system with a potential problem or soft failure. Soft failures in particular are often difficult or impossible to detect and can slowly lead to the degradation of the solution that is using z/OS.

  In many cases, when Runtime Diagnostics finds a critical message, it performs additional analysis based on the job name or other information in the message text. For example, if Runtime Diagnostics identifies an XCF stalled-connector message, it processes additional analysis of the identified address space to help narrow down the problem. A key feature of Runtime Diagnostics is its ability to summarize internal processing errors and return the results to you in a message response. It does so much quicker than someone attempting to identify symptoms manually.
- AutoIPL is an automated function, which is defined in the DIAGxx parmlib member, that the system checks at wait state time. AutoIPL can re-IPL z/OS LPARs, take a SADMP[7], or even take a SADMP and have the system LPAR re-IPL when the dump finishes.
- IBM Health Checker for z/OS is a component to help identify potential configuration problems before they impact availability or cause system outages. Check routines programmatically check the current active z/OS and sysplex settings and definitions for a system and compare them to IBM-recommended best practice configuration values or user-defined settings. It generates output with detailed messages to inform you of any

[7] The stand-alone dump program (SADMP) produces a dump of z/OS memory when the system has failed.

potential problems and suggested actions to take to resolve them. IBM Health Checker for z/OS runs continuously on the system to detect potential problems and to alert you if such a situation exists. Although it is not intended as a monitoring tool, you can run checks after a change window to catch inadvertent configuration settings that might lead to unexpected behavior.

IBM Health Checker for z/OS consists of the following two parts:

- Framework
  The framework is an interface that manages services like check registration, messaging, scheduling, command processing, logging, and reporting. It is an open architecture in support of check development by IBM products, independent software vendors (ISVs), and customers.
- Checks
  The checks are routines that evaluate component, element, or product-specific settings and definitions, looking for potential problems on a running system. Checks are independent of the framework.

  Checks are provided by IBM and ISVs as part of the component. For example, IBM RACF® checks are provided with RACF, and are not available to installations that do not have RACF installed. You can also write your own checks in assembler language, IBM Metal C, or system REXX.

  IBM also ships migration checks that can be activated before migration to a new z/OS release. For more details, see:
  https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.2.0/com.ibm.zos.v2r2.e0zm100/mighlch.htm

In addition, Health Checker for z/OS can help improve availability for cross-system communication in a Parallel Sysplex or data sharing through the use of the Coupling Facility.

- Predictive Failure Analysis (PFA) is designed to predict potential problems with your systems. PFA extends availability by going beyond failure detection to predict problems before they occur. PFA provides this support by using remote checks from IBM Health Checker for z/OS to collect data about your installation. Using this data, PFA constructs a model of the expected or future behavior of the z/OS images and compares the actual behavior with the expected behavior. If the behavior is abnormal, PFA issues a health check exception. PFA uses a z/OS UNIX System Services (z/OS UNIX) file system to manage the historical and problem data that it collects.

z/OS resiliency capabilities

The key storage, network, and clustering capabilities (from a z/OS perspective), for continuous access to data and applications are described in the following subsections.

Storage

In a z/OS environment, space management involves the allocation, placement, monitoring, migration, backup, recall, recovery, and deletion of data sets. The primary method for managing space in z/OS is through the Data Facility Storage Management Subsystem (DFSMS), which is a suite of related data and storage management products.

There are five different DFSMS components that provide specific data management options to allocate data in the correct media type and volume, and provide data access, data availability, and data retention management. The following components make up DFSMS:

- DFSMSdfp is the base component of z/OS. It provides storage, data, program, and device management. It consists of access methods, OPEN/CLOSE/EOV routines, catalog

management, DADSM (DASD space control), utilities, IDCAMS, SMS, NFS, ISMF, and other functions.

- DFSMSdss is optional. It provides data movement, copy, backup, and space management functions.
- DFSMShsm is optional. It provides backup, recovery, migration, and space management functions. It calls DFSMSdss for certain functions.
- DFSMSrmm is optional. It provides management functions for removable media such as tape cartridges and optical media.
- DFSMStvs is optional. It enables batch jobs and IBM CICS online transactions to update shared VSAM data sets concurrently.

DFSMS can ensure that space is available on your volumes and that backup copies of data sets are always available. Copies of all vital data sets should be stored at a separate location to satisfy disaster-recovery requirements.

When data management is automated, the operating system determines object placement and automatically manages data set backup, movement, space, and security. Typically, a z/OS environment includes both manual and automated processes for managing data sets.

For more information about DFSMS, refer to *ABCs of IBM z/OS System Programming Volume 3*, SG24-6983, here: http://www.redbooks.ibm.com/redbooks/pdfs/sg246983.pdf

Other high-availability options for data in z/OS include these options:

- Virtual Storage Access Method (VSAM) is an access method that is primarily for application use, such as IBM CICS. VSAM Record-level sharing (RLS) is an access mode for VSAM data sets supported by DFSMS. With RLS, CICS regions that share VSAM data sets can reside in one or more z/OS images within a sysplex.
- z/OS File System (zFS) is a UNIX System Services file system that stores files in VSAM linear data sets. zFS data sets and z/OS data sets can reside on the same volume. zFS supports shared file system capability in a sysplex environment and seamless moves of mounts from one sysplex member to another by using the AUTOMOVE option of the MOUNT command. The term shared file system environment refers to a sysplex that has a specification of SYSPLEX(YES) in the BPXPRMxx parmlib member. That is, users in a sysplex can access zFS data that is owned by another system in the sysplex.
- PAV and HyperPAV are concepts of using multiple devices or aliases to address a single IBM ECKD disk device (“Parallel Access Volumes”) to increase data redundancy. If there is no aliasing of disk devices, then only one I/O transfer can be in progress to a device at a time, regardless of the actual capability of the storage server to handle concurrent access to devices. HyperPAV simplifies operational procedures. It removes static alias-to-base bindings and associates a PAV alias device in a pool of available PAV base and alias devices with any base device in a pool per a request that is instantaneously serviced. Upon completion of an I/O request that a HyperPAV alias device services, the alias immediately returns to the pool of available devices. This significantly reduces a number of I/Os, and increases I/O parallelism. HyperPAV does not require coordination throughout a sysplex because each system with access to a HyperPAV-enabled storage server has the same pool of available base and alias devices. Use of HyperPAV alias devices allows significant reduction of their number and definition of many more base devices. Remember that HyperPAV alias devices and corresponding base devices are managed through the same logical control unit within the range of unit addresses from 00 through 255 (ff).

  The Base and HyperPAV devices are defined on the storage server and on the IOCDS on Z, whereas the DASD devices are defined with unit addresses. An LCU is a logical set of DASD devices on an IBM DS8K disk storage unit, and it can have up to 256 possible

DASD device addresses from 00 to ff (255). After the base devices are defined, any remaining device numbers in an LCU can be used as an alias by the system to access any of the base addresses in the same LCU.

SuperPAV is an extension of the HyperPAV architecture and implements multiple logical subsystems (LSS) within an alias management group (AMG). With SuperPAV, when a new I/O request occurs — and there are no alias PAV devices available in the alias pool for the base PAV device's LSS — z/OS attempts to use an alias PAV device from another LSS within the AMG subgroup. SuperPAV can provide relief for systems that experience high I/O queue time (IOSQ) during periods of peak I/O load. When a small number of aliases have been defined in an LSS, it is possible that during a heavy I/O period, aliases might not be available. z/OS checks peer LSS alias pools to borrow an alias to start I/O requests, which previously would have been left queued when aliases were not available.

- z/OS Basic HyperSwap is a base z/OS function. It provides high availability of data for disk storage system failures. Basic HyperSwap does not replace a disaster recovery solution. If a session is suspended, but the suspend operation was not caused by a HyperSwap trigger, the session is not frozen to ensure that it is consistent. Basic HyperSwap replication performs the following actions:
  - Manages CKD volumes in a Metro Mirror relationship
  - Monitors events that indicate a storage device failed
  - Determines whether the failing storage device is part of a Metro Mirror pair
  - Determines the action to be taken from policy
  - Ensures that data remains consistent
  - Swaps the I/O between the primary logical devices in the consistency group with the secondary logical devices in the consistency group

  When HyperSwap is combined with Metro Mirror or Metro Global Mirror replication, you can prepare your system for disaster recovery and ensure high availability of data. If a session is suspended but the suspend operation was not caused by a HyperSwap trigger, the session is frozen to ensure that it is consistent.

  Copy Services Manager for IBM Z provides high availability and disaster recovery for multiple sites. It builds upon z/OS Basic HyperSwap with support for Point in Time and Remote Copy configurations, including three- and four-site management and HyperSwap technology. CSM was designed to support fast failover/failback, fast reestablishment of mirroring, and data recoverability management at the remote site. CSM handles volume-level copy service management, including Data Consistency across a set of volumes with logical dependencies.

  CSM coordinates the following copy service functions:

  - FlashCopy
  - Metro Mirror
  - Global Mirror
  - Metro Global Mirror
  - Multi-Target Metro Mirror
  - Basic HyperSwap

Note: z/OS Copy Services Management (CSM) handles management of data replication for z/OS HyperSwap. You must provide and maintain automation to determine that there was a problem and to restore all business application data (and to complete other required steps).

For more information on IBM Copy Services Manager (CSM), see https://www.ibm.com/support/knowledgecenter/en/SSESK4_6.2.3/com.ibm.storage.csm.help.doc/frg_c_ugover.html

Network

z/OS Communications Server is the IBM implementation of the standard TCP/IP protocol suite on the z/OS platform. TCP/IP is a component product of the z/OS Communications Server, and it provides a multitude of technologies. Collectively, those technologies provide a resilient open systems environment for the development, establishment, and maintenance of applications and systems.

High availability can be provided for applications on the IBM Z platform by using the following features:

- Virtual IP addressing (VIPA) provides physical interface independence for the TCP/IP stack (the part of a z/OS Communications Server software that provides TCP/IP protocol support) and applications so that interface failures do not impact application availability.
- Address Resolution Protocol (ARP) takeover enables your system to transparently exploit redundant physical interfaces without implementing a dynamic routing protocol in your Z.
- Dynamic routing uses network-based routing protocols (such as OSPF) in the Z environment to use redundant network connectivity for higher availability (when used with VIPA).
- Port sharing enables you to run multiple instances of an application for higher availability and scalability (to the extent possible in a single system).

High availability can be provided for applications that use IBM Parallel Sysplex technology with the following features:

- Dynamic Virtual IP Address (DVIPA) provides TCP/IP application availability across z/OS systems in a sysplex and allows participating TCP/IP stacks to provide backup and recovery for each other, for planned and unplanned TCP/IP outages.
- Sysplex Distributor provides intelligent load balancing for TCP/IP application servers in a sysplex, and along with Dynamic VIPA provides a single system image for applications that connect to those servers.
- The Load Balancing Advisor (LBA) provides z/OS Sysplex server application availability and performance data to outboard load balancers through the Server Application State Protocol (SASP).

For more information on z/OS Communications Server, TCP/IP availability, see *IBM z/OS V2R2 Communications Server TCP/IP Implementation: Volume 3 High Availability, Scalability, and Performance*, SG24-8362:
http://www.redbooks.ibm.com/abstracts/sg248362.html?Open

Clustering

Clustering can be as simple as sharing data with manual control across systems or sharing of locking and enqueuing controls among all systems. The most sophisticated of the clustering techniques is a Parallel Sysplex. This technology allows the linking of up to 32 z/OS systems with nearly linear scalability to create a clustered system. Every system in a Parallel Sysplex has access to all data resources, and every application can run on every system. Simply put, Parallel Sysplex permits multisystem data sharing with high-performance read/write integrity.

Other high-availability options in a sysplex environment include these:

- System logger is a z/OS component that allows an application to log data from a sysplex. You can log data from one system or from multiple systems across the sysplex. The sysplex maintains the shared data logs in the CF.
- Sysplex failure management (SFM) allows you to define a sysplex-wide policy that specifies the actions that z/OS is to take when certain failures occur. You can assign to each system a weight, to be used if signaling connectivity errors occur (WEIGHT parameter). The system also uses the assigned weights with the REBUILDPERCENT value that is specified in the CFRM policy to detect a need to rebuild a structure in a

coupling facility that a system has lost connection to. The policy can specify the use of SFM for automatic recovery of XCF signaling connectivity failures. You can also specify what reconfiguration actions are to be taken when the PR/SM Automatic Reconfiguration Facility is being used.

- z/OS has the ability to run multiple workloads at the same time within one z/OS image or across multiple images. The function that makes this possible is dynamic workload management, which is implemented in the z/OS Workload Manager (WLM) component. WLM classifies work that runs on the z/OS operating system in distinct service classes and defines goals for them that express how the work should be performed. And it uses these goal definitions to manage the work across all systems of a sysplex environment.

  Workload routing support in WLM provides multiple benefits, such as:

  - High availability through redundancy.
  - High performance through increased cluster efficiency, business goals, and importance considerations.
  - Distribution of work across the Parallel Sysplex cluster according to the systems' load. This distribution avoids overloading systems with serious storage constraints in favor of:
    - Systems where similar work meets its goals
    - Underutilized systems
    - Systems where the least important work can be displaced

  The implementation variations can be:

  - Session placement with IBM VTAM® generic resources in a sysplex
  - Transaction routing using sysplex routing services, such as CICS regions available to process workload requests that are located on multiple system members of a Parallel Sysplex.

  For more information see:
  https://www.ibm.com/support/knowledgecenter/en/SSGMCP_5.5.0/fundamentals/wlm/zos-wlm.html

Role of the infrastructure and operating system in the journey to exceptional resilience

Table 2-2 shows how the role of the infrastructure and operating system deepens as your enterprise makes the journey toward higher availability. As stated previously, four deployment models encapsulate the key stages of this journey:

- Deployment model 1: Starting with a resilient and reliable base
- Deployment model 2: Reducing the duration of outages with failover capability
- Deployment model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS
- Deployment model 4: Adding GDPS Continuous Availability for exceptional resilience

The indentions of the following table emphasize how features that are located below an upper level inherit the features of that upper level.

Table 2-1 Role of the infrastructure and operating system in the journey to exceptional resilience \

Deployment model 1: Resilient / Reliable **Single System** Depends on resiliency of the hardware. Tape Data replication FlashCopy (within system)

Deployment model 2: Failover Capable
The preceding list plus,
Multi-system
Sysplex 1+ systems
Resource sharing
Application to LPAR affinities still exist
Coupling Facility
GM / zGM / Metro Mirror data replication
CyberResiliency/ Safeguard Copy
HyperSwap (optional)
High Availability Tape Grid

Deployment model 3: Fault Tolerant
The preceding list plus,
Multi-site (optional for true Fault Tolerance)
Parallel Sysplex with 2+ systems
Rapid Recovery
Dynamic Routing
Workload cloned across LPARs
Data Sharing
CyberResiliency / SGC / LCP
HyperSwap

Deployment model 4: Continuous Availability
The preceding list plus,
Multi-site
Multiple Parallel Sysplexes across 2+ sites with 2+ systems in each
Rapid Recovery within and across sites
Software replication

2.4 Middleware layer

Middleware is software that resides between the operating system and end users or applications. It supplies functions that are not provided by the operating system. Middleware makes collaboration and integration across different applications possible. Typical IBM Z middleware workloads include web serving, data serving, message queuing services, and so on.

As with IBM Z hardware, operating system, and applications, so too is middleware continuously improving with new features and functions to ensure that the IBM Z platform delivers exceptional resiliency.

The integration of IT systems makes essential the ability to interface across many environments with multiple applications reliably and quickly. In this section, the resiliency capabilities of IBM Z message queuing services and data serving middleware are discussed.

2.4.1 IBM MQ for z/OS

Messaging is an effective way to connect systems. It hides many of the details of communication from the application developer and gives a simple interface. This simplification allows the developer to concentrate on the business problem instead of worrying about matters such as recovery, reliability, and operating system differences.

Another feature of messaging solutions is the decoupling of one application from another. Many mechanisms for communicating between applications require that both are available at

the same time. Messaging uses an asynchronous model, meaning that an application that is generating messages does not have to run at the same time as an application that consumes those messages. Reducing the requirements for simultaneous availability reduces complexity and can improve overall availability. Messages can be sent to specific applications or distributed to many separate applications at the same time.

IBM MQ is a robust, reliable, and secure messaging solution. It simplifies and accelerates the integration of different applications across multiple platforms, and it supports a wide range of APIs and languages. IBM MQ allows the server infrastructure to span the data center, mainframe, and cloud frameworks. For example, it allows a GUI desktop application that runs on Windows to reliably communicate with an IBM CICS transaction that runs on IBM z/OS.

With IBM MQ, applications connect to a queue manager and send messages to and receive messages from queues. The queue manager ensures that messages are stored reliably and securely until they are needed by an application. Messages can be persistent, or non-persistent. Persistent messages survive queue manager restart and are written to the queue manager's recovery log. A log is a linear VSAM data set. When it runs, a queue manager tries to keep messages in memory as far as possible, as this allows for faster processing. Over time, as memory fills up, both persistent and non-persistent messages might be written to another type of linear VSAM data set called a page set.

New applications can be written that use alternative interfaces and still maximize the reliability and performance of MQ. Those new applications do not need a complete replacement of existing infrastructure. The applications work with what you already have and know how to manage. At the same time that new interfaces, protocols, and environments are added, much MQ workload continues to be executed in Z-based data centers. Efficient use of, and integration with, the capabilities of the Z hardware and operating systems is critical. Performance, throughput, and availability of MQ for z/OS can be significantly improved when it runs in a Parallel Sysplex, compared to running on a single Z platform.

Developing backup and recovery procedures at your site is vital to avoid costly and time-consuming losses of data. IBM MQ provides means for recovering both queues and messages to their current state after a system failure. The queue manager restart process recovers your data to a consistent state by applying log information to the page sets. If your page sets are damaged or unavailable, you can resolve the problem by using the backup copies of your page sets (if all the logs are available). If your log data sets are damaged or unavailable, it might not be possible to recover completely.

For IBM MQ, creation of a point of recovery must be considered through the use of a reliable backup procedure to ensure reliable recovery. ‘Creating a point of recovery’ is the term used to describe a set of backup copies of page sets and the corresponding log data sets that are required to recover these page sets. These backup copies provide a potential restart point in the event of page set loss (for example, page set I/O error). If you restart the queue manager by using these backup copies, the data in IBM MQ is consistent up to the point that these copies were taken. If all logs are available from this point, IBM MQ can be recovered to the point of failure.

In general, when you decide how often to take backup copies, consider the time needed to recover a page set. The time needed is determined by the following details:

- The amount of log to traverse.
- The time it takes an operator to mount and remove archive tape volumes.
- The time it takes to read the part of the log needed for recovery.
- The time needed to reprocess changed pages.
- The storage medium used for the backup copies.
- The method used to make and restore backup copies.

The more recent your backup copies, the quicker IBM MQ can recover the data in the page sets. The recovery of the page sets depends on all the necessary log data sets being available. For each queue manager, you should take backup copies of the following data by using the utilities that are provided with the product:

- Active log data sets
- Archive log data sets
- Bootstrap data sets (BSDS)
- Page sets

To reduce the risk of your backup copies being lost or damaged, consider these measures:

- Storing the backup copies on different storage volumes to the original copies.
- Storing the backup copies at a different site to the original copies.
- Making at least two copies of each backup of your page sets and, if you are using single logging or a single BSDS, two copies of your archive logs and BSDS. If you are using dual logging or BSDS, make a single copy of both archive logs or BSDS.

IBM MQ might need to use archive logs during restart. You must keep sufficient archive logs so that the system can be fully restored. IBM MQ might use an archive log to recover a page set from a restored backup copy. If you have discarded that archive log, IBM MQ cannot restore the page set to its current state. So, think before you discard archive logs!

For further details on MQ restart and recovery process, see:

https://www.ibm.com/support/knowledgecenter/en/SSFKSJ_9.1.0/com.ibm.mq.pro.doc/q004110_.htm

Persistent messages can be recovered in the event of a queue manager failure, and queue managers are resilient to failures in general. However, a single queue manager still represents a single point of failure. There are two ways to remove this single point of failure: IBM MQ clusters and IBM MQ queue sharing groups.

IBM MQ clusters

An IBM MQ cluster is a set of related queue managers that know how to connect to each other. Queue managers in the cluster can choose to mark a subset of their queues as cluster queues. Each queue manager in the cluster is automatically made aware of cluster queues and can route messages to them without needing manual configuration. As a result an MQ cluster reduces the amount of administration needed to manage a set of queue managers.

IBM MQ clusters can also provide resilience. Multiple queue managers in a cluster can contain a cluster queue with the same name. If an application sends messages to a queue with that name the cluster will use workload management to spread the messages across the individual instances of the queue in the cluster. If one of the queue managers that hosts the cluster queue fails, new messages can still be sent to the remaining queue managers that host the queue. This means that whatever backend service that is processing the messages on the cluster queue can still be reached.

However, existing messages on the cluster queue on the failing queue manager are only made available again when the queue manager is restarted. So IBM MQ clustering provides high availability of the queue and the backend service, but not the individual messages.

IBM MQ queue sharing groups

A queue sharing group is a set of cooperating z/OS queue managers that can connect to one or more coupling facilities (CFs). Queue managers in a queue sharing group can be configured with a special type of queue called a *shared queue*. A shared queue has all its

data stored in a list structure in a CF. This means that all queue managers in the group can access the queue at the same time and perform operations on the messages on the queue.

If one of the queue managers in the queue sharing group is unavailable, then applications can connect to another queue manager in the group and still access the same messages. In contrast, the normal, nonshared queues that have been discussed up to this point can be accessed only by their owning queue manager. If the queue manager is not available, neither are the messages. Therefore, queue sharing groups and shared queues provide true high availability of messages. And that is the most resilient form of messaging possible.

Role of IBM MQ for z/OS in the journey to exceptional resilience

Table 2-2 shows how the role of IBM MQ for z/OS deepens as your enterprise makes the journey toward higher availability. As stated previously, four deployment models encapsulate the key stages of this journey:

- Deployment model 1: Starting with a resilient and reliable base
- Deployment model 2: Reducing the duration of outages with failover capability
- Deployment model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS
- Deployment model 4: Adding GDPS Continuous Availability for exceptional resilience

The indentions of the following table emphasize how features that are located below an upper level inherit the features of that upper level.

Table 2-2 Role of IBM MQ for z/OS in the journey to exceptional resilience

Deployment model 1: Resilient / Reliable **Logging** - MQ records all significant changes to persistent data in a recovery log. Dual logging offers protection against data loss. **Archiving** - Logs automatically archived to secondary storage (tape or DASD).
Deployment model 2: Failover Capable The preceding list plus, **Data replication** - MQ active log data sets can be mirrored to a secondary storage subsystem using data replication. zHyperwrite support improves performance.
Deployment model 3: Fault Tolerant The preceding list plus, **Shared Queues** - Groups of queue managers are connected via a coupling facility. If a queue manager fails, messages on shared queues are available to other queue managers in the group. Automatic peer recovery enables other queue managers to complete any in-flight transactions started by the failed queue manager.
Deployment model 4: Continuous Availability The preceding list, plus, Asynchronous replication of queue group data to DR or secondary data center. Manual or automated recovery process is in place to bring up queue sharing group in DR or secondary data center.

2.4.2 IBM Db2 for z/OS

IBM Db2 for z/OS is still the undisputed leader for storing and harnessing mission-critical data. It remains the “gold standard” for system availability, scalability, security, and cost effectiveness. A majority of Fortune 500 companies, including the world's top banks, retailers, and insurance providers trust Db2 for z/OS for storing their data.

Db2 provides businesses the capabilities needed to meet rapidly changing business demands by way of:

- Providing scale and speed for next era of mobile and interconnected devices
- Supporting cloud, analytics, and mobile
- Delivering faster analytical insights
- In-memory database performance, reducing costs

Db2 is capable of effectively handling terabytes of data in both decision support and transactional environments. Db2 is a robust database platform for a variety of solutions and vertical applications, including:

- Back end for web and application servers
- Business Intelligence and Data Warehousing
- Transactional enterprise systems
- Enterprise applications, such as Enterprise Resource Planning (ERP), Customer Relationship Management (CRM), and Supply Chain Management (SCM)
- Information Integration and Content Management

Although high availability of data is a goal for all Db2 subsystems, unplanned outages are difficult to avoid entirely. A key to the perception of high availability is getting the Db2 subsystem restarted quickly after an unplanned outage. However, a good backup, recovery, and restart strategy can reduce the elapsed time of an unplanned outage. To reduce the probability and duration of unplanned outages, you should periodically back up and reorganize your data to maximize the availability of data to users and programs. Db2 relies on the log and the bootstrap data set (BSDS) to record data changes as they occur. The log and BSDS provide critical information during recovery. A number of utilities that Db2 provides also play an important role in the backup and recovery process. For example COPY, QUIESCE, MERGECOPY, and BACKUP SYSTEM can be used for backup purposes. RECOVER, REBUILD INDEX, REPORT, and RESTORE SYSTEM can be used for recovery.

A problem can occur with hardware or software. Damage can be physical or logical. Here are a few examples:

- If a system failure occurs, a restart of Db2 restores data integrity. For example, a Db2 subsystem or an attached subsystem might fail. In either case, after Db2 restarts, it backs out uncommitted changes, and completes the processing of committed changes.
- If a media failure (such as physical damage to a data storage device) occurs, you can recover data to the current point.
- If data is logically damaged, the goal is to recover the data to a point in time before the logical damage occurred. For example, if Db2 cannot write a page to disk because of a connectivity problem, the page is logically in error.
- If an application program ends abnormally, you can use utilities, logs, and image copies to recover data to a prior point in time.

Data Sharing

In Db2 data sharing, applications that use multiple Db2 subsystems in a Parallel Sysplex can read from and write to the same Db2 for z/OS data concurrently, with integrity, performance, scalability, and dynamic workload balancing. A collection of one or more Db2 subsystems that share Db2 data is called a data sharing group. Db2 subsystems that access shared Db2 data must belong to a data sharing group. A Db2 subsystem that belongs to a data sharing group is a member of that group. Db2 data sharing improves the availability of Db2 data, extends the processing capacity of your system, provides more flexible ways to configure your environment, and increases transaction rates. Db2 for z/OS uses special data sharing locking and caching mechanisms to ensure data consistency across the applications.

Db2 recovers data from information that is contained in both the logs and the bootstrap data sets (BSDSs) of members. However, because updates can be logged on several different members, Db2 coordinates recovery by using the shared communications area (SCA) in a coupling facility. The SCA is also used to coordinate startup. You can stop and start an individual member of a data sharing group while the other members continue to run. The startup process for each member is similar to that of non-data sharing Db2 subsystems. Db2 uses a process that is called group restart in the rare event that critical resources in a coupling facility are lost and cannot be rebuilt. When this happens, all members of the group terminate abnormally. Group restart rebuilds the lost information from individual member logs. However, unlike data recovery, this information can be applied in any order. Because there is no need to merge log records, Db2 can perform many of the restart phases for individual members in parallel.

Restart Light

Imagine a situation in a Db2 data sharing environment where a z/OS system has experienced an unplanned outage. The Db2 subsystem that is running at the time of failure might be holding locks on shared resources. These resources that have a retained lock will not be accessible from other subsystems. Therefore, it becomes extremely critical to restart the failed Db2 subsystem on another z/OS image in the same sysplex to remove the retained locks. It can be a common scenario that the other z/OS image — already running its own workload — does not have enough resources to restart the failed Db2 subsystem and handle additional workload. With Restart Light, you can perform a light startup of the failed Db2 subsystem. Restart light mode does the following operations:

- Minimizes the overall storage that is required to restart the member.
- Removes retained locks as soon as possible, except for the following locks:
  - Locks that are held by postponed-abort units of recovery, if the LBACKOUT subsystem parameter is set to LIGHT or LIGHTAUTO.
  - IX and SIX mode page set P-locks, if LIGHT(YES) or LIGHT(NOINDOUBTS) is specified. These locks do not block access by other members. However, they do block drainers, such as utilities.
- Terminates the member normally after forward and backward recovery is complete. No new work is accepted.
- By default, INDOUBT threads are not automatically resolved when Db2 is participating in a two-phase commit processing. In such a scenario, Db2 does not terminate until the INDOUBT thread(s) are manually resolved by committing or aborting the INDOUBT thread.

  In an environment where two-phase commits are present, Restart Light (INDOUBTS) can allow Db2 to perform a faster restart and shut down without having to manually resolve the INDOUBT threads.

Restart light mode is intended only for a cross-system restart of a system that does not have adequate capacity to sustain the Db2 and IRLM in the event of a failed z/OS system.

Role of IBM Db2 for z/OS in the journey to exceptional resilience

Table 2-3 shows how the role of IBM Db2 for z/OS deepens as your enterprise makes the journey toward higher availability. As stated previously, four deployment models encapsulate the key stages of this journey:

- Deployment model 1: Starting with a resilient and reliable base
- Deployment model 2: Reducing the duration of outages with failover capability

- Deployment model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS
- Deployment model 4: Adding GDPS Continuous Availability for exceptional resilience

The indentions of the following table emphasize how features that are located below an upper level inherit the features of that upper level.

Table 2-3 Role of IBM Db2 for z/OS in the journey to exceptional resilience

Deployment model 1: Resilient / Reliable **Recovery -** Db2 image copy and log based recovery with backup on tapes or DASD. **Manual restart** of Db2 subsystem. **Db2 Restart Light (INDOUBTS)** - For faster restart and shut down.
Deployment model 2: Failover Capable The preceding list plus, **Data replication**. **Dynamic Resource Definition.** **Restart** - Automated Db2 subsystem restart.
Deployment model 3: Fault Tolerant The preceding list plus, **Db2 data sharing**. **Cross system restart** - cross system automated Db2 subsystem restart.
Deployment model 4: Continuous Availability The preceding list plus, **Db2 restart light**. **GDPS CA** for near zero RTO and RPO.

2.5 Application layer

An application is software that is designed to perform a group of coordinated functions, tasks, or activities for users or other applications. An application can be self-contained logic or a group of programs. Applications use the operating system and other supporting programs, such as middleware to function. An application requests services from and communicates with other applications, middleware, or users via an application programming interface (API). Common IBM Z application workloads include, transaction processing, batch processing, bulk data processing, and so on.

For end users, the discussion of applications might cover the role of each application in your environment. However, in the journey to resiliency, the following broader question is foremost:

How can we manage a myriad of critical activities by so many applications?

Over the years, two IBM products have provided answers to this question:

- IBM Customer Information Control System (CICS)
- IBM Information Management System (IMS)

These IBM applications evolve over time to address the ever-changing and expanding capabilities of the modern enterprise. If an application could speak, it might say:

- I need CICS to manage my transactions with users.
- I need IMS to manage my transactions with data repositories.

The journey to resilience for some companies also includes IBM WebSphere Application Server on z/OS. This product extends the management capabilities into a Java Platform, Enterprise Edition implementation that helps support applications at an API level.

Although batch workloads are common in a IBM Z environment, they are often forgotten. In most cases, timely completion of batch jobs is as mission critical as online transactions.

In this section the resiliency capabilities for CICS, IMS, WebSphere Application Server on z/OS, and batch processing are discussed.

2.5.1 Customer Information Control System

CICS Transaction Server (CICS TS) is the world's most advanced mixed language application server. CICS TS exploits the IBM Z ecosystem to run critical business applications in a secure, transactional, multi-user environment. Large enterprises use CICS to deliver their applications' functions with exceptional capacity, performance, and availability.

CICS TS provides services for applications including:

- A highly efficient and optimized environment for running transactions. CICS TS securely manages concurrency, sharing of resources, integrity of data, and task prioritization.
- Support for business applications written in COBOL, C, C++, Java, assembler language, PL/I, REXX, PHP. These applications use an application programming interface that provides access to CICS services.
- Support for Java EE applications, using dedicated CICS regions that run the Liberty server within a CICS JVM server.
- CICS application program data can reside in Db2 and IMS databases, as well as traditional hierarchical files and record-oriented VSAM and BDAM data sets.
- Inter-operation with IBM MQ. CICS application programs can access message queues by using the IBM MQ API. Java applications can access the queues by using JMS.
- Inter-operation with Web Services including SOAP Web Services and JSON Web Services
- Distribution of work between multiple CICS regions in both a z/OS sysplex and non-sysplex environment.

CICS TS provides a wide range of connectivity options, including:

- Direct connectivity to end users through TCP/IP and to traditional terminal equipment through SNA.
- Connectivity with non-CICS systems in client/server and peer-to-peer configurations. This can use protocols such as HTTP, SOAP Web services and JSON Web services, and asynchronous messaging.

CICS TS also provides administrative services including:

- Interfaces for configuring and managing CICS regions.
- Aids for debugging application programs and for diagnosing system problems.

As well as using services provided by the z/OS operating system layer (see 2.3.1, "z/OS" on page 32), CICS systems use other IBM Z software such as:

- Db2, which provides relational database services, including SQL access (see 2.4.2, "IBM Db2 for z/OS" on page 42)

- IMS DB, which provides hierarchical database services, including DLI access (see 2.5.2, “Information Management System” on page 62)
- IBM MQ, which provides asynchronous messaging services, MQI access, including JMS access for Java applications, and message-driven interfaces (see 2.4.1, “IBM MQ for z/OS” on page 39)

Thus, CICS systems benefit from running on an exceptionally resilient platform. And CICS builds on the platform, by adding its own adding resiliency features. The following subsections provide more information about CICS resilience.

Single-region CICS resilience features

The largest CICS configurations are CICSplexes, which are clusters of CICS regions that work cooperatively. But many installations have single-region CICS systems.

Even a single-region CICS system typically runs many different application programs. Together, they serve many different users — people and systems — that generate large numbers of input messages to transactions, as many as thousands of input messages a second.

Obviously, such systems demand a high level of resilience. In this section we describe how one CICS region can achieve that. Later we describe how CICSplexes combine multiple CICS regions into systems that meet even the highest resilience requirements.

Non-disruptive deployment of application updates

A CICS system typically runs many different application programs that interact with each other and with the corporate data in complex ways. Inevitably, changes can occur that reveal errors or omissions in the application code. When this happens, the enterprise must develop and install code fixes to correct the problem. Changes to the business environment, innovation, regulations, and so on can also require updates to existing application programs.

From a resilience perspective, it must be possible to deploy these updates to a production system with the little or no disruption to users of the services that the system provides.

SET PROGRAM command

The CICS command SET PROGRAM enables non-disruptive updates as follows:

- SET PROGRAM with the PHASEIN option ensures that any transactions that are already running continue to use the old (unmodified) copy of the program. Only new transactions use the new (updated) version. This avoids problems that might occur when a transaction calls the program more than once by ensuring that all these calls use the same copy of the program.
- Alternatively, SET PROGRAM with the NEWCOPY option ensures that different transactions that run at the same time all use the same copy of the program. For this, CICS continues to use the old version until a moment when no transactions are using the program. Then, it starts to use the new version.

SET BUNDLE command

The CICS command SET BUNDLE provides a similar capability for non-disruptive phase-in of updated Java applications deployed in OSGi bundles.

Transactional integrity

The atomicity, consistency, isolation, and durability (ACID) criteria for transactions are at the heart of resilient business computing. Large numbers of ACID-compliant transactions can use and update the same business-critical databases safely. For more information see:

https://www.ibm.com/support/knowledgecenter/en/SSGMCP_5.4.0/product-overview/acid.html

CICS, IMS (see 2.5.2, “Information Management System” on page 62), and WAS (see 2.5.3, “IBM WebSphere Application Server for z/OS” on page 73) provide the infrastructure for ACID-compliant transactional behavior, relieving application programs of that responsibility.

To understand how ACID-compliance impacts the resilience of these systems, consider that a single business transaction typically comprises more than one step that updates data in business-critical databases. Different steps can involve different “players” (for example CICS, Db2, external microservices, and so on). And some steps can run parallel (that is, transaction might start a processing step without waiting for the previous steps to complete).

Clearly, there are many things that might cause one or more steps to fail before the transaction completes. If the transaction is not ACID-compliant, this can result in inconsistencies in databases or between databases, or both. These inconsistencies might cause other transactions to malfunction or fail. But even if the initial problems do not cascade, the inconsistencies are likely to be a problem for the enterprise that ACID-compliance can address.

With ACID-compliance:

When a transaction fails…

When a CICS transaction fails, CICS limits the damage to just the failing transaction.

It discards the updates from any partially completed steps (rollback). The databases are not damaged by the failure. Other transactions continue without significant impact.

CICS also notifies other systems that perform updates as part of its transactions. They too discard updates for the failed transaction. Other databases are not damaged by the failure.

When CICS fails…

When a CICS system fails, CICS limits the damage to just the transactions that run at the time of the failure.

When it restarts, it discards the updates from partially completed transactions (rollback) and keeps the updates from completed transactions. The databases are not damaged by the failure. Other transactions can run without significant impact after the CICS restart.

CICS also notifies other systems that perform updates as part of its transactions. They too discard updates for the partially completed transactions. Other databases are not damaged by the failure.

Storage protection

Errors in application code that are not detected during testing can cause the code to store data in the wrong place. This corrupts data that other transactions or even CICS itself might be using and causes failures or malfunctions. CICS uses IBM z/Architecture® memory capabilities to protect data in memory against this type of corruption — and to locate and help fix the errors — as follows:

Protecting executable code

By default, CICS loads reentrant programs (programs that do not modify themselves) into key-0 memory. Application programs cannot modify key-0 memory. This improves resilience because any change to reentrant code is a mistake (a program that intentionally modifies itself is not reentrant) and is very likely to cause malfunctions or failures.

CICS configuration settings (see
https://www.ibm.com/support/knowledgecenter/SSGMCP_5.5.0/tuning/dfht3_storage_p

rotection.html) can override this default for a region so that CICS loads reentrant application executables into memory that they can modify. This option is available for regions that are used for application development where it is required by some debuggers. It is not recommended for CICS regions that run production work.

Protecting critical CICS data

CICS uses memory that application programs cannot modify (that is, key-0 memory) for some critical data.

Protecting all data owned by CICS

By default, CICS allocates two types of memory for data:

CICS-key storage

For data owned by CICS, applications can read CICS-key storage. But only CICS can modify it. This improves resilience because it protects data that is owned by CICS against any application that accidentally tries to modify data that it should not modify.

User-key storage

For transaction data, both application programs and CICS can read and modify user-key storage.

CICS configuration settings can override this default for a region so that all transactions in the region use CICS-key storage. This allows the programs to modify CICS data. This option is available for CICS regions that do not run business logic. It is not recommended for CICS regions that do.

In regions where the protection is enabled (the default), transaction settings can specify that some transactions run with CICS-key. This is available, for example, for transactions that administer CICS functions. It is not recommended for transactions that run business logic.

Protecting storage that is owned by one transaction from modification by code that runs other transactions

Optionally, CICS uses the z/Architecture subspace group facility to prevent programs that run one transaction from accidentally overwriting the data of another transaction. This is called *transaction isolation*.

Transaction isolation improves resilience because programs do not normally modify data that is owned by other transactions. When they do, it is usually a mistake that is very likely to cause malfunctions or failures.

But transaction isolation does have a performance cost and some installations prefer to avoid this by disabling transaction isolation. Installations that choose to do this can still benefit by thoroughly testing applications on systems that have the facility enabled. This can help identify and remove resilience exposures that would otherwise impact production.

By default, transaction isolation is disabled but CICS configuration settings can override this default for a region. When this option is enabled, transactions default to running with transaction isolation. This protects them from each other.

But some installations have existing CICS application code that does not work when transaction isolation is enabled. For these programs, transaction settings can selectively disable transaction isolation. Those transactions are not protected from each other.

Also, transactions that access VSAM NSR files in update mode do not work when transaction isolation is enabled. Again, transaction settings can selectively disable transaction isolation. But a better alternative is to use VSAM RLS.

Automation

For good resilience, it is important to recover as quickly as possible when a CICS region fails or a region that CICS is using fails. Automation can dramatically reduce delays between a failure and the response and reduce the risk of inappropriate “panic” responses. Specific CICS automation options include:

Automatic restart manager (ARM)

CICS regions register with ARM automatically as part of CICS system initialization. When a CICS region fails, ARM can restart it automatically.

For more complex failures that involve several regions, possibly with interdependencies, installations can specify ARM policies to “orchestrate” restarts. For example, when CICS regions depend on Db2 or MQ, ARM can restart those regions before it restarts the CICS regions that use them. When regions do not have dependencies, ARM can restart them in parallel and so minimize the overall time to recovery.

A CICS region that has not failed automatically reconnects when ARM restarts a region — such as Db2 — that the CICS region was previously using.

CICS policies

Installations can use CICS policies to write a message or drive automation code via a CICS event. The policy might be invoked when a task exceeds a resource-usage threshold or when a specified change in system state occurs. CICS supports two types of policy rules:

System rules

Can monitor the state of critical system resources, or the overall health of a CICS system, and respond automatically when any changes occur. System rules support two actions: message or event. For more information about the system rules that are supported, see Policy system rules.

Task rules

Can monitor the resource utilization of individual user tasks and respond automatically when the resource usage of a task exceeds a predefined threshold. In this way, excessive resource usage and looping transactions can be detected and dealt with appropriately. Task rules support three actions: message, event, or abend. For more information about the task rules that are supported, see Policy task rules.

System Automation for z/OS (SA z/OS)

Many CICS installations use IBM System Automation for z/OS (SA z/OS) to implement automated responses to incidents that CICS reports in console messages (see 2.6.1, “IBM System Automation for z/OS” on page 82).

MQMONITOR

CICS applications that process MQ messages are often designed to run if and only if CICS is connected to an MQ queue manager. A typical example is a long-running application that “monitors” a queue. That is, it waits for messages to arrive on the queue and initiates processing for each message that arrives. (CICS itself includes the CKTI and MQBR transactions that do this.)

The CICS MQMONITOR facility can automate starting and restarting this type of application when CICS connects to the queue manager or reconnects to it after a queue manager outage.

Multi-region CICS (CICSplex) resilience features

Depending on the requirements of the application, a single region that runs CICS TS can be adequate. But multiple CICS regions that work cooperatively can deliver better resilience —

as well as greater capacity — than a single region. A group of CICS regions configured in this way is called a CICSplex.

z/OS provides isolation between different regions even in the same operating system image. For example, different regions can have different resources, priorities, and so on. And when a region fails, other regions can continue. In this section, we describe how a CICSplex exploits this isolation to deliver enhanced resilience.

Note: CICSplexes exploit the very high-performance communication options that are available between z/OS regions. CICS multi-region operation (MRO) meets this requirement by using z/OS shared memory protocols for communication between CICS regions in a single-LPAR CICSplex.

Specialized regions and region cloning

To maximize the benefits of a CICSplex, installations can assign different roles to different regions. In an example configuration, some regions, often called terminal-owning regions (TORs), act as communication front-ends. They receive inbound work for the CICSplex and pass the work to regions often called application-owning regions (AORs) that run the application programs that implement the business functions. In some configurations, AORs delegate access to underlying corporate data through dedicated regions, often called data-owning regions (DORs) or file-owning regions (FORs).

As well as assigning these and other roles to different regions, installations can assign different AORs to run application functions with different resource requirements or service level obligations. An example is that many CICS installations prefer to run Java applications in specialized AORs — often called Java-owning regions (JORs).

The Figure 2-5 shows (schematically) a CICSplex that includes TORs, AORs, and FORs.

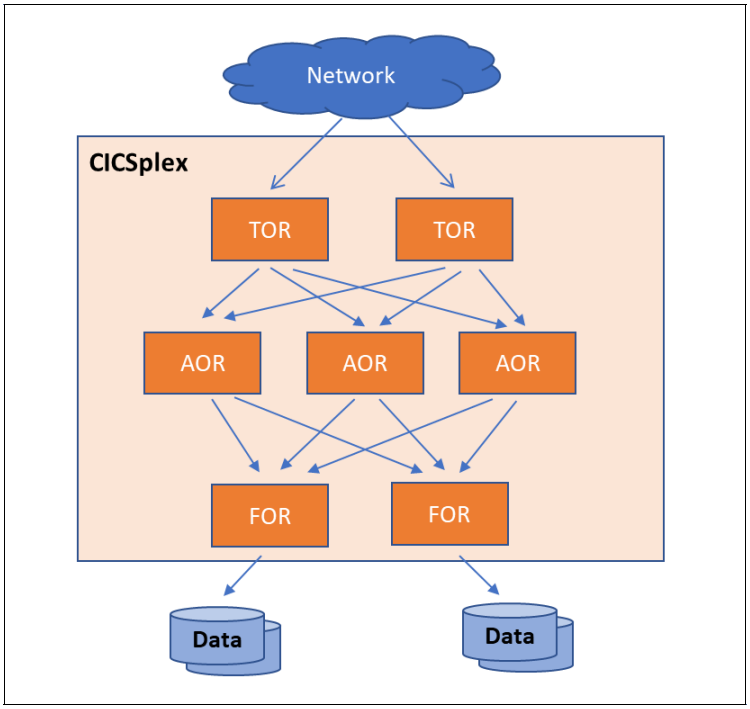


Figure 2-5 Specialized CICS regions in a CICSplex

Connecting multiple CICS regions to communication resources

CICS systems connect to communication resources that can include:

- Communication networks that connect end users or other systems
- HTTP servers
- Asynchronous messaging systems such as IBM MQ
- Private communication networks and links that connect specialized or legacy equipment

This section identifies extra issues that affect a CICSplex when there are multiple CICS regions that must be connected rather than a single-region CICS configuration.

Inbound communication resources

Inbound communication resources deliver work to CICS and deliver responses from CICS. In a typical CICSplex, dedicated regions (TORs) “own” the connections to inbound communication resources. TORs enhance resilience directly by routing inbound work to alternative AORs when an AOR fails or is too busy, and by handling recovery when connections to communication resources fail. Installations can use multiple TORs either to handle different types of resource or to safeguard against a TOR failure or both.

As an alternative, each AOR can have its own connections to inbound communication resources.

Outbound communication resources

Applications that access outbound communication resources typically establish their own connections to the communication resources and appear as unique endpoints to the target programs that receive the messages.

TCP/IP port sharing

For TCP/IP communication, z/OS TCP/IP Communications port sharing allows multiple CICS regions in a single LPAR to appear as a single endpoint.

In a CICSplex that uses TORs, multiple TORs can be configured to share the same port. The TCP/IP subsystem distributes inbound connection requests to the TORs. This safeguards against TOR failures. When a TOR fails, other TORs continue to service inbound connection requests and continue to distribute the work to the AORs.

In a CICSplex where each AOR has its own connections to inbound communication resources, the AORs can be configured to share the same port. The z/OS TCP/IP Communications subsystem distributes inbound connection requests to the AORs. This safeguards against AOR failures. When an AOR fails, the TCP/IP subsystem continues to distribute inbound connection requests to the surviving AORs.

Port sharing can be configured to distribute inbound connections by using a simple round-robin of the CICS regions via the shared port or by using information from WLM to select between the regions.

MQ input queues

MQ message producers send messages to queues. Consumers get inbound messages from queues. MQ allows multiple consumers that use the same queue manager to get messages from the same queue.[8] Originators can send messages to one target queue where multiple consumers get and process them.

In a CICSplex that uses TORs (in this case, sometimes called “QORs”) for MQ input, multiple QORs can be configured to get messages from the same queue or queues. This

[8] Do not confuse this with MQ shared-queue support. A single MQ queue manager allows multiple connectors to the same queue manager to access the same queue for input (**get**) processing. This is enough for a single-LPAR CICSplex. z/OS MQ shared-queue support allows connectors to different queue managers to access the same queue for **get** processing. It is required for a multi-LPAR CICSplex.

safeguards against QOR failures. When a QOR fails, other QORs continue to get messages from the input queue or queues and continue to distribute the work to the AORs.

In a CICSplex where each AOR has its own queue manager connection, the AORs can be configured to get messages from the same queue or queues. This safeguards against AOR failures. When an AOR fails, surviving AORs continue to get and process messages from the same queue or queues.

Connecting multiple CICS regions to data resources

CICS application programs access data resources including SQL and DL1 databases, VSAM files, and zFS files. In a single-region configuration, CICS performs an application's data updates within a CICS unit of work. It uses internal locks to prevent different transactions from updating the same data item concurrently. And it exploits in-memory data caching to optimize performance.

For SQL databases, CICS uses Db2 to perform the data accesses. For DLI, it uses IMS DB. Db2 and IMS DB are two-phase commit capable resource managers. They provide their own transaction management, fine-grained locking, and data caching. This allows multiple CICS regions to access the data without compromising data integrity.

For VSAM data, CICS itself provides transactional function such as commit, rollback, and forward recovery logging for data that requires these capabilities — that is, for data sets with the CICS recoverable attributes. But multiple CICS transactions can access the same VSAM data set at the same time. This requires a locking mechanism to prevent one transaction from updating a record that another transaction is using or updating. This is called record-level locking.

In a single-region configuration, CICS file control provides record-level locking. A multi-region configuration requires one of the following to provide record-level locking:

- A file-owning region (FOR)
- VSAM record-level sharing (RLS)

VSAM data — file owning regions (FORs)

An FOR "owns" the VSAM data and AORs use it to access the VSAM data. Notice that restrictions apply to transactions that use VSAM and transaction isolation[9].

Figure 2-6 on page 54 shows CICS regions configured to use an FOR.

[9] CICS does not support VSAM non-shared resources (NSR) with transaction isolation.

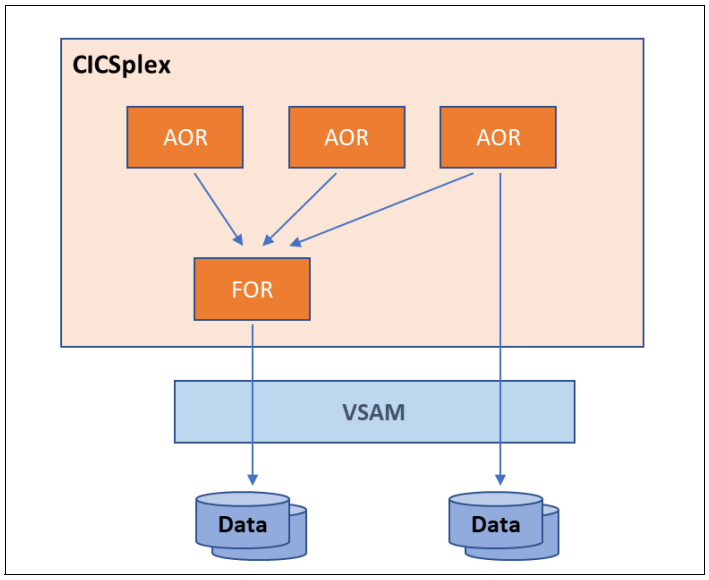


Figure 2-6 Using a CICS FOR to access VSAM data

VSAM data — record-level sharing (RLS)

VSAM RLS eliminates the need for an FOR, which could be a single-point-of-failure and a potential performance bottleneck. And it avoids the restrictions that apply to transaction isolation.

VSAM RLS is a two-phase commit capable resource manager. It uses the IBM Coupling Facility (CF) for its cache and lock structures. It also supports a common set of buffer pools for each z/OS image.

The Figure 2-7 shows CICS regions configured to use VSAM RLS.

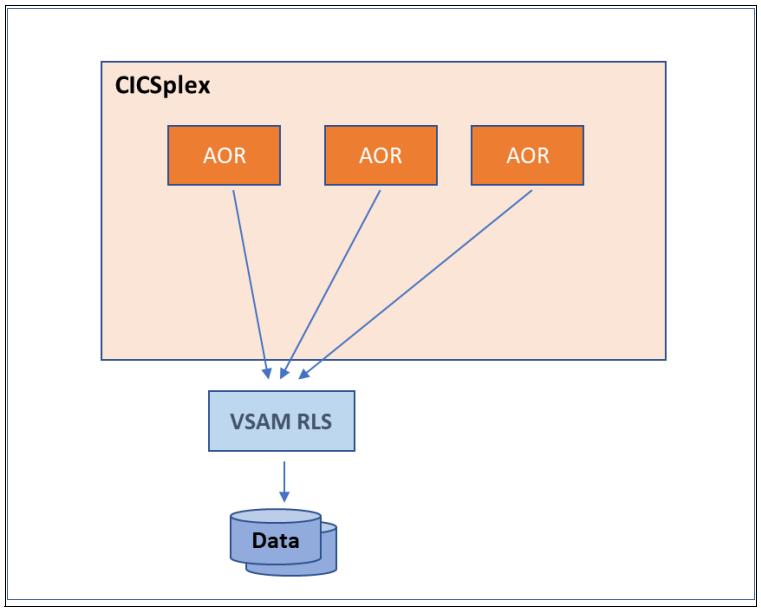


Figure 2-7 Using VSAM RLS to access VSAM data

Coordinating multiple CICS regions to enhance resilience — CICSPlex SM

IBM CICSPlex® System Manager (CICSPlex SM) is the CICS component that coordinates the regions in a CICSplex. CICSPlex SM itself runs in specialized regions, such as the CMAS and WUI servers, that are part of the CICSplex cluster.

The Figure 2-8 shows how CICSPlex SM regions connect with the other specialized regions in the CICSplex.

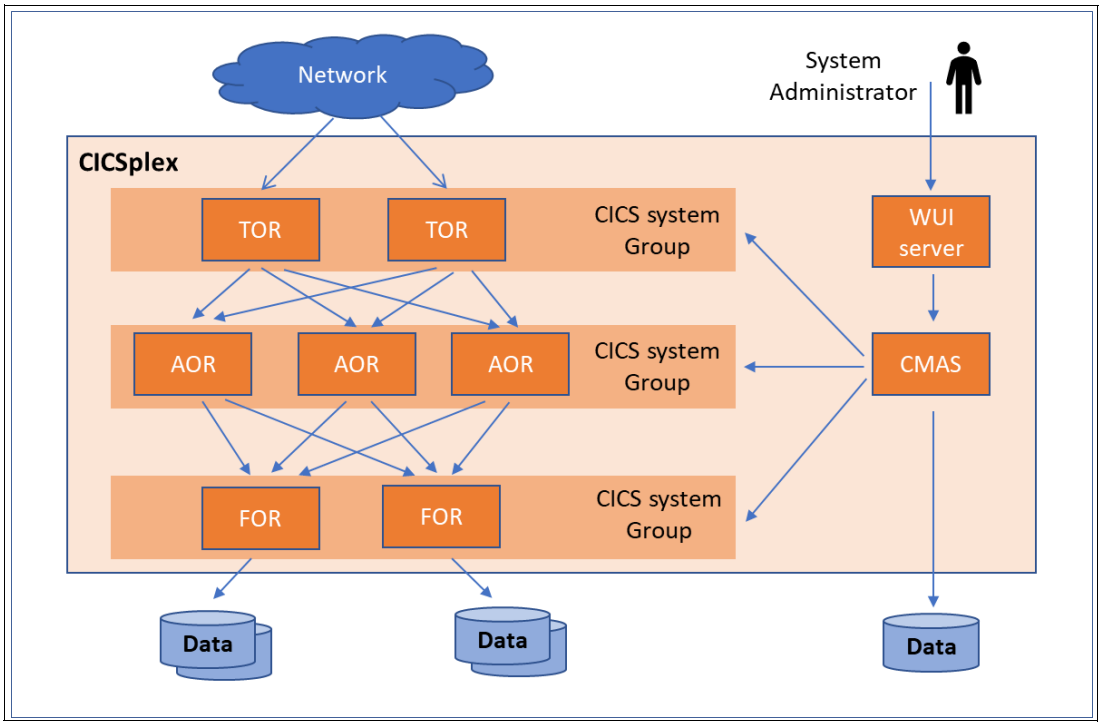


Figure 2-8 A CICSplex with CPSM regions

CICSPlex SM provides many functions, including:

Workload management

CICSPlex SM dynamic routing is particularly important for resilience. It monitors the state of the regions and connections in the CICSplex and uses this information — together with rules and policies configured for the specific requirements of the installation — to optimize the distribution of work within the CICSplex.

This allows the CICSplex to respond automatically when a region becomes unresponsive or fails completely — because of excessive load, resource contention, application error, processor failure, or other unexpected conditions — by routing work to a different region.

Non-disruptive deployment of application updates

CICSPlex SM can orchestrate distribution of CICS commands to multiple CICS regions. As an example, this facility can be used to orchestrate deployment of application updates across an entire CICSplex or across a subset of the regions (such as the AORs) in a CICSplex.

Managing inter-transaction affinities in a CICSplex

CICS application programs use many techniques to pass information between transactions and to synchronize activity between them. Some of these techniques require the transactions that are exchanging data to execute in the same CICS region. To exploit CICSplex effectively, it is necessary to identify these dependencies in the application code and establish dynamic

routing rules. In this way, you ensure that transactions that must run in the same region do not get routed to different regions.

The CICS Interdependency Analyzer (CICS IA) is a tool that can help you understand the relationships, dependencies, and flows of CICS TS applications. It has three components:

Dependency collector

Intercepts CICS system commands and commands that can create affinities. The collector also records the details of the resources that are used.

Scanner

Scans the load-module data sets that show information, for example, the programming language and the CICS commands that are included in the program. Scanner queries show potential CICS command usage, while dependency queries show actual usage. Other queries enable comparison of potential versus actual API usage, which identifies possible dead code or missing test cases.

Command Flow feature

Captures all CICS, Db2, IMS, and IBM MQ commands in chronological order. Also captures a wide range of related information including current and previous TCB ID, response and reason codes, times of day, and CSECT offset. It identifies resources that are used by a specific instance of a transaction for threadsafe analysis. This information can help you to understand the flow and structure of umbrella transactions or application entry points.

The CICS IA generates both human- and machine-readable output. The machine-readable output can be directly fed into CICSPlex SM to manage the transactions.

Multi-LPAR (sysplex) CICS resilience features

A CICSplex integrates multiple CICS regions into a cluster. As described in the previous section, this provides resilience in several ways. For example:

- A CICSplex can survive the failure of one or more regions by rerouting work to surviving regions.
- A busy region might provide a degraded quality of service, or even fail if it becomes overloaded. A CICSplex can mitigate this by rerouting work from busy regions to less busy ones.
- A CICSplex can run different functions with different requirements in separate regions. This can improve resilience by limiting the impact of a region failure to just those functions that are running in that region.

A CICSplex can deliver these resilience benefits even when all the regions are in a single LPAR. But a CICSplex can also integrate regions in different LPARS into the same cluster. This can be used to increase the capacity of a CICSplex. It also extends the resilience benefits as follows:

- A multi-LPAR CICSplex can survive the failure of one or more LPARs by rerouting work to regions in surviving LPARs.
- A busy LPAR might provide a degraded quality of service, or even fail if it becomes overloaded. A multi-LPAR CICSplex can avoid this by rerouting work from regions in busy LPARs to regions in less busy ones.
- A multi-LPAR CICSplex allows rolling IPLs (see 2.6.1, “IBM System Automation for z/OS” on page 82) to replace or upgrade critical system software — including the z/OS operating system — and hardware. The rolling IPLs do not interrupt the services that are provided by CICS applications that are running in the CICSplex.

Note: CICSplexes exploit the very high-performance communication options available between z/OS regions. Coupling-facility protocols — in particular, XCF for MRO — meet this requirement for communication between CICS regions in different LPARs in CICSplex within one z/OS sysplex.

Connecting CICS regions in multiple LPARs to communication resources

This section identifies extra issues that affect connecting communication resources in a CICSplex when there are multiple LPARs that must be connected rather than just one in a single LPAR CICS configuration.

TCP/IP Sysplex Distributor

In a single LPAR CICSplex, z/OS TCP/IP Communications port sharing allows multiple CICS regions to appear as a single endpoint. The Communications subsystem distributes inbound connection requests across regions using the same (shared) port.

Sysplex Distributor (a component of IBM z/OS Communications Server) provides TCP/IP load balancing across multiple LPARs. It can be combined with TCP/IP port sharing. Sysplex Distributor combines the high-availability features of distributed DVIPA and the workload optimization capabilities of WLM.

Sysplex Distributor enables the resilience benefits of a multi-LPAR configuration for CICS by safeguarding against an LPAR failure and by allowing changes. For example, you might need operating system upgrades on one LPAR while one or more other LPARs continue to provide service.

MQ shared-queue support

In a single LPAR CICSplex, multiple CICS regions connect to the same queue manager and can all access the same queues for input (**get**) processing.

z/OS MQ shared-queue support allows CICS regions in different LPARs to connect to different queue managers in the same queue-sharing group (QSG) to access the same queues for **get** processing. In a multi-LPAR CICSplex, a CICS region in one LPAR can connect a queue manager in the same LPAR. CICS regions in different LPARs must connect to queue managers only in the same queue-sharing group so that they can all get messages from the same queues.

MQ shared-queue support enables the resilience benefits of a multi-LPAR configuration for CICS. It safeguards against an LPAR failure and allows changes such as operating system or queue manager upgrades on one LPAR while one or more other LPARs continue to provide service.

MQ shared queues have sophisticated transaction support that provides resilience benefits. For these queues, MQ stores both messages and transaction state information on the coupling facility. When a queue manager fails, another queue manager in the same QSG can access this state information and attempt to resolve the failed queue manager's transactional updates to shared queues. That is, it can back out the updates for in-flight transactions and complete commits. MQ group UR support adds the ability to resolve two-phase commit transactions. Transactions can be resolved without waiting for the failed queue manager to restart.

CICS application programs access data resources including SQL and DL1 databases, VSAM files, and zFS files. For SQL databases, CICS uses Db2 to perform the data accesses. For DLI, it uses IMS DB. For VSAM, CICS uses either FORs or VSAM RLS.

Db2, IMS, zFS, and VSAM RLS are all two-phase commit capable resource managers and can all share access to the same data between manager instances in different LPARs. In a multi-LPAR CICSplex, the CICS regions in any one LPAR connect to and use the resource manager instance in that LPAR.

VSAM data — file owning regions (FORs) and VSAM RLS

Multiple CICS regions — in the same or different LPARs — can provide application access from multiple AORs to the same VSAM data in either of the following ways:

- Using FORs. All accesses to one VSAM data set must use the same FOR to ensure data consistency. An AOR in a different LPAR from the FOR must use a communication link with the FOR to request the access and transfer the data. Using a CICS FOR to access VSAM data from multiple LPARs shows CICS regions in different LPARs configured to use an FOR.
- Using VSAM RLS. Each LPAR contains its own RLS resource manager (SMSVSAM address space) that CICS AORs in the LPAR use. Using VSAM RLS to access VSAM data from multiple LPARs shows CICS regions in different LPARs configured to use VSAM RLS.

VSAM RLS provides significant advantages in single-LPAR configurations but these are even more important in multi-LPAR configurations (see Figure 2-9).

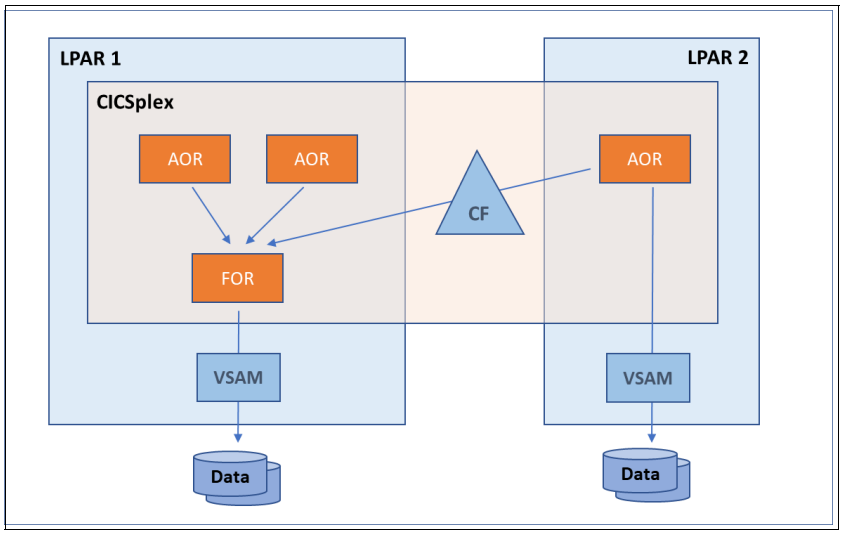


Figure 2-9 Using a CICS FOR to access VSAM data from multiple LPARs

In particular, VSAM RLS avoids the more complex and expensive inter-LPAR communications required for the FOR solution (see Figure 2-10 on page 59).

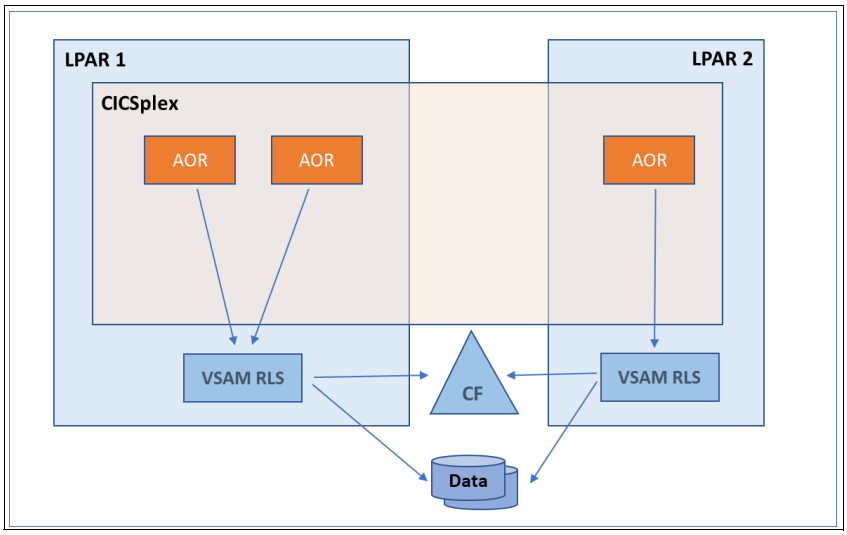


Figure 2-10 Using VSAM RLS to access VSAM data from multiple LPARs

VSAM data — coupling facility data tables

CICS can (optionally) use z/OS data spaces to provide a form of in-memory caching for VSAM KSDS. This caching is called shared data tables. Shared data tables use FORs. It can deliver significant performance benefits over native VSAM.

However, for a multi-LPAR configuration, better resilience can be achieved by using coupling facility data tables (CFDTs). Each LPAR connects to the CFDT using its own CFDT server region. This avoids the FOR, and the LPAR that it runs in, as single points of failure.

Coordinating CICS regions in multiple LPARs to enhance resilience — CICSPlex SM

CICSPlex SM offers essentially the same facilities for coordinating CICS regions in all regions of a multi-LPAR CICSplex as it does for all the regions of a single-LPAR CICSplex, for example:

- Management and monitoring of the CICS regions
- Management of the CICS resource definitions
- Deployment and management of CICS applications

These capabilities exploit direct links — typically XCF for MRO — between the CICSPlex SM CMAS regions in the different LPARs.

Link-neutral routing

When a front-end region such as a TOR receives a work request, it passes the request to an available AOR for processing. The preferred mechanism for selecting the target AOR is CICSPlex SM dynamic routing. Selection criteria can include:

- Affinities or other dependencies that limit the choice to one AOR or group of AORs
- The current task load of each AOR
- The health state of each AOR
- The type of connection between the front-end region and the AOR.

The last of these criteria (the type of connection) is significant in a sysplex that includes more than one central processor complex (CPC). Its effects include routing work that is received by a front-end region preferentially to AORs in the same CPC. Depending on

details of the configuration, this can result in highly asymmetric workload distribution between CPCs.

When a more symmetric workload distribution is required — for resilience or for other reasons — CICSPlex SM can be configured to use a link-neutral routing algorithm. For more details, see:
https://www.ibm.com/support/knowledgecenter/en/SSGMCP_5.5.0/fundamentals/wlm/wlm-algorithms.html

Sysplex-optimized workload management

CICSPlex SM dynamic routing uses information about the status of the regions in the CICSplex to manage workload distribution across the regions. For optimal resilience, this dynamic routing must react quickly when the status of a region changes. CICSPlex SM can achieve this by maintaining region status information in a coupling facility structure that CICS regions update and that CICSPlex SM can interrogate.

This mechanism is called sysplex-optimized workload management. It is described in more detail in:
https://www.ibm.com/support/knowledgecenter/en/SSGMCP_5.5.0/configuring/parallel-sysplex/cpsm_sysplex_optimized_wload_management.html

Multi-data center CICS (GDPS) resilience features

As described in the previous section, by using sysplex technology a multi-LPAR CICSplex can deliver exceptional resilience, including:

- It can survive LPAR failures
- It can eliminate the impact of overloaded LPARs
- It allows replacement or upgrades of critical system software — including the z/OS operating system — and hardware.

For all of these, a multi-LPAR CICSplex can continue to deliver the services that the business applications provide to end users and other connected servers with no noticeable interruption.

By deploying CICSplexes at more than one site, it is possible to provide a service that can survive even major incidents such as natural disasters, through a GDPS site switch:

GDPS Metro

To achieve the highest levels of availability and minimize the recovery for planned and unplanned outages, various clients deployed GDPS Metro multi-site workload configurations. However, the signal latency between sites can affect online workload throughput and batch duration. This issue results in sites typically being separated by no more than approximately 20 km (12.4 miles) fiber distance.

GDPS Continuous Availability (GDPS CA)

The GDPS CA concept consists of having two sites that are separated by virtually unlimited distances, running the same applications, and having the same data to provide cross-site workload balancing and continuous availability and disaster recovery. This change represents a fundamental paradigm shift from a failover model to a continuous-availability model.

Refer to IBM GDPS: An Introduction to Concepts and Capabilities, SG24-6374:
http://www.redbooks.ibm.com/abstracts/sg246374.html?Open

Multi-site Workload Lifeline

IBM Multi-site Workload Lifeline enables intelligent load balancing of TCP/IP workloads across two sites at unlimited distances for near-continuous availability. It also helps with planned outages by rerouting workloads from one site to another without disruption to users.

Refer to *The Value of Active-Active Sites with Q Replication for Db2 for z/OS An Innovative Customer Experience*, REDP-5140:
http://www.redbooks.ibm.com/abstracts/redp5140.html?Open

IBM InfoSphere Data Replication for Db2

InfoSphere Data Replication for DB2® for z/OS, V11.4.0 provides a single offering to address a wide variety of data replication requirements. The requirements include continuous data availability (whether local or remote), dynamic data integration with traditional data warehousing, newer big data initiatives, and replication, to support database upgrades or migrations.

VSAM replication logging

Replication logging is a form of logging in CICS where updates to VSAM files, including RLS files, in one location are replicated on another site. In this way, multi-site operations can reduce their recovery time objective (RTO) to near zero.

The replication is done by an external product that uses the replication log records that CICS issues.

Role of IBM CICS in the journey to exceptional resilience

Table 2-4 shows how the role of IBM CICS deepens as your enterprise makes the journey toward higher availability. As stated previously, four deployment models encapsulate the key stages of this journey:

- Deployment model 1: Starting with a resilient and reliable base
- Deployment model 2: Reducing the duration of outages with failover capability
- Deployment model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS
- Deployment model 4: Adding GDPS Continuous Availability for exceptional resilience

The indentions of the following table emphasize how features that are located below an upper level inherit the features of that upper level.

Table 2-4 Role of IBM CICS in the journey to exceptional resilience

Deployment model 1: Resilient / Reliable **Transaction recovery** - State of all transactions stored in recovery log. **Resource cataloging** -Resource definitions and modifications recovered on warm restarts on any sysplex image. **Storage protection** - Isolation of transaction and system storage. **Application phasein** - Seamless application updates into running systems without outage.
Deployment model 2: Failover Capable The preceding list plus, **Region cloning** - Cloning TORs and AORs provides isolation and redundancy. **Data sharing** - FORs allow shared access to data from AORs. **Transaction routing** - static routing or simple round robin. **System policies** -Customizable automation rules.

Deployment model 3: Fault Tolerant
The preceding list plus,
Transaction classification -Granular throughput/capacity limits and prioritization of work under load.
CICSPlex SM - Workload management, with affinity management and WLM service classes for dynamic routing. WLM health reporting integrates CICS region health with other components.
Transport failover - Cross LPAR failover with Sysplex Distributor, MQ queue sharing, VTAM Generic Resource.
Data sharing - VSAM RLS, Db2 data sharing.

Deployment model 4: Continuous Availability
The preceding list plus,
Multi-site Workload Lifeline - Intelligent load-balancing advice for TCP/IP workloads across two sites.
VSAM Replication logging - updates to files are logged and replicated to remote site via IIDR for VSAM.

2.5.2 Information Management System

IBM Information Management System (IMS) is the trusted name in transactional databases and processing on IBM Z. It delivers accurate, consistent, timely, and critical information to application programs, which deliver the information to many end users simultaneously. IMS is made up of a hierarchical database and an online transaction management system.

IMS provides an environment for applications that require very high levels of performance, throughput, availability, and resiliency. IMS consists of three components, the Database Manager (IMS DB), the Transaction Manager (IMS TM), and a set of system services that provide common services to the other two components. IMS TM/DB creates a complete online transaction processing environment providing continuous availability and data integrity. Users can alternatively choose to use only IMS TM (DCCTL environment) or only IMS DB (DBCTL environment).

IMS TM is a message-based transaction processor. IMS TM provides services to:

- Process input messages received from a variety of sources (such as the terminal network, other IMS images, and from any platform).
- Process output messages created by application programs.
- Provide an underlying queuing mechanism for handling these messages.
- Provide high-volume, high-performance, high-capacity, low-cost transaction processing for both IMS DB hierarchical databases and Db2 relational databases.

IMS TM supports many terminal or client sessions at extremely high transaction volumes. IMS applications can be written in several programming languages such as assembler language, C, COBOL, Java, Pascal, PL/I, and REXX. The IMS resources are accessed by the application by calling several standard IMS functions. Applications access these functions through a standard application programming interface (API) for both the Transaction Manager and Database Manager components. This interface is DL/I (Data Language/I).

IMS DB is a DBMS that helps you organize business data with both program and device independence, including:

- Maintain data integrity. The data in each database is guaranteed to be consistent and guaranteed to remain in the database even when IMS DB is not running.

- IMS databases are accessed internally by using several IMS database organization access methods. The actual database data is stored on disk storage using normal z/OS access methods.
- Define the database structure and the relationships among the database elements.
- Provide a central point of control and access for the IMS data that is processed by IMS applications.
- Perform queries against the data in the database.
- Perform database transactions (inserts, updates, and deletes) as a single unit of work so that the entire transaction either occurs or does not occur.
- Perform multiple database transactions concurrently with the results of each transaction kept isolated from the others.
- Maintain the databases. IMS DB provides facilities for tuning the databases by reorganizing and restructuring them.

Additionally, IMS DB lets you adapt IMS databases to the requirements of varied applications. Application programs can access common and, therefore, consistent data, thereby reducing the need to maintain the same data in multiple ways in separate files for different applications.

IMS databases can be managed and accessed by using JDBC or SQL, similar to how you would use JDBC or SQL against a relational table.

IMS DB provides access to databases from applications that run under:

- IMS Transaction Manager
- CICS Transaction Server for IBM z/OS
- z/OS batch jobs
- WebSphere Application Server for z/OS
- Db2 for z/OS stored procedures
- Any distributed application from any platform using JDBC

IMS databases are hierarchical in structure. An IMS database is made up of segments arranged in hierarchical order. An example of hierarchical arrangement of data is an organization chart. The chart might have a President at the highest level, Senior Vice Presidents below that, lower-level Vice Presidents and Directors below that followed by levels of management, until you get to the individual contributors.

In an IMS database, the topmost segment is called the root segment. Information relevant to the root segment is stored in child segments. For example, you want to store information about an individual who has many accounts. Typically, the top segment stores the individual's name, address, contact number, and so on.

The second level of segments might be the accounts they have. Each account will be one segment. Under accounts, you might have transactions for each account segment.

In this structure, the account segments would be children segments of the parent, the client segment. However, the account segments are parents of the transaction segments.

SQL support allows you to issue standard SQL query to access IMS data instead of using DL/I calls. To use SQL calls in an IMS application, you need to understand the difference between the hierarchical model for IMS databases and the standard relational database model, because SQL calls are commonly used for relational databases. You also need to understand how IMS database elements are mapped to relational database elements. To use SQL or JDBC with IMS requires the application metadata to be in the IMS Catalog.

A database segment definition defines the fields for a set of segment instances like the way that a relational table defines columns for a set of rows in a table. In this regard, segments relate to tables, and fields in a segment relate to columns in a table. An occurrence of a segment in a database corresponds to a row in a table.

Database Definition (DBD)

DBD defines the structure of database. Each database has one DBD created through an IMS process called DBDGEN. To create a DBD, the database administrator codes a set of assembler language macros that are expanded to build control blocks to define the database. The generated code is stored in the DBD library. The DBD defines the physical structure of the database. In relational databases, you create tables by using SQL statements called DDL (Data Definition Language). IMS can also use DDL to define the database definitions in the IMS Catalog.

Program Specification Block (PSB)

To restrict the data that an application can access, create logical views in the PSB that is created through the IMS PSBGEN process. Similar to the DBDGEN, a PSBGEN is coded by using a set of assembler language macros that create control blocks. The generated control block is placed in a PSB library. IMS can also use DDL to define the program views in the IMS Catalog.

Application Control Blocks (ACBs) Library (ACBLIB)

ACBLIB data sets contain the ACBs, which describe IMS applications, and data management blocks (DMBs), which describe databases and the applications that can access them. An ACBGEN creates the ACBLIB by using the definitions from DBDLIB and PSBLIB. When using IMS managed ACBs, IMS gets this information from the IMS Catalog instead of ACBLIB.

IMS Catalog

The IMS catalog contains trusted metadata and definitions of the IMS databases and application program views that are defined to IMS.

If the IMS management of ACBs is enabled, the IMS catalog also determines the active databases and program views (PSBs) in the IMS system, because ACB libraries are not used. When IMS uses ACB libraries, the ACB library determines which databases and program views are active, and you must ensure that the IMS catalog is always in sync with the ACB library.

The IMS catalog is itself a HALDB PHIDAM database. Each database and application program view that is defined to IMS is stored in a separate record in the IMS catalog. In each record, the root header segment identifies the type of resource that it contains: either a database definition (DBD) or a program view (PSB).

When IMS manages the ACBs, you define databases and program views either by using SQL data definition language (DDL) statements or by using the input macros of the DBD Generation utility and PSB Generation utility.

When you use DDL statements, IMS adds the database and program view definitions to the IMS catalog and builds the required runtime control blocks. You can then use an IMPORT command to activate the new definitions.

You can use the DBD and PSB Generation utilities to define databases and program views in an IMS system that manages ACBs. After you run the utilities, you must also run the ACB

Generation and Populate utility (DFS3UACB) or equivalent utilities to build the ACBs, update the IMS catalog, and load the ACBs into the IMS system.

In an IMS system that manages ACBs, the IMS catalog completely replaces DBD, PSB, and ACB libraries. The IMS catalog becomes the only component that determines which database and program view definitions are used by the online IMS system and by batch application programs.

Data Sharing

An IMS system includes databases whose data can potentially be made available to, or shared with, all declared application programs. Access to a database is a characteristic defined in an application program's PSB. With data sharing support, application programs in separate IMS systems can concurrently access databases. When more than one IMS system has concurrent access to an IMS database, those systems are sharing data, and you have a data sharing environment.

Data sharing among IMS systems is supported in both sysplex and nonsysplex environments. Nonsysplex data sharing, also referred to as local data sharing, is data sharing between IMS systems on a single z/OS operating system. A coupling facility can be used, but it is not required. Sysplex data sharing is data sharing between IMS systems on different z/OS operating systems. IRLM uses a coupling facility to control access to databases.

High Availability Large Database (HALDB)

High Availability Large Database (HALDB) was introduced to extend the functions provided by IMS full-function databases. HALDB is designed to increase the capacity and availability of full-function databases. HALDB has three database types:

- PHDAM - Partitioned HDAM
- PHIDAM - Partitioned HIDAM
- PSINDEX - Partitioned secondary index

HALDB support includes the following features:

- OSAM and VSAM (ESDS and KSDS) access methods
- Logical relationships and secondary indexes
- Data sharing
- Online change
- OSAM sequential buffering
- IMS Monitor
- IMS Performance Analyzer product

The capacity of a non-HALDB full-function database is limited by the number of data sets allowed in a database and their maximum sizes. The capacity of a data set is either 4 GB if it is VSAM or 8 GB for OSAM. For OSAM, up to 10 data sets are allowed. All segments of the same type must be in the same data set. So, when one data set is full, the database is full.

HALDB removes this limit by removing the restriction that all occurrences of the same segment type must be in the same data set. HALDB groups database records into sets of partitions. The hierarchic structure is maintained within a partition and a whole database record always resides in a single partition. Partitions might be very large. Each partition might have up to 10 data sets. A database might have up to 1001 partitions, so segments in a HALDB database might be stored in up to 10,010 data sets. Each HALDB data set might be up to 4 GB, a HALDB database might contain over 40 terabytes of data. If OSAM is used without Online Reorganization, the database can be up to 8 GB so the limit for these would be 80 TB.

HALDB also has availability and manageability benefits. Partitions might be managed, allocated, authorized, and allocated independently. Some partitions might be available to an online system while others are being processed by utilities. Partitions support online reorganization, which does not require a database outage during reorganization. All of which increases the flexibility available in handling databases.

For further details on IMS HALDB, see *The Complete IMS HALDB Guide*: https://www.redbooks.ibm.com/redbooks/pdfs/sg246945.pdf

Modifying an IMS Database

HALDB ALTER allows you to change the structure of an IMS HALDB without a DB outage. This applies if you want to add a new field (or fields) to space at the end of an existing segment or increase the length of an existing segment. The ALTER is performed as part of the online reorganization process.

Dynamic change to an OSAM or VSAM buffer pool allows you to modify the buffer pools without an IMS outage. Commands are used to Add, Change, or Delete FF Database Buffer Pools.

Fast Path DEDB ALTER allows dynamic change to DEDB Area without an unload/reload of the area while areas remain online. The types of changes you can ALTER:

- CI size of an area
- Root addressable and independent overflow parts of a DEDB
- Randomizer name

Choosing to use the Fast Path 64-bit buffer manager provides benefits. It can eliminate U1011 abends due to ECSA fragmentation. It is self-tuning in that IMS automatically allocates and manages the subpools. The user doesn't have to specify the number of buffers. Additionally, it supports multiple buffer sizes, which better utilizes buffers when you use areas that have differing CI sizes.

Recovering in an IMSplex using FDBR

The recovery of databases and releasing of locks in an IMS system can be managed by Fast Database Recovery (FDBR). An FDBR region monitors an IMS subsystem and can automatically recover database resources (shared databases and areas) and release locks if that IMS subsystem fails. That allows other IMS systems in a sharing environment to quickly regain access to the database without waiting for the failed IMS to restart and recover.

An IMS Database Control (DBCTL) warm standby region and an IMS Fast Database Recovery region (FDBR) can also be used for recovery purposes. While one DBCTL subsystem is active, you can start another DBCTL subsystem as a standby alternate. This alternate is a fully initialized IMS DBCTL subsystem that is waiting for a restart command.

IMS DBRC

DBRC is an IMS component that facilitates easier recovery of IMS databases and controls access to database resources (for example, authorizes IMS access to the databases). Also, it maintains information on the log data sets. DBRC maintains information that is required for database recoveries, generates recovery control statements, verifies recovery input, maintains a separate change log for database data sets, and supports the sharing of IMS databases and areas by multiple IMS subsystems. DBRC is an integral part of IMS. IMS relies on DBRC to record and manage information about many items. DBRC keeps this information in a set of VSAM data sets that are collectively called the *recovery control* (RECON) *data set* and advises IMS (based on the information in the RECON data set) about how to proceed. DBRC automatically records information in dual recovery control (RECON)

data sets. Both data sets contain identical information, and so are usually referred to as one: the RECON. The information from the RECON is needed during warm and emergency restarts. DBRC helps you ensure IMS system and database integrity by recording and managing information associated with the logging process. And DBRC assists IMS in the restart process by notifying IMS which logs to use for restart. DBRC also plays a key role in managing the log data needed to restart and recover IMS online subsystems. IMS can remain active while a RECON upgrade is performed for migration to a new version. Most IMS configurations require DBRC, including:

- Online configurations: DB/DC, DCCTL, or DBCTL
- Data-sharing environments, including IMSplexes

IMS Log

Logs are lists of activities, lists of work that have been done, and lists of changes that have been made. By knowing the state of the system when things were good, and knowing what the system has done since then, you can recover it in the event of a failure.

Logs make recovery and restart possible. As IMS operates, it writes event information in log records. The log records contain information including:

- When IMS starts up and shuts down
- When programs start and terminate
- Changes made to the database
- Transaction requests received and responses sent
- Application program checkpoints
- System checkpoints

IMS externalizes log records by writing them to an Online Log Data Set (OLDS). To enhance performance and to optimize space on an OLDS, incomplete or partially filled buffers are written to a Write Ahead Data Set (WADS) when necessary for recoverability. A WADS is a high-speed DASD data set with a high write rate. Only complete log buffers are written to OLDS. When the log data is on an OLDS, the equivalent WADS records are ignored. The OLDS and WADS are failure tolerant when properly configured. *IMS provides two modes for logging. Single logging* uses a single data set. *Dual logging* uses two data sets, where both data sets are identical. Use dual logging whenever possible because the OLDS is of primary importance to system integrity.

IMS uses a set of OLDSs in a cyclical way, which allows IMS to continue logging when an individual OLDS is filled. Also, if an I/O error occurs while writing to an OLDS, IMS can continue logging by isolating the defective OLDS and switching to another one.

After an OLDS has been used, it is available for archiving to a System Log Data Set (SLDS) on DASD or tape by the IMS Log Archive utility. The utility can be executed automatically through an IMS startup parameter (ARC=).

When IMS is close to filling the last available OLDS, it warns you so you can ensure that archiving completes for used OLDSs or add new OLDSs to the system. You can also manually archive the OLDSs to SLDSs by using the IMS Log Archive utility. An SLDS can reside on DASD or tape. After an OLDS is archived, it can be reused for new log data. You use SLDSs as input to the database recovery process.

When you archive an OLDS, you can request that IMS write a subset of the log records from the OLDS to another log data set called the recovery log data set (RLDS). An RLDS contains only those log records that are required for database recovery.

While IMS is running and logging its activities, it takes periodic system checkpoints, and writes the checkpoint notification to another data set called the restart data set (RDS). IMS uses the RDS when it restarts to determine the correct checkpoint from which to restart.

IMSplex

An IMSplex is made up of IMS and z/OS components that work together. Connected systems such as Multiple Systems Coupling, and data sharing systems are used in an IMSplex. In an IMSplex, a set of IMS systems work together to share resources or message queues (or both) and workload. An IMSplex can also be defined as a single IMS system by using the Common Service Layer (CSL) with or without the Resource Manager (RM) to provide a single point of control.

By definition, a Parallel Sysplex is multiple z/OS images that work together, connected by a coupling facility. In contrast, one or more IMSplex systems can be defined on one or more z/OS images. However, you do not need an IMS instance on every z/OS image in the sysplex.

The concept of data sharing in an IMSplex — specifically in an IMSplex that shares resources and workload — allows IMS batch jobs, online applications, or both, to run anywhere in the IMSplex. These batch jobs and applications can also access all the data in the shared IMS databases.

Consider a case where you have multiple IMS systems that share resources or message queues. You can simplify administration and operation by using the IMS Common Service Layer (CSL), which reduces complexity by providing a single-image perspective. With CSL, you can manage multiple IMS systems as if they were only one system.

When you manage an IMSplex with the CSL, the resulting environment is comprised of multiple IMSplex components. An IMSplex component either manages resources, manages operations, or facilitates communication among IMSplex components. The following are IMSplex components:

- Operations Manager (OM)
- Resource Manager (RM)
- Open Database Manager (ODBM)
- Structured Call Interface (SCI)
- IMS Connect
- Database Recovery Control (DBRC)
- Common Queue Server (CQS)
- Repository Server (RS)
- TSO Single Point of Control (SPOC)
- Automated operator applications such as one that uses the REXX SPOC API for automation
- IMS Online control region

Any program that registers with the SCI is considered an IMSplex component. Users can also write programs that register with SCI. These programs are also considered IMSplex components. Note that batch regions and dependent regions (MPR, BMP, IFP, JMP, and JBP) are not IMSplex components. Although DBRC code that runs within a batch region registers with SCI, the batch region itself is not a component. When an IMSplex component is initialized and joins the IMSplex, it becomes an IMSplex member.

For further details on IMSplex concept, planning, implementation, backup, and recovery, see the following documents:

- IMS in the Parallel Sysplex - Volume I: Reviewing the IMSplex Technology
  http://www.redbooks.ibm.com/redbooks/pdfs/sg246908.pdf

- IMS in the Parallel Sysplex - Volume II: Planning the IMSplex
  http://www.redbooks.ibm.com/redbooks/pdfs/sg246928.pdf
- IMS in the Parallel Sysplex - Volume III: IMSplex Implementation and Operations
  http://www.redbooks.ibm.com/redbooks/pdfs/sg246929.pdf)

IMS shared queues

Operating in a shared-queues environment allows multiple IMS systems in a sysplex environment to share IMS message queues and Expedited Message Handler (EMH) message queues. The IMS systems work together as an IMSplex that provides a single-image view of multiple IMS systems. The shared-queues environment distributes processing loads between the IMS systems in the IMSplex. Transactions that are entered on one IMS can be made available on the shared queues to any other IMS that can process them. Results of these transactions are then returned to the initiating end user. If one IMS fails, the messages on the shared queues can be processed by the remaining IMS systems. When systems are cloned so that any application can run on any IMS, an IMS outage is not an application outage. End users view the processing as if they were operating a single-system:

- A shared queue is a collection of messages that are associated by the same queue name.
- A shared queue is managed by a Common Queue Server (CQS) and can be shared by CQS clients in an IMSplex.
- CQS receives, maintains, and distributes data objects from a shared queue that resides in a coupling facility (CF) list structure on behalf of multiple clients. CQS is a generalized server that manages objects on a CF structure, such as a queue structure or a resource structure.
- A CQS client for the message-queue structure is an IMS DB/DC or DCCTL system that accesses shared queues through CQS.

The following options help you to tune your IMS shared-queues system to provide higher availability and resiliency:

- Adjust number of queue buffers to prevent QBUF waits.
- Size queue buffers according to majority of queue activity.
- Watch over the message queues to prevent runaway or looping applications that would otherwise fill up the message queue. Also look for high input message arrival rates.
- Use the SEGNO parameter on the transaction definition to limit number of segments that can be inserted to the message queue.
- Use the DFSQSPC0 (non-SQ) / DFSQSSP0 (SQ) user exits to control queuing.
- Use tools such as Queue Control Facility to offload messages.

IMS Resource Structure

The coupling facility IMSRSC resource structure when used with RM in a sharing environment allows for consistency of resources and allows certain types of status to be retrieved across a cold start. A CQS client for the resource structure is an IMS DB/DC, DCCTL, or DBCTL system that accesses the structure through CQS. The benefits of using RM and a resource structure include:

- Enforcement of resource type consistency:
  Prevents the same name being used for more than one resource type.
- Enforcement of resource name uniqueness:
  Ensures that certain resources can be active only one at a time within the IMSplex.

- Global callable services:
  Allows exit routines to obtain LTERM, node, and user resource information across the IMSplex.
- Terminal and user status recovery:
  Allows terminals and users to resume work without having affinity to any IMS.
- VTAM can manage Generic Resource affinities:
  Specifying GLOBAL as the status recovery mode allows VTAM to manage Generic Resource affinities instead of IMS. If an IMS system fails, VTAM can reassign terminals to another IMS system even if the terminals have affinity to the failed IMS system.
- Retrieval of enablement values for global IMS functions across IMS cold starts:
  If you enable an IMS function globally by using the UPDATE IMSFUNC command and you use RM and a resource structure, the function enablement value is stored in a resource structure so that the value is retrieved across an IMS cold start.

Dynamic Resource Definition

With dynamic resource definition (DRD), you can use type-2 commands to define MODBLKS resources such as application programs, databases, routing codes and transactions, and resource descriptors. You also can use type-2 commands to define MSC resources such as definitions for MSC physical links, logical links, logical link paths, and remote logical terminals. This allows the definitions to be defined while IMS is active. If you use DRD to define your resources, you can eliminate some system generation tasks and libraries. Using DRD can eliminate certain types of planned outages for definition changes. Dynamic changes should be hardened to the IMSRSC repository so they can be recovered on an IMS cold start. On a warm start, the definitions are recovered from the IMS log.

IMS Repository

A repository is a generalized data storage facility that can be used to store any type of information.

The repository is managed by the Repository Server (RS) address space. Resource and descriptor definitions can be added, queried, modified, or deleted from the repository with commands.

The IMS resource definition (IMSRSC) repository is a set of VSAM key sequenced data sets (KSDSs) used for storing information. The IMSRSC repository is a repository to store resource (and descriptor) definitions for IMS databases, transactions, programs, and routing codes. The repository provides the ability for an IMSplex to use a single shared repository to store resource and descriptor definitions for all the members of an IMSplex.

The repository data sets are defined as two data sets with one spare. The two data sets are intended to be a duplexed pair. If one of the repository data sets in the pair loses integrity, it is discarded. If one of the pair is discarded due to a write error, the repository is stopped to enable recovery. In this event, the RS drives recovery automatically if a SPARE repository data set pair is available. If no SPARE repository data set is available, the repository is stopped, and administrator intervention is required to restart the repository.

IMS user exits

Several user exits can be refreshed dynamically by using the REFRESH USEREXIT command. This can eliminate planned outages to update exits such as when installing a new tool.

IMS Connect

IMS Connect is the TCP/IP gateway into IMS for operations, transactions, and database. It is a system service function provided by IMS. IMS Connect uses IP Spraying, workload balancing, and sysplex distribution mechanisms, and it connects IMS Connect clients to IMS Systems (both IMS DB and IMS TM). IMS Connect uses the Open Database Manager (ODBM) feature of IMS Common Service Layer to access IMS DB. For transaction processing, IMS Connect uses Open Transaction Manager Access (OTMA) to access IMS TM. IMS Connect can also be used with MSC and ISC processing to communicate with other IMS and CICS systems.

Multiple IMS Connect instances can be set up to connect to OTMA by using cross-system coupling facility (XCF) service that is provided by z/OS for workload sharing and increasing scalability and availability. Similarly, multiple instances of IMS Connect can connect to ODBM by using IMS Structured Call Interface (SCI) services.

When IMS assets serve as REST API endpoints through the IMS service provider, the first place to configure for IMS Connect high availability is in the TCP/IP network that sends and receives messages. IMS Connect availability is typically achieved with static or dynamic virtual IP addresses (VIPA) or Sysplex Distributor for TCP/IP load balancing and failover.

Connections to IMS from the IMS service provider in z/OS Connect Enterprise Edition can be configured to go through a port on the Sysplex Distributor that is shared by multiple IMS Connect instances for high availability.

Certain IMS Connect configuration definitions can be specified via a command, which eliminates an IMS Connect planned outage.

With z/OS WLM integration, region health information is integrated with Sysplex Distributor for achieving better workload balancing. WIth the IMS Connect super member group, you can retrieve (Resume TPIPE requests) of any queued asynchronous messages regardless of the connection path. Shared queues can be used to allow any IMS to deliver the message even if the IMS that originally created the message is unavailable.

Using a super member group, you can configure OTMA to send output messages to a remote IMS installation through up to eight local instances of IMS Connect. The use of a super member group is transparent to both the IMS application that sends the messages and the remote IMS installation that receives the messages. After the super member group is defined and the IMS application begins sending output messages, OTMA distributes the output messages to each participating IMS Connect instance in turn by using a round-robin algorithm. OTMA dynamically updates the super member round robin list each time an IMS Connect instance joins or leaves the super member group.

Just like any other transaction processing system and database on mainframe, IMS provides a variety of backup, restore, and high-availability features for data sharing as well as non-data sharing environments. IBM also offers a wide range of products that can simplify your IMS system management, backup, recovery, and high-availability solution and make them more effective and agile. Depending on the nature of failure and the component that has failed, appropriate recovery actions and restart procedures should be performed.

Open Transaction Manager Access

IMS Open Transaction Manager Access (OTMA) is a transaction-based, connectionless client/server protocol. The domain of the protocol is restricted to the domain of the z/OS cross-system coupling facility (XCF).

OTMA addresses the problem of connecting a client to a server so that the client can support a large network, or many sessions, while maintaining high performance.

If IMS cannot process the messages received from an OTMA client quickly enough or at all, the buildup of messages in the IMS system can result in a *message flood* condition. In this condition, below-the-line storage might be exhausted by the requirements of both the messages and the transaction instance blocks (TIBs) that IMS uses to process the messages. As a result, a z/OS S40D system abend could occur.

To avoid the problems associated with a message flood condition, OTMA can monitor the number of input messages that are waiting to be processed in the IMS system. When monitoring is active, OTMA monitors both the number of input messages from each OTMA client and the total number of input messages from all OTMA clients combined. When the number of input messages from an individual OTMA client reaches its defined maximum, OTMA suppresses new input messages from that client. When the total number of input messages reaches the global maximum, OTMA issues a warning message, but still accepts new input messages.

OTMA detects when the number of messages in the IMS system reaches 80% of the maximum number defined for either an OTMA client or for all OTMA clients. Then, OTMA issues a server state protocol command to the appropriate OTMA clients and IMS issues message DFS1988W to the master console. IMS also issues DFS1988W at every 5% increase thereafter until the maximum number is reached.

When the number of messages in the IMS system for an individual client exceeds the maximum, OTMA issues another server state protocol command and IMS issues message DFS1989E. OTMA then rejects all subsequent input messages from the OTMA client until the message flood condition is resolved.

OTMA detects when the number of messages in the IMS system falls to 50 percent of the maximum. Then, OTMA issues a new server state protocol command to the OTMA clients notifying them that the flood condition has been relieved.

OTMA descriptors can be defined dynamically, thereby eliminating a planned IMS outage in order to change a descriptor.

IMS software-based data mirroring - InfoSphere IMS Replication for z/OS

Software replication can provide availability to the data on an alternate site when a disruption occurs at the primary site. IMS supports replication of IMS databases to an alternate site as part of an active/standby or active/query environment. Data is captured by using the IMS 99 log records. IIDR for IMS sends the data to an IIDR apply server, which uses another IMS to apply the changes.

You can use IIDR to produce copies of your IMS databases and maintain current data in near-real time, typically on geographically dispersed IMS subsystems. It addresses your organizational requirements for reliable and available data: High availability and disaster recovery (HADR), Workload distribution, Business intelligence, Site/database maintenance, and Data backup. IMS Replication can capture changes from a single DB/DC or DBCTL subsystem. It can also capture changes from hundreds or thousands of shared databases in an IMS data-sharing environment. You can capture updates from multiple IMS DL/I batch jobs, DB/DC subsystems, or DBCTL subsystems in a sysplex. IMS Replication also supports high-availability environments that implement FDBR (Fast Database Recovery) regions.

IMS Replication has the following key components and capabilities: Subscriptions, Transactional consistency, Replicating historical changes, and Monitoring and reporting.

For further details about this product offering, see:
https://www.ibm.com/support/knowledgecenter/en/SS4T2J_11.3.0/home/product_landing.html

Role of IBM IMS in the journey to exceptional resilience

Table 2-5 on page 73 shows how the role of IMS deepens as your enterprise makes the journey toward higher availability. As stated previously, four deployment models encapsulate the key stages of this journey:

- Deployment model 1: Starting with a resilient and reliable base
- Deployment model 2: Reducing the duration of outages with failover capability
- Deployment model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS
- Deployment model 4: Adding GDPS Continuous Availability for exceptional resilience

The indentions of the following table emphasize how features that are located below an upper level inherit the features of that upper level.

Table 2-5 Role of IBM IMS in the journey to exceptional resilience

Deployment model 1: Resilient / Reliable Backup to tapes and disks with customized procedures, stand-alone IMS systems, manual recovery procedures
Deployment model 2: Failover Capable The preceding list plus, **Data replication**. **IMSplex**. **IMS Connect**. **Data sharing**. **Dynamic Resource Definition.** **IMS Repository.** **Shared Message Queues.** **Tool-based recovery for CPC** - Tool-based recovery and automated restart and failover capabilities within a CPC.
Deployment model 3: Fault Tolerant The preceding list plus, **Cross-system configuration** - IMSplex and IMS Connect configuration that spans multiple systems.
Deployment model 4: Continuous Availability The preceding list plus, **Software-based replication**. **GDPS CA** for near zero RTO and RPO.

2.5.3 IBM WebSphere Application Server for z/OS

IBM WebSphere Application Server for z/OS is a comprehensive, sophisticated, Java Platform, Enterprise Edition, and Web Services technology-based application system.

WebSphere Application Server for z/OS is the Java Platform, Enterprise Edition implementation that conforms to the current Software Development Kit (SDK) specification supporting applications at an API level. As mentioned, it is a Java Application deployment and runtime environment built on open standards-based technology that support all major functions such as servlets, JavaBeans (EJBs) including the latest technology integration of services and interfaces.

The application server runtime is highly integrated with all inherent features and services, allowing z/OS qualities of service to be extended to modern-day Java and Web applications. The application server can interact with all major subsystems on the operating system including Db2, CICS, and IMS. It has extensive attributes for security, performance,

scalability, and recovery. The application server also uses sophisticated administration and tooling functions, thus providing seamless integration into any data center or server environment.

WebSphere Application Server is an e-business application deployment environment. It is built on open standards-based technology such as CORBA, HTML, HTTP, IIOP, and J2EE-compliant Java technology standards for servlets, JavaServer Pages (JSP) technology, and Enterprise JavaBeans (EJB). And it supports all Java APIs that are needed for Java Platform, Enterprise Edition compliance.

Because different application scenarios require different levels of application server capabilities, WebSphere Application Server is available in multiple packaging options. WebSphere Application Server Network Deployment provides a flexible, secure server runtime environment for large-scale and mission-critical application deployments. It extends the base package of WebSphere Application Server to include key features that tend to be more important in larger enterprises. In these environments, applications tend to service a larger client base, and more elaborate performance and high-availability requirements are in place. These features include various functions for high availability and high scalability to ensure responsiveness to your application requests. This package is also referred to as the Network Deployment package.

IBM WebSphere Application Server for z/OS provides the same functions as WebSphere Application Server Network Deployment, but adds key functions that use IBM z/OS qualities of service. You can use the z/OS qualities of service to optimize performance and provide continuous availability for mission-critical applications.

Stand-alone application servers: All WebSphere Application Server packages support a single stand-alone server environment. With a stand-alone configuration, each application server acts as a unique entity, functioning independently from other application servers. An application server runs one or more applications, and provides the services that are required to run these applications.

Application servers in Network Deployment: With Network Deployment, you can build a distributed server configuration to enable central administration, workload management, and failover. In this environment, you integrate one or more application servers into a cell that is managed by a central administration instance, a deployment manager. With a distributed server configuration, you can create multiple application servers to run unique sets of applications, and manage those applications from a central location. More importantly, you can cluster application servers to allow for workload management and failover capabilities. Applications that are installed in the cluster are replicated across the application servers. The cluster can be configured so that when one server fails, another server in the cluster continues processing. On z/OS, workload is distributed among containers in a cluster by using Workload Manager (WLM).

WebSphere Application Server for z/OS provides two different implementations: traditional and Liberty. Traditional WAS is sometimes referred to as WebSphere Application Server “classic” or “full profile”, whereas Liberty is sometimes called “WAS Liberty” or “Liberty profile”.

Liberty is a highly composable, fast to start, dynamic application server runtime environment. Liberty’s runtime is very compact and lightweight, making it particularly well-suited for virtualized infrastructure and cloud computing, for lightweight containers, and also for devices in the Internet of Things. WebSphere Liberty is Java EE8 and Jakarta EE8 compliant and it is based on the open source project Open Liberty, which can be found in GitHub. For a detailed discussion on Jakarta EE8 and Java EE8 in Liberty and its benefits see:

https://www.ibm.com/support/knowledgecenter/SS7K4U_liberty/com.ibm.websphere.wlp.zseries.doc/ae/cwlp_javaee8.html

On z/OS systems, Liberty provides an operations environment. You can work natively with this environment by using the z/OS console. For application development, consider using the Eclipse-based developer tools on a separate distributed system, or on Mac OS. Liberty supports a subset of the following parts of the full WebSphere Application Server programming model:

- Web applications
- OSGi applications
- Enterprise JavaBeans (EJB) applications

Associated services such as transactions and security are available to these application types. With the Liberty zero-migration architecture, you can move to the latest version of Liberty with minimal impact to your current applications and configurations.

Features are the units of functionality by which you control the pieces of the runtime environment that are loaded into a particular server. For a list of the main Liberty features, see Liberty features. You can also create your own features, as described in Extending Liberty. You can organize Liberty servers into collectives to support clustering, administration, and other operations that act on multiple Liberty servers at a time. In that way, you can efficiently and accurately deliver application services to your organization. See Collective architecture for more information.

For a detailed discussion on WebSphere Liberty and its features see:
https://www.ibm.com/support/knowledgecenter/SS7K4U_liberty/com.ibm.websphere.wlp.zseries.doc/ae/cwlp_about.html

Clustering and high availability

A cluster is a collection of servers that are managed together. With clusters, enterprise applications can scale beyond the amount of throughput that can be achieved with a single application server. Also, enterprise applications are made highly available because requests are automatically routed to the running servers in the event of a failure. An application server cluster is a logical collection of application server processes that provides workload balancing and high availability. Application servers that are a part of a cluster are called *cluster members*. It is a grouping of application servers that run an identical set of applications that are managed so that they behave as a single application server (parallel processing).

Vertical Cluster: When cluster members are on the same system, the topology is known as vertical scaling or vertical clustering. Vertical clusters offer failover support within one operating system image, provide processor level failover, and increase resource utilization.

Horizontal Cluster: Horizontal scaling or horizontal clustering refers to cluster members that are spread across different server systems and operating system types. In this topology, each system has a node in the cell that is holding a cluster member. Horizontal clusters increase availability by removing the bottleneck of using only one physical system and increasing the scalability of the environment. Horizontal clusters also support system failover.

Mixed Cluster: A mixed cluster combines both vertical and horizontal clustering. It typically has multiple members on participating z/OS images and members that are spread over multiple physical systems. This configuration provides a mix of failover and performance.

Dynamic Cluster: Dynamic clusters are application deployment targets that operate at the application-layer virtualization. Dynamic clusters provide capabilities to better manage dynamic workload by using the on-demand router server.

WebSphere Application Server provides various backup methods that vary in cost and effectiveness. The key to the appropriate strategy is to find the correct balance between recovery time and cost. Typically sites use a combination of backup, load balancing, and high availability to maintain the service available. Based on the criticality of the business applications, one can decide the type of backup and restore strategy.

Consider this example: A site has multiple HTTP servers that are deployed along with other application servers. Your backup strategy could be to stop one of the HTTP servers. The configuration repository is then saved in the remote storage, and the server is restarted. Then, the same operation is run for the second server and so on. WebSphere Application Server provides commands such as 'backupConfig' and 'restoreConfig' commands to back up the configuration online and restore it. Depending on your backup and restore strategy, you can schedule periodic backup of the product components. Applications can be backed up by copying the application directory.

Note that the topics discussed so far in this section are applicable to both traditional and Liberty implementation of WebSphere Application Server for z/OS.

High Availability Manager

WebSphere Application Server includes a high availability manager component. The services that the high availability manager provides are only available to product components.

A high availability manager provides several features that allow other product components to make themselves highly available. A high availability manager provides:

- A framework that allows singleton services to make themselves highly available. Examples of singleton services that use this framework include the transaction managers for cluster members, and the default messaging provider, also known as the service integration bus.
- A mechanism that allows servers to easily exchange state data. This mechanism is commonly referred to as the *bulletin board*.
- A specialized framework for high speed and reliable messaging between processes. This framework is used by the data replication service when the product is configured for memory-to-memory replication.

A high availability manager instance runs on every application server, proxy server, node agent, and deployment manager in a cell. A cell can be divided into multiple high-availability domains known as core groups. Each instance of the high availability manager establishes network connectivity with all other instances of the high availability manager in the same core group. Each instance uses a specialized, dedicated, and configurable transport channel. The transport channel provides mechanisms that allow each high availability manager instance to detect when other members of the core group start, stop, or fail.

Within a core group, high availability manager instances are elected to coordinate high-availability activities. An instance that is elected is known as a *core group coordinator*. The coordinator is highly available, such that if a process that is serving as a coordinator stops or fails, another instance is elected to assume the coordinator role, without loss of continuity.

The coordinator also associates a high availability policy with each high availability group. A high availability policy is a set of directives that aid the coordinator in managing highly available components.

The coordinator is notified as core group processes start, stop, or fail and knows which processes are available at any given time. The coordinator uses this information, in conjunction with the high availability group and policy information, to ensure that the

component keeps functioning. The coordinator uses the policy directives to determine on which process it starts and runs each component. If the chosen process fails, the coordinator restarts the component on another eligible process. This reduces the recovery time, automates failover, and eliminates the need to start a replacement process.

The high availability manager provides a specialized messaging mechanism that enables processes to exchange information about their current state. Each process sends or posts information related to its current state, and can register to be notified when the state of the other processes changes. The workload management (WLM) component of WebSphere uses this mechanism to build and maintain routing table information. Note that WLM component of WebSphere is different from WLM component of z/OS. Routing tables that are built and maintained through this mechanism are highly available.

The data replication service (DRS) that is provided with the product, is used to replicate HTTP session data, stateful EJB sessions, and dynamic cache information among cluster members. When DRS is configured for memory-to-memory replication, the transport channels defined for the high availability managers are used to pass this data among the cluster members.

Role of IBM WebSphere Application Server for z/OS in the journey to exceptional resilience

Table 2-6 shows how the role of IBM WebSphere Application Server deepens as your enterprise makes the journey toward higher availability. As stated previously, four deployment models encapsulate the key stages of this journey:

- Deployment model 1: Starting with a resilient and reliable base
- Deployment model 2: Reducing the duration of outages with failover capability
- Deployment model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS
- Deployment model 4: Adding GDPS Continuous Availability for exceptional resilience

The indentions of the following table emphasize how features that are located below an upper level inherit the features of that upper level.

Table 2-6 Role of IBM WebSphere Application Server in the journey to exceptional resilience

Deployment model 1: Resilient / Reliable **Backup** to tapes and disks with customized procedures. **Stand-alone application servers**. **Manual restart** of application severs.
Deployment model 2: Failover Capable The preceding list plus, **Data replication**. **Automated restart** of application servers in network deployment setup in single CPC.
Deployment model 3: Fault Tolerant The preceding list plus, **Router** - Implementation of on-demand router. **Clustering mix** - Both vertical clustering and horizontal clustering. **Workload balancing**. **Spanning** - Cell that spans more than one CPC.
Deployment model 4: Continuous Availability The preceding list plus, **Dynamic clustering** with on-demand router. **HA manager** - High availability manager implementation. **GDPS CA** for near zero RTO and RPO.

2.5.4 Batch processing

In every industry, batch processing is still a fundamental, mission-critical component that is commonly used by businesses to do key tasks, such as:

- Generating reports of all daily processed data
- Printing or sending account statements to customers
- Paying salaries for employees
- Running analytics on a large Data Warehouse
- Archiving historical data at the end of month
- Optimizing databases
- Creating backups of files and databases for disaster-recovery purposes
- Processing files with large amounts of data from business partners

Batch jobs do not always include an application program. For example, many times a utility is used to extract data, reorganize a database, create a backup copy of data, or replicate a file.

Batch processing has significant advantages when it comes to performing repetitive logic. Static data is read only once, then cached and used throughout the program. Also, for example, one SQL query to update a status field in 10,000 rows of a database table is far more efficient than 10,000 online transaction processing (OLTP) programs that each run a query to update the status field, one row at a time.

Batch processing is bulk processing of massive amounts of transactions that typically runs during non-office hours, and remains a viable and strategic option to this day. In addition, because businesses want results faster without waiting for overnight processing, batch processing also needs to run any time during the day, in parallel with the OLTP window.

This section describes running batch processing together with the database environments, such as Db2, IMS, and CICS and non-database environments, such z/OS data files.

Considerations for batch processing can be found in 4.4.5, “Batch processing” on page 137.

Batch processing with data files

For batch processing with data files, there are several recovery scenarios to prepare for, such as:

- Batch job failures
- Disk device failures
- Site failures (Disaster Recovery)[10]

Recovery after job failures

Because batch does not do any logging, if a job fails, you need to back out to the state at the beginning of the job. A process to do this by making copies of files before they are updated is described in “VSAM Record Level Sharing and Transactional VSAM” on page 145.

Recovery after disk failures with synchronous disk remote copy

After a disk failure, GDPS or Copy Services Manager can HyperSwap to the secondary disk non-disruptively. Without HyperSwap capability, recovery will be similar to what happens after job failures. Note that FlashCopies can be done on the primary and secondary site simultaneously by using the “Remote Pair FlashCopy (Preserved Mirror)” process. There is no need to initiate a FlasyCopy on the primary site and then replicate the file or volume to the secondary site.

[10] These backups are stored off site and, in the event of a major disaster, are recovered at another site.

Recovery after site failures with synchronous disk remote copy

After a site failure, restart on the secondary site will be similar to restarts after disk failures. The difference is that if HyperSwap is enabled, often the disk would have swapped, but it is not guaranteed. This depends on the timing of the failures between the disk, servers, and network connections.

Recovery after disk/site failures with recovery from tape

If backing up data to tape or virtual tape, things can get more complex. Many sites do not give the *storage administrator* the responsibility of backing up application data. Rather, it is the job of the *application owner* to ensure that their data is backed up and to determine how often — based on their understanding of the criticality of the application — if they are “Tier 1”, “Tier 2”, or “Tier 3.”

A concern is how the data is restored. SMF type 15, 30, 62, and 64 records (and others) can be used to verify all the files are being archived. Often the Tier 1 application data — which might be archived daily — is the first to be restored. After completion, these Tier 1 applications might start running in the D/R location. The Tier 2 application data then gets restored, and finally Tier 3 application data, which might be archived weekly. But what process is used to identify if a file is used by multiple applications in multiple tiers? For example, a SALES application file can be also used by the PAYROLL application to figure out commission. When the “Tier 3” data gets loaded from the weekly restore, it might overwrite a file that previously was loaded for a Tier 1 application. This can introduce data integrity problems. There needs to be either coordination between all the application owners for all the files being used, or you must centralize control over data archives under the storage administrator.

How do you know that you captured the needed files

For all the possible failures (Job, disk, DR), you should verify that all the files that you are updating are backed up before the batch jobs run.

One method to verify that data has been backed up is to use the 'TSO FCQUERY DEVN(....) SHOWRELS' command. It is used to determine the copy status of any device that is available to the host system. The information reflects the storage control type and whatever information is available about the device will be returned, such as whether the device is the source or target of a FlashCopy relationship, the number of total relationships, and so on. The SHOWRELS ALL keyword displays the relationship table with extent information for all of the relationships for the device, although not sent to the SYSLOG and not broken down by file names.

Batch processing with databases

Database managers do locking for serialization, and logging to support backouts. Batch jobs can optionally call database managers such as IMS or Db2 to access data with embedded DLI or SQL calls. VSAM, used by CICS, is not a database manager, so care must be taken when batch jobs access VSAM data that is also used by CICS. Batch jobs can call both IMS and Db2 in the same job step.

There are several ways in which batch jobs can call Db2. These can be either a normal job that calls Db2 as the program name. It can be a job that uses the TSO interface (PGM=IKJEFT01). Or more interactively, the call can be the “SPUFI” SQL issued from the DB2 Interactive (DB2I) set of ISPF panels.

IMS supports two main categories of batch:

- DLI Batch and DBB Batch (Database Batch)
- Batch Message Processing (BMP) regions. This includes:

- Transaction-oriented BMPs
- Batch-oriented BMPs
- Java Batch Processing (JBP) regions

DLI Batch and DBB Batch are both stand-alone batch jobs, similar to typical z/OS batch jobs. They just happen to call IMS and run it in batch mode so they can access IMS data. DLI and DBB batch differ in how they get information on what IMS resources are accessible. DLI batch reads from IMS PSB and DBD libraries, while DBB Batch reads from the ACB library. They cannot access the IMS message queue.

In the past, DLI/DBB jobs were used to process bulk IMS data efficiently. They were designed to have exclusive control over the database. As such, there was no need for locking and simpler logging. This allowed for better performance than Batch Message Processing (BMP) regions. However, this required the online IMS system to either terminate or issue commands to stop access to the databases until the DLI and DBB batch jobs finished. Because of this limitation, they can run together with other IMS regions only if a data sharing environment has been implemented or if the batch jobs access the databases by using RO (Read Only without integrity).

BMPs are similar to IMS Message Processing Regions (MPRs) that run transactions, but the BMPs are designed for batch jobs instead of transactions to handle larger bulk processing of data. Think of them as just long running transactions. They run under the same Control Region as the MPRs, which are responsible for logging, backouts, and managing database access.

BMPs that run under the same control regions as MPRs have these traits:

- Have access to the databases used by the online system.
- Can share databases with other BMPs and MPRs that run in the same IMS system without needing to implement data sharing.
- If there is only one IMS CTL region in the environment, the internal IMS lock manager is sufficient without the need for IRLM.
- Use IMS facilities such as logging, database buffering, automatic backout, managing database access, and so on.
- Can read and write to z/OS sequential files with integrity by using GSAM.

Transaction-oriented BMPs

Transaction-oriented BMPs are not scheduled by IMS as with MPPs. They are scheduled as needed and started with JCL. Transaction-oriented BMPs can:

- Read and process messages from the IMS message queue.
- Optionally write (insert) messages to the IMS message queue to be processed later by another application program.

As an MPP processes each message, it might send an output message that gives details of the transaction to the message queue. A transaction-oriented BMP could then access the message queue to produce a daily activity report.

Typically, you use a transaction-oriented BMP to simulate direct update online: Instead of updating the database while processing its transactions, an MPP sends its updates to the message queue. A transaction-oriented BMP then performs the updates for the MPP. You can run the BMP as needed, depending on the number of updates. This improves response time for the MPP, and it keeps the data current. This can be more efficient than having the MPP process its transactions if the response time of the MPP is very important. However, one

disadvantage in doing this is that it splits the transaction into two parts, which might not be necessary.

BMPs can process transactions that are defined as wait-for-input (WFI). This means that IMS allows the BMP to remain in virtual storage after it has processed the available input messages.

Batch-oriented BMPs

A batch-oriented BMP performs batch-type processing in any online environment. A batch-oriented BMP:

- Cannot read the message queues for input. They are simply batch programs that run in the online system.
- Can optionally write (insert) messages to the IMS message queue to be processed later by another application program.

A batch-oriented BMP can be simply a batch program that runs online. (Online requests are processed by the IMS DB/DC, DBCTL, or DCCTL system rather than by a batch system.) You can even run the same program as a BMP or as a batch program.

To use batch-oriented BMPs most efficiently, avoid a large amount of batch-type processing online. If you have a BMP that performs time-consuming processing such as report writing and database scanning, schedule it during non-peak hours of processing. This will prevent it from degrading the response time of MPPs.

Java Batch Processing (JBP) regions

Java Batch Processing (JBP) regions are similar to batch-oriented BMPs, but have JVMs associated with the region. JBP applications must be written in Java, object-oriented COBOL, or object-oriented PL/I. Like BMP applications, JBP applications can use symbolic checkpoint and restart calls to restart the application after an ABEND.

Note that, processing types supported by batch-oriented BMPs equally apply to JBPs.

2.6 Management layer

In the journey toward RAS (Reliability, Availability, and Serviceability), IBM's management products contribute the baseline capability: *automated detection and fixes for error conditions*. Furthermore, the IBM products actually help to prevent problems before they become error conditions. Also, the entire system management team benefits from a powerful graphical user interface.

Product	Role
IBM System Automation for z/OS	Automation can be triggered at the moment that an error condition is detected. But that is only the beginning of automation's contribution to RAS. ► **Range:** The range of actions that your automation solution offers defines how many situations it can handle. Consider the range of potential tasks: startup, monitoring, recovery, and shutdown of z/OS subsystems, components, applications, UNIX (USS; UNIX System Services), and sysplex resources. ► **Type:** Scripts are often used to implement automation. But this approach is brittle (not adaptable) and requires deep expertise to code. This IBMproduct offers two alternatives to scripts: – **Policy-based automation:** You base your automation on policies written by non-programmer members of your team. The key knowledge that is required is familiarity with the system and its dependencies. – **Goal-driven automation:** The operator chooses an initial management action. Then, the automation takes a set of actions that are required to ensure the success of the operator's initially chosen action. This set of necessary actions is performed in the logical sequence that the system dependencies require. IBM System Automation for z/OS offers system-level automation at the processor level. As a result, the system management team works more efficiently.
IBM Z Operations Analytics	Automation makes a powerful contribution to management, but what about prior detection and prevention of problems? With IBM Z Operations Analytics, you use a friendly graphical user interface to search, visualize, and analyze large amounts of structured and unstructured operational data across IBM Z environments. You detect and prevent problems. Furthermore, the insights from analytics teach you about your system and its trends. Most system managers have seen new patterns that emerge that seem to be error conditions but are actually new patterns that the system is adapting to. Analytics can anticipate new patterns that might trigger alarms, but are normal events.

2.6.1 IBM System Automation for z/OS

IBM System Automation for z/OS (SA z/OS) is an IBM NetView®-based application that is designed to provide a single point of control for a full range of systems management functions. SA z/OS functions include monitoring, controlling, and automating a large range of system elements that span both the hardware and software resources of your enterprise. System Automation for z/OS is a policy-based and goal-driven, high-availability solution to maximize efficiency and availability of critical systems and applications and to reduce operations efforts.

The automation product has two key components that are designed to automate processor and system operation. These components are called Processor Operations and System Operations.

The System Operations component automates many system console operations and selected operator tasks such as startup, monitoring, recovery, and shutdown of z/OS subsystems, components, applications, UNIX (USS), and sysplex resources.

Applications can be started or stopped by using timers, events, triggers, or service periods. Dependencies can be automatically established.

Resources can be member of groups, which can greatly reduce the complexity of automation definition and operations. Server groups control *which* and *how many* group members are started. Group members can represent, for instance, application servers.

Controlling application move and server management is done with preference values. Preference values range from 0 through 3200. They express the importance of each individual member of the group.

Note that only MOVE groups and SERVER groups support preference values. With BASIC groups you cannot specify preferences, because all members need to be available. For MOVE and SERVER groups, the preference defines which resource members of the group should preferably run. Remember that a MOVE group is designed for backup solutions. A MOVE group can have many components, but one and only one is supposed to be active. All other components are for backup only. Thus MOVE groups usually contain members of one kind (where one application can replace the others).

SERVER groups are created when only a certain number of their resources need to be available for the application group to be available. The number of desired available resources is controlled by the Availability Target parameter in the customization dialog box. You can use the preference value in the customization dialog box to control which resources that you want to be available and which you want to keep as backups.

Thresholds can be defined to alert operators or stop recovery.

Various active monitors are available, including a fast control block scanner and a monitor for UNIX resources. You can specify your own monitor and interval. Monitor resources can determine the health status by using external monitors or IBM OMEGAMON®. The IBM OMEGAMON software family provides real-time and historical performance management for your IBM Z environment and key subsystems. See “IBM OMEGAMON product family” on page 85.

All that you must do is specify to System Operations in an ISPF dialog box what resources you want to automate and monitor, and what your policies are for automation and monitoring. The automation engine, which is identical on all systems, will then start, monitor, recover, and shutdown resources and complex applications according to your configuration and goals. At any time, the operator can monitor and control from an enterprise-wide single point of control at the application level.

Processor Operations helps operators manage more systems with greater efficiency. This means that even from a remote location, one operator can configure, initialize, monitor, shut down, and recover a central processor complex (CPC), Logical Partitions (LPAR), and multiple systems in parallel, and also respond to a variety of detected conditions. Using one standard interface, an operator can do all that across multiple types of systems such as z/OS, VSE, z/VM, and Linux. In addition, automated routines for redundant tasks speed the work of skilled operators and assist less experienced operators in becoming more productive. Processor Operations is a focal point function.

Policy-based automation

Policy-based automation is a sophisticated methodology that allows to easily incorporate business goals into an automation framework. As the term implies, policy-based automation uses simple policy statements, with no program scripting or special education required.

Policy-based automation includes resource information, groups of resources, and relationships in the decision-making process before taking action. Resource information defines resource class and name; how to start, stop, and monitor the resource; and what the preferred systems are. Resources can be members of cluster-wide groups and relationships.

Policy-based automation can:

- Reduce the need to code rules and scripts that require special skills and are expensive to build and maintain
- Improve automation quality through consistent and reliable automation actions, whereas scripts might have coding errors and limitations and are hard to reuse
- Achieve higher automation levels that are next to impossible when using scripts

- Reduce time and effort to build and maintain automation
- Provide a web-based automation report

Goal-driven automation

With goal-driven automation operators can change a goal of resource A to online, and the automation product figures out the required actions according to the components and the relationships. So, it might mean that resources on which resource A is dependent are started or stopped.

A command-driven automation would only issue that command and not deal with dependencies that are more complex than a basic parent/child relationship.

Goals are being remembered, whereas command-driven automation forgets that it has issued the command.

Goal-driven automation also weighs conflicting goals, for example from sources like service periods, scheduler, and so on. This type of automation also weighs conflicting dependencies.

Goals also include the preferred systems for a resource and allow exclusion of a system. The result is easy operations and avoidance of operator errors like shutting down something that is needed elsewhere.

Another place where the automation product really stands out is the fact that you can also set a resource to have a desired state of offline. In this scenario, the automation product makes sure that everything stops in the right order. But also, if something were to be manually started, the automation product would stop it, thereby restoring the desired state of offline.

Apart from the standard IBM System Automation for z/OS product, there are other variants of this product that are available:

- **Automation Control for z/OS:**
  This is a single package that inherits functionality from IBM's premier automation products: IBM NetView for z/OS, which provides the automation platform. Also, it inherits System Automation for z/OS, which provides solutions for automated operations of both software and hardware. It can be used to automate, monitor, and control single system IBM z/OS software environments and local IBM Z hardware resources, including the Hardware Management Console (HMC). By monitoring IBM Z applications, hardware components, automated processes, messages, and alerts, you can respond to outages before they impact business users.
- **1) System Automation for GDPS/PPRC HyperSwap Manager** and
  2) System Automation for GDPS/PPRC HyperSwap Manager with NetView:
  Both of these are only available in combination with GDPS/PPRC HyperSwap Manager. They provide a limited-use version of IBM Tivoli® System Automation for z/OS (SA z/OS). This saves you the cost of purchasing a full SA z/OS license because it is enough to give you all the functionality that is needed to implement a GDPS/PPRC HyperSwap Manager solution for data failover and recovery within a single sysplex.

  For more information, see *IBM GDPS: An Introduction to Concepts and Capabilities*, SG24-6374: https://www.redbooks.ibm.com/redbooks/pdfs/sg246374.pdf

2.6.2 IBM Z Operations Analytics (IZOA)

IZOA is a tool that enables you to search, visualize, and analyze large amounts of structured and unstructured operational data across IBM Z system environments, including log, event, and service request data and performance metrics. Identify issues in your workloads, locate

hidden problems, and perform root cause analysis faster. These days, complex, integrated data centers require a team of experts to monitor for abnormal behavior in everything that is running. The experts must not only diagnose but also fix anomalies before they result in failures and outages that are visible beyond the data center. These tasks are costly and difficult for many reasons, including the fact that various everyday changes can cause system anomalies. The IBM z Advanced Workload Analysis Reporter (IBM zAware) provides a smart solution for detecting and diagnosing anomalies in z/OS and Linux on Z. IBM zAware creates a model of normal system behavior based on prior system data and uses pattern recognition techniques to identify unexpected messages in current data from the systems that it is monitoring. This analysis of events provides nearly real-time detection of anomalies that you can easily view through a graphical user interface (GUI). You also use the GUI to diagnose the cause of past or current anomalies.

IZOA also has ability to integrate logs from different sources, including application logs. As a result, you get a full view of an error scenario, almost end-to-end, as opposed to just the z/OS or Linux on Z system side.

For more information, refer to this website:
https://www.ibm.com/us-en/marketplace/operations-analytics-for-z-systems

2.6.3 IBM OMEGAMON product family

IBM OMEGAMON delivers performance and availability monitoring for IBM Z resources and workloads across the enterprise. Utilize product provided alerts to reduce potential delays and outages by monitoring near real-time metrics, identifying root-cause with bottleneck analysis, and executing remediation with reflex automation. Access coupling facility structure statistics, shared-queue counts, and database lock conflicts in a Parallel Sysplex with a single point of truth.

For further details about this product offering, see:
https://www.ibm.com/us-en/marketplace/omegamon-for-ims-on-zos

2.6.4 CICS tools

There are several CICS-related tools that can help improve or maintain resiliency. These include the following tools:

- CICS VSAM Recovery for z/OS
- CICS Interdependency Analyzer for z/OS
- CICS Performance Analyzer for z/OS
- CICS Configuration Manager for z/OS

CICS VSAM Recovery for z/OS

CICS VSAM Recovery (CICS VR) can help recovery of CICS and batch VSAM data from physical or logical corruption. It enables fast recovery from errors, reduces the risk that offline processing might exceed its batch window, and improves the availability of online CICS systems.

For more information see: https://www.ibm.com/us-en/marketplace/cics-vsam-recovery

Or see *CICS VR Version 4*, SG24-7022:
http://www.redbooks.ibm.com/abstracts/sg247022.html?Open

CICS Interdependency Analyzer for z/OS

The CICS Interdependency Analyzer (CICS IA) is a runtime tool for use with CICS Transaction Server for z/OS. CICS IA allows both system programmers and application developers to get an understanding of the relationships and dependencies of your CICS applications and the environment on which they run. By analyzing data collected by CICS IA, you can make changes to your environment in a safe and controlled but timely manner to address changing demands on your business applications.

For more information see:
https://www.ibm.com/us-en/marketplace/cics-interdependency-analyzer

Or to *CICS Interdependency Analyzer*, SG24-6458
http://www.redbooks.ibm.com/abstracts/sg246458.html?Open

CICS Performance Analyzer for z/OS

CICS Performance Analyzer (CICS PA) is a powerful offline reporting tool to help you tune and manage your CICS systems. CICS PA reports on the performance of CICS applications and systems. These reports help to identify and eliminate the causes of online performance issues, tune CICS systems for optimal performance, and analyze trends for capacity planning and performance bottlenecks. CICS PA can also help to improve CICS resource usage, evaluate the effects of CICS system tuning efforts, improve transaction response time, and increase availability of resources.

For more information see:
https://www.ibm.com/us-en/marketplace/cics-performance-analysis-tool

https://www.ibm.com/support/knowledgecenter/SSPPU4/welcome.html

Or to *CICS Performance Analyzer*, SG24-6063:
http://www.redbooks.ibm.com/redbooks/pdfs/sg246063.pdf

CICS Configuration Manager for z/OS

CICS Configuration Manager (CICS CM) is a tool for administering and maintaining CICS resource definitions. It provides audit, reporting, and lifecycle change management control facilities to support the build, management, and deployment of complex mainframe CICS applications.

For more information see:

- https://www.ibm.com/us-en/marketplace/cics-configuration-management-tool
- https://www.ibm.com/support/knowledgecenter/SS2L7A/welcome.html

2.6.5 IMS tools

There are several IMS-related tools that can help improve or maintain resiliency. These include the following tools:

- IMS Connect Extensions for z/OS
- IMS Queue Control Facility for z/OS
- IMS Database Administration and Autonomics
- IMS Program Restart Facility for z/OS
- InfoSphere IMS Replication for z/OS
- IMS Recovery Expert for z/OS

IMS Connect Extensions for z/OS

IMS Connect Extensions for z/OS (also referred to as IMS Connect Extensions) is a tool that enhances the operation of IMS Connect. It consists of components that run with IMS Connect; journal data sets that record IMS Connect activity; and ISPF, z/OS Explorer, and Rexx interfaces to manage IMS Connect systems and their IMS Connect Extensions features.

IMS Connect Extensions rules-based routing responds to changes in workload and processing capacity. IMS Connect Extensions can be used to schedule peak and off-peak routing rules, to manage data stores that are experiencing flood conditions, to deliver uninterrupted service during planned and unplanned data store outages, and to redistribute workload after services have been restored.

IMS Connect Extensions provides dynamic workload management of TCP/IP transactions allowing a user to:

- Define routing rules to automatically balance and distribute workloads and reroute messages when IMS system failures occur.
- Temporarily suspend routing of messages to an IMS data store to reduce the likelihood of transactions failing or being rejected when IMS is shut down.
- Assign selected OTMA routing rules to a routing plan and dynamically switch between routing plans.
- Automatically respond to changes in the IMS environment such as dynamically added IMS data stores and flood conditions.
- Automatically respond to the failure of an IMS Connect system by balancing the number of TCP/IP sessions that are directed to each remaining live IMS Connect system.

For further details about this product offering, see:

https://www.ibm.com/us-en/marketplace/ims-connect-extensions-for-zos

IMS Queue Control Facility for z/OS

IMS Queue Control Facility, or QCF, is an IMS message-queue management tool that manipulates the local, live IMS message queues in both the shared and nonshared-queue environments. IMS Queue Control Facility can:

- Requeue unprocessed messages from your IMS shared or nonshared-queue system onto the IMS message queues after an IMS cold start. This capability can be used to reduce the amount of time it takes to bring IMS back online. It also allows for IMS to be cold started, should an emergency restart fail, without losing previously unprocessed messages.
- Protect your IMS nonshared-queue system from a queues-full condition (U758 abend) with queue overflow protection. IMS Queue Control Facility implements queue overflow protection by monitoring the message queue usage. IMS Queue Control Facility analyzes the message queue usage to determine when to send an alert or to take action to prevent the excessive queue usage from continuing.

You can configure queue overflow protection by providing various parameters to specify partition values, set queue usage limits, and specify appropriate alerts and actions.

IMS Queue Control Facility provides three methods for implementing queue overflow protection. Each subsequent method provides more capability and flexibility in configuring queue overflow protection.

Type 1

Queue space notification exit (DFSQSPC0), which is available in an IMS shared and nonshared-queue system

Type 2

Queue overflow protection that uses threshold settings to define queue space partitions in your IMS nonshared-queue system

Type 3

Queue overflow protection that uses area and fail-safe settings to define queue space partitions in your IMS nonshared-queue system

For further details about this product offering, see:
https://www.ibm.com/us-en/marketplace/ims-queue-control-facility-for-zos

IMS Database Administration and Autonomics

IBM offers three database solution packs that provide tools to help manage and administer IMS databases:

- IBM IMS Fast Path Solution Pack for z/OS for users with Fast Path databases
- IBM IMS Database Solution Pack for z/OS for users with full-function databases, including High Availability Large Databases (HALDBs), that have strict 24x7 availability requirements
- IBM IMS Database Utility Solution for z/OS for users with full-function databases, including HALDBs, who have periodic downtime to be able to perform database maintenance

These solution packs provide extensive and easy-to-use utilities to help database administrators (DBAs) analyze, maintain, and tune IMS databases, including features that help boost system availability.

These solution packs provide the following elements that simplify management of IMS databases, improve the performance of applications accessing the IMS data, and prevent potential database outages:

- A solution that helps to keep databases in optimal condition by automating database monitoring tasks.
- A database reorganization solution that supports both an offline, single-step database reorganization process and an online, single-step database reorganization process that can accomplish database maintenance tasks with minimal database downtime.
- A database optimization solution that helps DBAs analyze IMS databases and build the most effective database organization and definition.

What is autonomics

Autonomic Computing concept arose in early 2000s in the IT industry. It envisioned computer systems capable of self-management or self-healing.

In the Autonomic Computing architecture (see Figure 2-11 on page 89), the system management cycle consists of the following four steps:

- **Monitor** - mechanisms that collect data from a managed resource
- **Analyze** - mechanisms that correlate and model complex situations
- **Plan** - mechanisms that construct the actions needed to achieve goals and objectives
- **Execute** - mechanisms that control the execution of a plan with considerations for dynamic updates

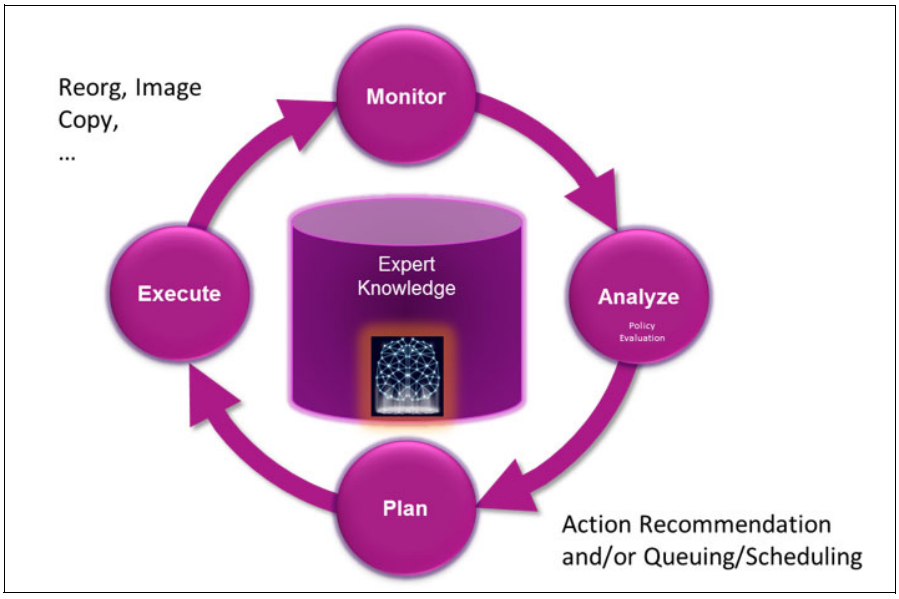


Figure 2-11 Autonomics system management cycle

Another key component is knowledge (or know-how). Data is used by the autonomic four functions. This can include historical logs, symptoms, and policies.

The database solution packs, in conjunction with IBM Tools Base for z/OS, provide the autonomic capabilities for database administration that not only lightens the DBA's workload but also keeps databases safe.

Automating database monitoring

DBAs assess database conditions by analyzing various reports that are generated by the tools or programs used in their shops. With any of the database solution packs, database statistics are collected automatically during the DBA's daily or weekly operations, and are accumulated in a repository. The repository is provided by a component in the IBM Tools Base known as IMS Tools Knowledge Base, or ITKB.

The accumulated statistics are automatically evaluated from various perspectives, and DBAs are alerted if databases are in exceptional conditions. When database reorganization is needed, reorganization tasks are automatically scheduled. The evaluation of the statistics and any alerts or notifications is provided by a component of IBM Tools Base referred to as Policy Services.

IBM Autonomics Director is another component in IBM Tools Base. It interfaces with the other components of the database solution packs to provide the capability to automatically detect the degradation of application performance and to proactively execute maintenance utilities that help to keep your data accessible. Autonomics Director can also save CPU time and help free your DBA staff to focus on more important work. The Autonomics Director helps maintain the health, performance, and recoverability of IMS databases.

For further details about these product offerings, see:

- https://www.ibm.com/us-en/marketplace/ims-database-solution-pack
- https://www.ibm.com/us-en/marketplace/ims-database-utility-solution
- https://www.ibm.com/us-en/marketplace/ims-fast-path-solution-pack
- https://www.ibm.com/us-en/marketplace/tools-base-for-zos

IMS Program Restart Facility for z/OS

IBM IMS Program Restart Facility helps you to correctly restart abended IMS batch jobs that use the IMS extended restart facility. IMS Program Restart Facility puts the checkpoint records that are required to restart jobs into DASD files called checkpoint tracking data sets (CTDS). When you restart a job that has abended, IMS Program Restart Facility automatically supplies the most recent restart checkpoint ID. The CTDS contains copies of the IMS log records that are required for an extended restart. IMS Program Restart Facility can also perform IMS batch backout processing when DL/I or DBB IMS batch jobs fail. When a DL/I or DBB batch job abends, IMS Program Restart Facility closes the log of the DL/I or DBB batch job and invokes the IMS batch backout utility. This time-critical process is required to free any IRLM locks that might be held by an IMS batch job. If a system failure occurs, IMS Program Restart Facility uses a batch backout data set (BBDS). This is a small DASD file that tracks DL/I and DBB batch jobs, to determine the status of the job. It performs the required IMS log close and batch backout when the job is resubmitted.

For further details about this product offering, see:
https://www.ibm.com/us-en/marketplace/ims-program-restart-facility-for-zos

InfoSphere IMS Replication for z/OS

You can use IMS Replication to produce copies of your IMS databases and maintain current data in near-real time, typically on geographically dispersed IMS subsystems. It addresses your organizational requirements for reliable and available data: High availability and disaster recovery (HADR), Workload distribution, Business intelligence, Site/database maintenance and Data backup. IMS Replication can capture changes from a single DB/DC or DBCTL subsystem. It can also capture changes from hundreds or thousands of shared databases in an IMS data-sharing environment. You can capture updates from multiple IMS DL/I batch jobs, DB/DC subsystems, or DBCTL subsystems in a sysplex. IMS Replication also supports high-availability environments that implement FDBR (Fast Database Recovery) regions. IMS Replication has the following key components and capabilities: Subscriptions, Transactional consistency, Replicating historical changes, and Monitoring and reporting.

For further details about this product offering, see:
https://www.ibm.com/support/knowledgecenter/en/SS4T2J_11.3.0/home/product_landing.html

IMS Recovery Expert for z/OS

IMS Recovery Expert for z/OS is part of IMS Recovery Solution Pack and helps you to avoid accidental data loss or corruption by providing the fastest, least costly method of backup and recovery. IMS Recovery Expert provides a fast and easy-to-use implementation of an IMS System Level Backup and recovery methodology. It reduces backup windows by leveraging storage-based fast-replication so that backups of multi-terabyte databases can be performed in seconds or less. It simplifies backup and recovery methodologies by allowing full-system, application, and database-level recoveries to be performed from a common System Level Backup. Consistent backups can be created by using full or data-only System Level Backup options. It provides IMS System Level Backup and recovery support even for complex applications, when data must be backed up, restored, and recovered as a unit. System Level Backups can be taken while the IMS system remains active. In addition, when creating System Level Backups, IMS Recovery Expert invokes storage-based fast-replication facilities through appropriate storage processor APIs. This reduces host CPU and I/O resource utilization. Legacy data copy methods can also be used to create System Level Backups while the IMS system is down.

IMS Recovery Expert has integrated, intelligent recovery and disaster recovery managers that analyze recovery assets and establish optimal recovery procedures to minimize recovery

time and recovery point objectives. Recovery jobs are tailored specifically to available backup and hardware resources.

- The Intelligent Recovery Manager supplies the ability to perform local recoveries efficiently by using all available recovery resources. Restore operations that invoke fast-replication facilities through appropriate storage processor APIs and parallel recovery can significantly reduce recovery time and complexity.
- The Intelligent Disaster Recovery Manager uses local site procedures to prepare for offsite disaster recovery or disaster restart in advance. The information that is acquired allows Intelligent Disaster Recovery Manager to intelligently perform remote site restoration operations and appropriate recovery or restart procedures.

For further details about this product offering, see:
https://www.ibm.com/support/knowledgecenter/en/SSS8TR_2.1.0/com.ibm.imstools.frx.doc/frxhome.htm

3

IBM Z resiliency models

The deployment of IBM Z resiliency can be regarded as a journey toward reducing risk and impact on service delivery. For example, businesses might start with a single system configuration and as they grow, system complexity increases. So, they embark on a journey to make their critical applications more resilient to outages (planned or unplanned), and eventually move towards achieving continuous availability.

This chapter describes four common deployment models that go through the various phases of that IBM Z resiliency journey (see Table 3-1), with a goal of fulfilling RPO and RTO objectives. Use cases for planned and unplanned outages of each deployment model are provided. The use cases describe the interactions of hardware and software components within the deployment model to illustrate how the highest possible level of resiliency can be achieved. Be sure to review 2, “Resiliency is built into the IBM Z platform” on page 15 for details on the hardware and software components that support these models.

Table 3-1 Deployment models

Section	Description
3.1, Model 1: Starting with a resilient and reliable base	The starting point is a single IBM Z system, which is based on the IBM RAS (Reliability, Availability, and Serviceability) design principles. These principles guide the system architecture to ensure resilience and reliability.
3.2, Model 2: Reducing the duration of outages with failover capability	Add failover capabilities to the IBM Z environment. Furthermore, your system can ensure fast data replication with the help of synchronous “Metro Mirror” for shorter distances and “Global Mirror” for longer distances. Mirroring can be enhanced with the help of IBM Copy Services Manager (CSM).
3.3, Model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS	Increase the fault tolerance of your system by adding z/OS Parallel Sysplex data sharing and disk replication of your storage resources. In this way, you exceed the basic redundancy that a z/OS Parallel Sysplex provides. The automation features of GDPS further enhance management of CPC and site recovery actions.
3.4, Model 4: Adding GDPS Continuous Availability for maximal resilience	Enable your workloads to fail over to another Parallel Sysplex for planned or unplanned workload outages within seconds. The GDPS Continuous Availability concept consists of two sites. The sites can be separated by virtually unlimited distances and run the same applications with the same data sources. In this way, you maximize cross-site workload balancing, continuous availability, and disaster recovery.

© Copyright IBM Corp. 2020. All rights reserved.

3.1 Model 1: Starting with a resilient and reliable base

This deployment model describes an entry-level for resiliency that takes advantage of the base IBM Z hardware and software for applications that can withstand long recovery times in an outage event. However, businesses have additional opportunities to lower even more the possibility of planned and unplanned outages. These opportunities are part of the deployment models that are described later in this chapter.

3.1.1 Description

Resilience value and functionality at this level are based on workloads that run in single system IBM Z platform environment. This is coupled with some level of automation and backup for recovery.

For hardware events, remediation might take minutes and in rare cases, a few hours. One source of this legendary resilience lies in how IBM Z platforms can automatically fence faulty hardware and restart the remaining resources, so that the services can resume. The resilience depends only on the hardware. Both hardware and software have Single Points of Failure (SPoFs).

Spare processor capacity is available (CBU and OOCoD) for recovery. Data on disk is an SPoF while Tape Replication provides passive backup.

When the operations team faces an outage, disaster recovery is highly manual because data is likely restored from tape.

The resiliency value of this model is limited and the business impact is potentially high. Although you have business viability, you are missing several opportunities to reduce single points of failure during an outage. Your business impact is higher and more severe because of potential disruption that can last from minutes to hours with this deployment model. Therefore, your resiliency value for RTO and RPO is equal to or greater than 24 hours.

3.1.2 Considerations

With this deployment model, some requirements that allow you to meet your availability objectives include:

- Single system
- Either a monoplex or a Parallel Sysplex with resource sharing
- Manual recovery and backup
- In-house written automation
- Tape Data Replication
- FlashCopy

There are many functions within the IBM Z platform that provide base availability. Some include:

- Transparent CPU sparing
- Concurrent processor drawer repair
- Concurrently checkstop individual CPU without system outage
- Concurrent repair / add of all cards in an I/O cage
- Concurrent repair / add of I/O cage
- Concurrent repair of cooling pumps, blowers, and so on
- Concurrent repair of all power parts
- Concurrent system processor maintenance

- Concurrent Oscillator maintenance
- Concurrent memory repair / add
- Concurrent memory upgrade
- Concurrent driver upgrades
- Concurrent firmware updates for most fixes
- Concurrent LIC upgrades (CPU, LPAR, channels, HMC, and so on)
- Non-disruptive capacity backup
- Dynamic PR/SM changes
- N+1 power subsystems, N+2 on point of load cards
- Active thermal management
- RAIM technology for memory
- Transparent Oscillator failover
- System Recovery Boost

Every IBM Z hardware and software release continues to improve on quality while looking for new ways to improve availability. See 2, “Resiliency is built into the IBM Z platform” on page 15 for details on the resiliency capabilities.

3.1.3 Use case: planned outage

Typically, planned outages consist of hardware or software maintenance or upgrades. If properly assessed and coordinated, you can minimize the number and duration of planned outages. System Recovery Boost (details can be found in System Recovery Boost , “IBM Z System Recovery Boost” on page 21), can help reduce the shutdown and startup times.

Software maintenance and upgrades

When indicated, maintenance can be applied to z/OS and middleware libraries while z/OS continues to run, but will require a restart to pick up the changes. To apply changes to z/OS, you need to:

1. Quiesce the production environment.
2. Restart (recycle) z/OS while pointing to the library with the new code.
3. Restart the middleware and subsystems.

Note that for the z/OS operating system base, maintenance should not be installed into live, running libraries. Instead, you install it into an inactive set of libraries that you can activate on your running system according to the instructions that are provided in the service.

Automation can wait until dependent subsystems are up before starting the next subsystem. For example, CICS requires Db2 for connectivity to threads. As a result, automation can wait until Db2 is fully up before starting CICS regions. However, if CICS is started at the same time, then commands can be issued to connect to the Db2 threads.

Depending on the configuration, the IPL process can take several minutes.

If applying changes to just middleware such as Db2, then only that product needs to be recycled.

System maintenance and upgrades

Most IBM Z maintenance and LIC upgrades can be done dynamically while the system continues to run production workload. See “IBM Z platform” on page 117. Yet, there are some events that might require a planned outage. They can include an upgrade such as going to a new generation of the IBM Z platform, or planned maintenance to a CPC drawer if there was only one CPC drawer configured. User impact time can be minimized if there is another system where the LPARs can be restarted. A second planned outage must then take place to

move operating system (in other words, z/OS) back to the original location. This assumes the following conditions:

- Second LPAR has access to the data.
- Software is licensed to run on the second system.
- Second system has capacity to handle the workload of the LPAR moving over.

3.1.4 Use case: unplanned / unexpected outages

You have a limited level of protection if your system might fail for reasons such as:

- Unplanned power loss
- System checkstop
- Software failures
- Inability to quickly access data from your storage devices due to a broken link or network connectivity loss in your infrastructure.

Note that the Z hardware and firmware include extensive recovery, failover, and concurrent repair capabilities.

For hardware events, remediation might take minutes and in rare cases, a few hours. One source of this legendary resilience is how IBM Z platforms can automatically fence faulty hardware and restart the remaining resources, so that the services are restored. For disk failures, it can take days to recover the data. Then, you must ensure that the right data is restored. In short, you face a time-consuming, manual effort.

3.1.5 Resiliency optimization and next steps

The preceding use cases point to high risks of potentially broad impact for a business. The impact to the management system and business is much greater due to the need to locate old data copies, recover the data, create a consistent point in time copy of the data, and perform an emergency restart.

Remediation of hardware events might take just over 2 hours in the rare, worst-case scenario. The overall outage might last this amount of time, because both the hardware and software are potential single points of failure in this deployment model. Depending on the industry, even brief outages can cost a business millions of dollars. For information on mitigating your risk and improving your resiliency / continuous availability level, see Chapter 4., “Planning for IBM Z resiliency” on page 115.

If the limitations of this first deployment model will not help you meet your service level agreements, then you should assess how to best achieve the second deployment model. The resiliency features of the second deployment model include ways to mitigate planned outages, handle automated failover, and using data replication technology.

Key items for the next section include:

- System software and workload outages are mitigated with resource sharing in a Parallel Sysplex.
- Enablement of a spare CPC drawer for EDA & Flex Memory - a defected CPC drawer can be dynamically reconfigured to use a spare CPC drawer and brought back online.
- Exploitation of Global Mirror or Metro Mirror data replication with IBM Copy Services Manager (CSM).
- Exploitation of redundant network adapter and virtual addresses.

- Adoption of (synchronous) Metro Mirror data replication or (asynchronous) Global Mirror for longer distances, with IBM Copy Services Manager or with GDPS.
- z/OS HyperSwap to minimize disk-related outages.
- Work on improving systems management processes for orderly change and problem management.
- As part of your resiliency journey, you should think about how your enterprise can leverage capabilities on the IBM Z platform. For more information, see these resources:
  - https://www.ibm.com/it-infrastructure/z/capabilities/resiliency
  - https://www.ibm.com/it-infrastructure/z/capabilities/enterprise-security

3.2 Model 2: Reducing the duration of outages with failover capability

In the failover deployment model, single points of failure (SPoF) are eliminated for the hardware and software environment with multiple LPARs. If there is an outage, the failed LPAR is simply restarted on the stand-by LPAR. Middleware is restarted with in-flight units of work being backed out and completed units of work are recommitted (forward recovery) as part of the normal middleware emergency restart process. Ideally, the stand-by LPAR is on a separate IBM Z platform to protect against hardware events, but can be on the same IBM Z platform. Single points of failure for data can be mitigated by using Metro Mirror or Global Mirror technologies. If using Metro Mirror, you should also set up HyperSwap to provide rapid failover for disk, which can be either z/OS Basic HyperSwap, IBM Copy Services Manager, or GDPS automation.

3.2.1 Parallel Sysplex Resource Sharing

A number of base z/OS components use the IBM Coupling Facility (CF) shared storage to provide a medium for sharing information for the purpose of multi-system resource management. This exploitation, called Resource Sharing, enables sharing of physical resources such as logs, tape drives, consoles, catalogs, GRS, and so on. With significant improvements in cost, performance, or simplified system management. This is not to be confused with Parallel Sysplex data sharing by the database subsystems. Resource Sharing delivers immediate value even for those who are not leveraging data sharing, through native system exploitation delivered with the base z/OS software stack. You can benefit from Parallel Sysplex resource sharing even if you have multiple z/OS LPARs on a single IBM Z platform. For more information on Parallel Sysplex see “Multi-system environment” on page 28.

3.2.2 Description

The resiliency benefits with this deployment model include:

- Redundancy in hardware, software, and data
- Failover for applications and middleware
- Single point of control for many service management processes
- Data replication
- Optional use of HyperSwap to mitigate the risk of disk errors

With this model, you incur shorter recovery times for your single points of failures during an outage event. Your business is still exposed to unplanned and planned outages. In failover

situations, impact level can be reduced to minutes. Improved automation might give you recovery times of a few hours.

For more information on Parallel Sysplex architecture, see
https://www.ibm.com/it-infrastructure/z/technologies/parallel-sysplex

3.2.3 Considerations

Some requirements that help meet your recovery objectives include:

Functional requirements

- Single system with multiple LPARs (ideally two or more systems are preferred)
- Rapid Cross-System Restart
- Parallel Sysplex with resource sharing
- Partial Automation
- Capacity provisioning
- System Recovery Boost

Non-functional requirements

- One production site
- Data backup at the same or at a different location
- Software solution to manage data replication
  - z/OS Basic HyperSwap
  - IBM Copy Services Manager
  - GDPS automation

For more information on IBM Copy Services Manager (CSM), see
https://www.ibm.com/support/knowledgecenter/en/SSESK4_6.2.3/com.ibm.storage.csm.help.doc/frg_c_ugover.html

GDPS provides even more automation and management support. It is discussed in detail in “Model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS” on page 101.

3.2.4 Use case: planned outage

Typically, planned outages consist of hardware or software maintenance or upgrades. If properly assessed and coordinated, you can minimize the number and duration of planned outages. System Recovery Boost (details can be found in System Recovery Boost “IBM Z System Recovery Boost” on page 21), can help reduce the shutdown and startup times.

System maintenance or upgrades

IBM Z is equipped with a very resilient (hardware) system along with z/OS, which has a robust resilient software infrastructure. You can dynamically reconfigure the I/O definitions, create new LPARs, add more capacity, even perform LIC upgrades. This list goes on. However, there are times when a planned outage is required. This can include:

- System upgrade
  The production system is quiesced. It can then be restarted on the new IBM Z platform. Depending on the configuration, it can take several minutes to restart the z/OS. This requires software that is running to be licensed on the target system.
- Coupling Facility upgrade
  Because there are two CFs, you can dynamically move the CF structures to the CF not

being upgraded, then restart the CF being upgraded and dynamically move the structures back. This can be done without any outages.

- Operating System or middleware maintenance or upgrades
  Because software changes are not possible when active, this requires the software (z/OS, middleware, etc.) to be quiesced and recycled in place. This will incur several minutes of downtime.
- Disk reconfigurations
  If using IBM DS8K Metro Mirror with HyperSwap, the production disk can be rapidly switched to the backup devices. Depending on the configuration it can take approximately 3 - 20 seconds of user impact time. After the reconfiguration, a planned HyperSwap can be performed again to continue running on the (new) primary disk.

  If you use Global Mirror, a full-site takeover to the site that contains the Global Mirror secondaries is required.

3.2.5 Use case: unplanned outage

In the event of an unplanned outage, the impact to the management system and business can be mitigated by rapid failover and recovery. Automation should be used as much as possible to speed up the recovery process. Outages can be caused by:

- System outage
  LPARs can be restarted on the new system. Depending on the configuration, it can take several minutes to restart the z/OS. This requires software that is running to be licensed on the target system. The operating system and middleware is then restarted. In-flight units of work need to be backed out and completed units of work are then recommitted (forward recovery). Depending on how far back the last uncommitted work is, the restart process can take in the 10's of minutes. However, with the z15 or older system, System Recovery Boost can significantly reduce startup time.
- Coupling Facility failure
  Recovery is automatic and performed by z/OS. "Resource Sharing" structures are automatically moved to the surviving CF instance.
- z/OS failure
  The process is similar to a system failure, other than the LPAR can be restarted in place.
- Middleware failure
  z/OS continues to run but key middleware has failed. Some sites have a process in place to try to identify the reason for the failure before attempting a restart. However, in almost all cases the middleware will be able to restart immediately. It is recommended that automation be put into place to attempt the restart right away. Only if this restart failures should the debug process start. This can easily be managed with the z/OS Automatic Restart Manager (ARM) or with other automation utilities being run.
- Disk failure / boxed device / offline device
  If using Metro Mirror with HyperSwap, HyperSwap will automatically detect these errors and perform a HyperSwap. Optionally, you can trigger a HyperSwap when the z/OS Missing Interrupt Handler (MIH) messages are produced.
- If using Global Mirror, a disk failure would require invoking a disaster recovery to the site with the secondary disk. This can take one or more hours to recover.
- Tape failure
  Use a tape grid - two or more physically separate IBM TS7700 clusters connected to one another via an IP network.

Risk with SPOFs that are spaced 10 KM or less (most common scenario)

There is risk associated with SPOF over 10 km because recovery depends somewhat on partial automation. However, it is exposed if data is not replicating because of these factors:

- More manual than automated recovery
- Recovery from GM replica
- Cold DR site
- Recovery from Safeguarded copy

Under this deployment model, you can handle outages if they are infrequent. However, businesses are still vulnerable to planned and unplanned outages.

3.2.6 Resiliency optimization and next steps

In this deployment model, your degree of resiliency is *Medium*, and there are several potential stages in your journey toward resiliency. A big improvement awaits your business in the form of Continuous Availability (CA), which greatly reduces the risk and impact that unplanned outages can cause. With CA, your business runs two or more systems, multiple LPARs, shared DASD in a Parallel Sysplex to mitigate outages, handles automated failover, and exploits data replication technology.

Key themes and terms from the next deployment model are listed here:

- Workload distribution
- Dynamic workload routing to regions on other systems in the Sysplex
- Data sharing across members of the sysplex
- Options to fail over to surviving hardware
- Recovery point with proper capacity required or CBU Activation
- Global Mirror or Metro Mirror data replication with GDPS automation
- HyperSwap required for fast recovery from DASD errors
- Synchronous/asynchronous replication to DR site
- High Availability Tape grid
- Fast recovery of data in under 2 hours, as opposed to much longer intervals
- Systems management, Automation to restart functions at machine speeds
- Use ARM or automation to take action, instead of a manual process to determine what's occurring and respond
- Auto-detect/auto-restart allows system to be up in 20 - 30 minutes in some scenarios
- Automate DR for/in the Sysplex
- Consider the next step in your journey with Continuous Availability with built-in resiliency at each layer:
  - **Systems:** IBM Z hardware, z/OS resiliency features, and Linux for IBM Z resiliency
  - **Data:** Resiliency Storage Solutions with IBM DS8K, IBM Copy Services Manager (CSM), z/OS HyperSwap, and GDPS automation
  - **Applications:** Application Availability with z/OS Availability Management, CICS Applications, IBM MQ Availability on z/OS, IMS, and so on
  - **Operations:** Operations Efficiency with Parallel Sysplex, IBM System Automation for z/OS and z/OS Management Facility (z/OSMF)

3.3 Model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS

The third deployment model takes advantage of Parallel Sysplex with data sharing and GDPS Metro Mirror with HyperSwap to achieve fault tolerance. With these features, systems immediately recover from hardware and software events as well as from storage I/O failures with potential zero data loss.

GDPS Metro configuration types

GDPS Metro provides four typical configuration options:

- Single Site Workload
- Multiple Site Workload
- BRS Configuration
- GDPS Metro dual-leg configuration

For an overview of GDPS, see these sections:

- “Model 3: Reducing the impact of outages with a fault tolerant architecture and GDPS” on page 12.
- “Model 4: Adding GDPS Continuous Availability for maximal resilience” on page 13.
- “Multi-site environment” on page 30.

We focus on GDPS Metro multi-site workload configuration because it is the most commonly used solution for this deployment model. The production systems are running in two (or more) sites.

3.3.1 Description

GDPS Metro supports both planned and unplanned outages, helping to maximize application availability and providing business continuity. GDPS Metro solution delivers the following possible benefits:

- Near-continuous availability
- Disaster recovery (DR) across metropolitan distances
- Automatic recovery for failures within the Parallel Sysplex
- Automated recovery for disk failures
- Recovery time objective (RTO) less than an hour
- Recovery point objective (RPO) of zero

3.3.2 Considerations

The considerations for this deployment model are as follows:

- Distance and connectivity
- Sysplex
- Controlling system
- Recovery point objectives

Distance and connectivity considerations

Connectivity between the primary and secondary disk subsystems can be provided by direct connections between the primary and secondary disk subsystems, by IBM FICON switches, by DWDMs, and by channel extenders.

As the distance between the primary and secondary disk subsystems increases, the time it takes for the data to travel between the subsystems also increases.

The distance between sites that a system can tolerate depends on the application workloads and service level requirements. Each enterprise must test with its own applications and workloads to determine the maximum distance that it can achieve. While single-site workload can be run spanning 100 km or more with RPQ, typically multi-site workloads are limited to 25 km due to latency. More information on this can be found in “Considerations for Multisite Sysplex Data Sharing” http://www.redbooks.ibm.com/abstracts/sg247263.html?Open

Parallel Sysplex considerations for GDPS

GDPS Metro uses XCF signaling for communication of status, information, errors, and for routing certain commands among members of the GDPS Metro complex. All systems in your Parallel Sysplex must run in the same GDPS Metro configuration to ensure that critical events are captured, communicated, and coordinated properly.

Under normal circumstances, the active primary and active alternate couple data set (CDS) for each of the CDS types should be split across the two sites. For example:

- The CDS in Site1 is the primary.
- The CDS in Site2 is the alternate.

The CDSs are not replicated by Metro Mirror or Global Mirror as the alternate is always an image of the primary.

Controlling system considerations

GDPS requires at least one separate, isolated, z/OS system known as the controlling system. This system must be a member of the same Parallel Sysplex as the production systems that it is associated with. And the system must have connectivity to all the Site1 and Site2 Metro Mirrored disks that it will manage. Here are some further guidelines:

- For redundancy reasons, two controlling systems should be set up, one in each site.
- The controlling system must not share any infrastructure with the production systems except for the Couple Data Sets.
- If available, it is preferable to isolate the controlling system infrastructure on a disk subsystem that is not housing GDPS-managed Metro Mirrored disks
- GDPS environments without a controlling system are not supported.

Recovery point objective (RPO) considerations

GDPS Metro with HyperSwap can achieve zero data loss. Because Metro Mirror uses synchronous remote copy, any updates done to the primary disk are immediately forwarded to the secondary disk. When the data is committed to cache at the secondary, an acknowledgment comes back to the primary, the primary disk signals that the I/O is completed, and application processing can continue. In this manner, the secondary disk always contains the data the primary site.

If there is a problem in the relationship between a primary and secondary disk device, the freeze function is initiated. This can be caused by a communication failure, a problem with one or the other disk devices, or the beginning of a site outage. The freeze function is designed to freeze the image of the secondary data at the very first sign of a disaster, even before any database managers are made aware of I/O errors. This can prevent the logical contamination of the secondary copy of data. Such contamination occurs if any storage subsystem mirroring were to continue after a failure that prevents some, but not all secondary volumes from being updated. Depending on the error, you have the option of either continuing to run on the primary disk only, perform a HyperSwap and run on the secondary disk only, or

perform a HyperSwap and switch sites. Note that if you continue to run on the primary disk only, the secondary disk will be time consistent but will not contain current data if a site failure later occurs.

3.3.3 Multi-site workload configuration

The multi-site sysplex is configured with redundant hardware, including Z platforms, coupling facilities, disk, and GDPS control systems. Server Time Protocol (STP) spans the environment to provide a single time source. Ideally, an Arbiter is configured for additional protection.

As Figure 3-1 depicts, LPARs P1 and P2 are in the production Parallel Sysplex, as are the coupling facilities CF1 and CF2. GDPS controlling system K1 is running in Site2 and K2 is running in Site1. We also have backup LPARs in each site for the production system in the opposite site, which are defined as inactive. There are two GDPS controlling systems defined, one in each site. The Control System K1, running on “Site 2” with the secondary disk, is active to assist in HyperSwap and failover automation if there is a Site 1 problem. K2 can be started up after a takeover. System Managed CF Duplexing is not required, but does improve availability for structures after a CF failure.

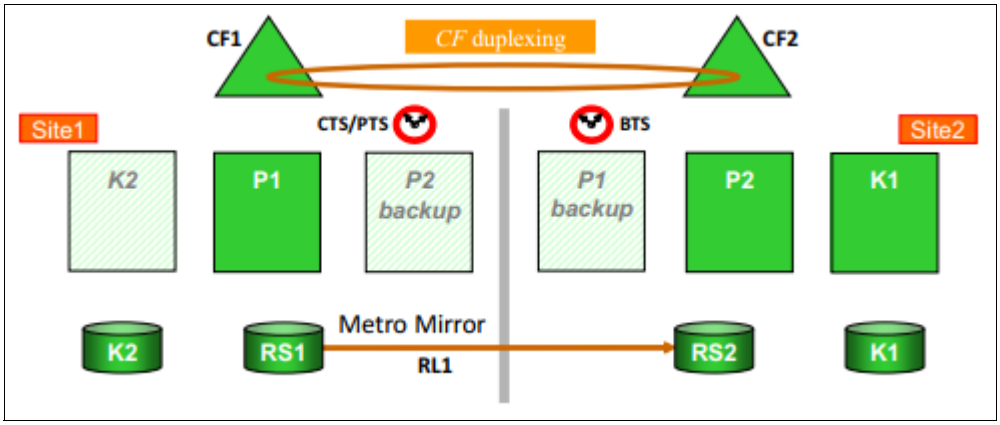


Figure 3-1 GDPS Metro multi-site workload configuration

Here controlling system K1 is in Site2 and controlling system K2 is in Site1 with their own local (not mirrored) disks. This provides the capability to recover from an outage in either site.

If there is an outage that disables either site, a GDPS controlling system is still available to decide how to react to that failure, help determine what recovery actions are to be taken, and manage the HyperSwap and automation to recover.

Depending on your availability requirements, specific GDPS policies should be in place to handle different types of outages.

- For Metro Mirror failures:

  PPRCFAILURE=GO

  GDPS allows production systems to continue operation after mirroring is suspended.
- For primary disk failures:

  PRIMARYFAILURE=SWAP

  After it performs the Freeze, GDPS proceeds with a HyperSwap. When the swap is complete, the systems are running on the new, swapped-to primary disks (the former secondaries).

3.3.4 Use case: planned outage

By going through the GUI, GDPS can perform tasks such as:

- Stop all Site1 images
- HyperSwap Metro Mirrored storage
- Update STP roles
- Create new alternate CDSs
- Move the CF structures at Site 1 into the Site2 CFs
- Activate CBU if needed
- IPL the production system on to Site2
- Respond to IPL messages

3.3.5 Use case: unplanned outage

Within a Parallel Sysplex, it is possible to construct a parallel processing environment with no single points of failure. All members in the Parallel Sysplex can have concurrent access to all critical applications and data. For this reason, the loss of a system due to either hardware or software failure does not necessitate loss of application availability. Cloned instances of a failing subsystem executing on remaining healthy system nodes can take over recovery responsibility for resources held by the failing instance. While the failing subsystem instance is unavailable, new work requests can be redirected to other data sharing instances of the subsystem on other cluster nodes. As a result, you provide continuous application availability across the failure and subsequent recovery. This provides the ability to mask planned as well as unplanned outages from the end user.

Full Site Failure: Site1

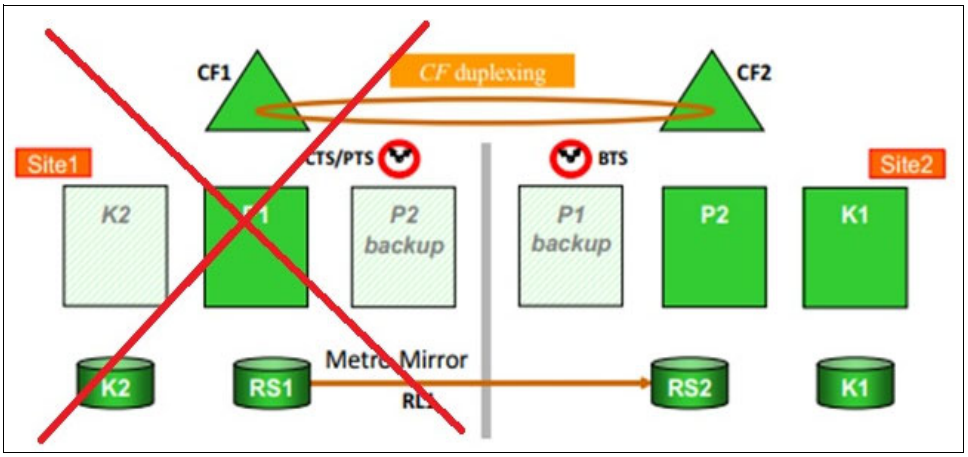


Figure 3-2 Site1 Failure in Multiple Site Workload configuration

In this Multi-site workload configuration, GDPS offers customizable takeover script to take over the failed site. The intent is to make resources available in Site2 and to restart all processing in Site2 as quickly as possible.

In this scenario, we make the following assumptions:

- The PTS/CTS are in Site1.
- The BTS is in Site2.
- Console Assisted Recovery informs Site2 about the failure of Site1.

GDPS can:

- The GDPS control system will detect the failure of Site1 either by XCF or the HMC (console assisted recovery).
- In many cases, be able to HyperSwap the disk so the LPARs on Site2 now use the disk at Site2.
- LPARs at Site2 continue running.
- GDPS invokes CBU to provide the Z platforms on Site2 with additional MIPS.
- GDPS manages the STP configuration to create a new Backup Time Server.
- GDPS creates new Alternate Couple Data Sets as needed.
- Database managers or the LPARs normally running on Site1 are restarted on Site2 to back out in-flight units of work and release “Retained” locks on CF structures.

3.3.6 Resiliency optimization and next steps

GDPS Metro multi-site workload configuration can achieve zero data loss but need to take the distance impact into account. GDPS Global - XRC and GDPS Global - GM, provide for effectively unlimited site separation. However, they require that the workload from the failed site is restarted in the recovery site. Typically, this takes more time. For this reason, the GDPS Continuous Availability concept was conceived to achieve these goals in software replication:

- Ensure that both sites are active to provide service.
- Ensure integration with other GDPS disk mirroring solutions to provide multiple sites continuous availability and disaster recovery.

3.4 Model 4: Adding GDPS Continuous Availability for maximal resilience

In this section, we discuss the fourth deployment model in the IBM Z Resiliency journey. This model takes advantage of the GDPS Continuous Availability solution. This solution can make both sites active with software data replication. The goal is to ensure continuous availability in planned and unplanned outage. As a result, you significantly reduce the time to recover systems in a disaster situation. And you enable planned and unplanned switching of workloads between sites.

3.4.1 Description

The goal if GDPS Continuous Availability (CA) is to have two sites with these characteristics:

- Separated by virtually unlimited distances.
- Run the same applications.
- Have the same data to provide workload balancing and continuous availability and disaster recovery to cross-sites.

GDPS CA allows the read/write workload to execute on only one site, although read-only workload can be run on either site.

GDPS CA solution brings these benefits:

- Ensure successful recovery through automated processes.
- Provide automatic workload distribution between sites.
- Provide application-level granularity in workload level.
- For unplanned outage, RPO can be impacted by end-to-end latency for data replication or RPO can achieve zero data loss with the Zero Data Loss (ZDL) solution. RTO can be in the range of minutes.

GDPS CA solution types

There are two types of GDPS CA solutions. One is the general GDPS CA solution, and the other is GDPS Zero Data Loss (ZDL) CA solution.

GDPS CA solution

The GDPS CA solution does not need to use any of the infrastructure-based data replication techniques that other GDPS products rely on, such as Metro Mirror (PPRC), Global Mirror (GM), or z/OS Global Mirror (XRC). However, the GDPS CA solution can also incorporate them for three or more data centers to ensure continuous availability and disaster recovery.

This solution relies on both of the following methods:

- Software-based asynchronous replication techniques for replicating data between sites such as Qrep for Db2 for z/OS data replication, VSAM replication, and IMS replication.
- Automation — primarily operating at a workload level — to manage the availability of selected workloads and the routing of transactions for these workloads. The GDPS Continuous Availability product based on Tivoli NetView for z/OS and System Automation acts primarily as the coordination point or controller for these activities. This activity is a focal point for operating and monitoring the solution and readiness for recovery.

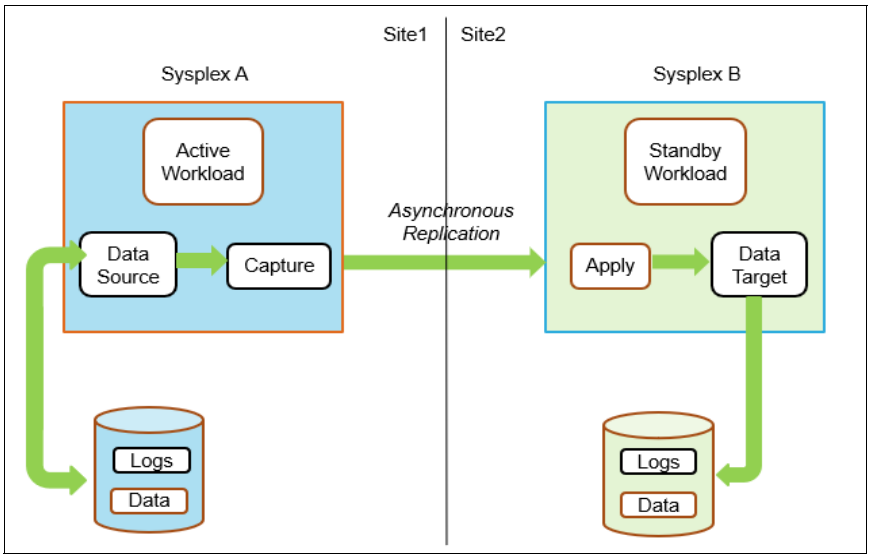


Figure 3-3 Typical GDPS CA Solution

GDPS CA ZDL solution

The GDPS Continuous Availability Zero Data Loss solution, which is known simply as ZDL, can support zero data loss for an unplanned outage in the active site.

Notice that the two sites must be located within supported Metro Mirror distances.

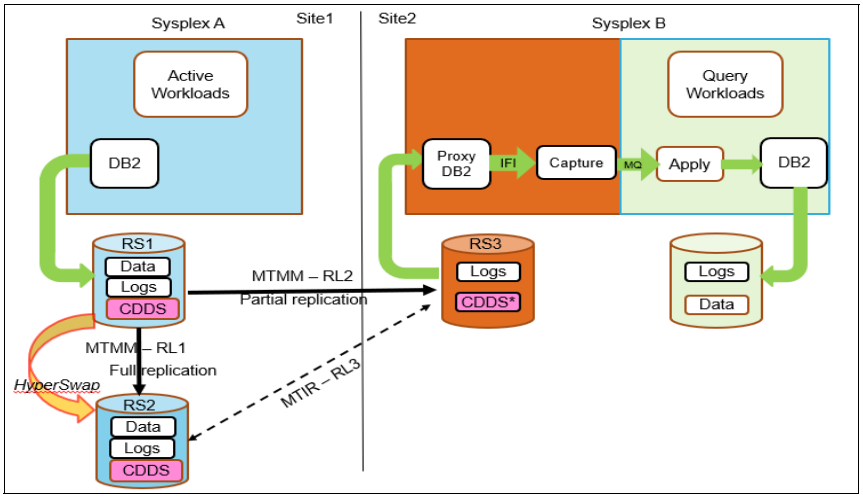


Figure 3-4 Typical Asymmetric GDPS CA ZDL system

At a high level, this solution is achieved by placing an MTMM secondary copy of the primary disk for the active workloads in the Site2. Then, you perform both the software replication capture and apply process in the Site2. Consider this scenario: the active site Site1 suffers an outage. In this case, the latest updates are available on disk in Site2. Therefore, the synchronous data can be replicated into target in the ‘normal’ or non-ZDL system.

\

Note: In this section, when we refer to the GDPS CA ZDL solution, we will use ZDL. Otherwise, we are referring to the GDPS CA solution. Currently, the ZDL solution only supports workloads that are using Db2 data.

3.4.2 Considerations

The considerations for this deployment model are:

- GDPS controller
- Software replication
- IBM Multi-site Workload Lifeline
- AA workload
- Implementation

GDPS controller considerations

Two GDPS CA Controllers are independent monoplex systems and require network access to all systems in GDPS CA environment, because communications between the systems and sites are through the network.

The GDPS Controllers must be failure-isolated from the Parallel Sysplexes that run the workload with independent network access. That way, we can still operate the GDPS CA environment if the production sysplex in that site is unavailable.

Controllers do require sufficient CPU and memory to respond in a timely fashion to situations as they occur.

Software replication considerations

A common consideration for all replication server products is that of cross-site bandwidth. Insufficient bandwidth typically leads to an increase in the latency (RPO) and might result in other undesirable conditions such as IBM MQ queues filling up.

For large scale and update-intensive Db2-based workloads, you must account for Qrep MCG (Multiple Consistency Groups) in this scenario:

- The replication work is spread across multiple capture/apply engines.
- Yet the time order (consistency) for the workload across all of the capture/apply engines is preserved in the target database.

A crypto workload uses IIDR for VSAM to replicate cryptographic keys from one sysplex to the other. The difference between this and a regular VSAM workload is as follows: No Lifeline setup is required for a crypto workload.

Furthermore, you must ensure these conditions when you switch workloads by using the ROUTING SWITCH option:

- That replication is restarted from the new standby site to the active site after the switch is completed, because the switch process leaves replication stopped in this direction.
- That the restarted point has the previous forwarding stopped timestamp.

The complete information about planning and how to set up, operate, and maintain a queue-replication environment is on the *IBM Information Management Software for z/OS Solutions Information Center*. See the following URL under the InfoSphere Data Replication topic:
http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/topic/com.ibm.swg.im.iis.prod.repl.nav.doc/topics/iiyrqcncreplepovu.html

IBM Multi-site Workload Lifeline considerations

IBM Multi-site Workload Lifeline is a critical product within GDPS CA solution that consists of Lifeline Advisors and Lifeline Agents. The primary Lifeline Advisor has these traits:

- Runs on the primary controller
- Communicates with these entities:
  - Load balancers by using Server/Application State Protocol (SASP)
  - Two or more Lifeline Agents (at least one per AA system in both Parallel Sysplexes)
  - The secondary Lifeline Advisor that is running on the backup Controller

For more information, see *IBM Multi-site Workload Lifeline V2.5 User's Guide*, SC27-4653: https://www.ibm.com/support/pages/sites/default/files/inline-files/$FILE/aqsb01O7.pdf

If Lifeline Advisor determines for a workload that all the server applications are no longer active or healthy, it generates a warning message to alert GDPS to the problem. Depending on how the workload has been defined to GDPS, a switch can automatically be performed. This switch routes this workload to the Site2 (if appropriate), or GDPS can issue an operator prompt to ask the operator how to proceed.

Active/Active workload considerations

The Active/Active (AA) solution is on the workload level, which is an aggregation of software, data, and network activity.

The only data types that are currently supported by the replication methods available are as follows: Db2, IMS, and VSAM. So, all data that is subject to updates and relating to the application must reside in the supported data types and must be replicated to the other site.

Db2 must be set up in data sharing mode, even if there is only a single Db2 member in the data sharing group.

The design of Active/Standby workload or Active/Query workload is based on the customer's requirements. If customer plans to use Site2, Active/Query workload is an option that provides the query service through the real-time replicated data.

There is no explicit support for batch updates where Site2 is unaware of the batch jobs that are running in Site1. It becomes a problem to restart batch jobs at Site2 if they are not synchronized with each other. Consider optimizing jobs so that steps are reentrant, without dependencies (file, database) between batch jobs. Knowing at what point and how to restart batch in Site2 is important. One possible example is to save the batch job checkpoint in a Db2 table, which is replicated with the rest of the Db2 data.

For Db2 workloads only, GDPS CA exploits the RREPL status to protect data object, which limits Qrep to applying data only. Otherwise, a query workload must be read-only and must not make updates to the data.

Implementation considerations

GDPS CA includes much more than data replication. It also includes many other aspects of your environment, such as sysplex, automation, network, workload routing, workload management, testing processes, and planned and unplanned outage scenario testing.

Most installations do not have all of these skills readily available. It is rare to find a team with this range of skills across many implementations. The most successful GDPS projects are those projects in which IBM and client skills form a unified team to perform the implementation. Specifically, the successful GDPS CA solution includes some or all of the following tasks:

- Planning to determine availability requirements, configuration recommendations, implementation, and testing plans.
- Emphasize the importance of data replication, which is the foundation of data consistency and routing switches. Various effective methods and best practices can be provided to improve the throughput of software replication and reduce the end-to-end latency. Performance testing and verification must be based on your environment, workload, and configuration, and continuous optimization might be required on the basis of your requirements.
- Installation, necessary customization, and performance tuning of InfoSphere Data Replication Server for z/OS (Db2, IMS, or VSAM) and IBM MQ.
- Then, installation, necessary customization, and performance tuning of LifeLine, NetView, SA, and GDPS CA.
- Planning, coding, and testing GDPS scripts and scenarios.
- Planning and implementing GDPS Continuous Availability cooperation, integration with GDPS Metro, GDPS MGM, or all of these.
- Project management and support throughout the engagement.

3.4.3 Use case: planned outage

There are many use cases where GDPS CA provides advantages for continuous availability. Benefits include a very low RTO (recovery time objective) for your critical workloads in the event of either planned or unplanned outages at one of your sites. Also see, “Use case: unplanned outage” on page 112.

Some examples of planned outages that can be avoided are as follows:

- Software migration, such as Db2 for z/OS upgrading or z/OS migration.
- Hardware upgrades, such as mainframe migration.
- Application releases, such as a large version application upgrade.
- Database schema changes and reorgs.

In these examples, you switch any affected workloads to one site while the other is being upgraded or changed. There is also the possibility for other use cases including these:

- Workload balancing between sites.
- Workload type segregation, for example, having one site that is dedicated (in normal use) to query-only, mobile, or analytics workloads.

Site1 maintenance in GDPS CA solution

Site1 maintenance is a planned outage. Corresponding planned switch processing is initiated when a ROUTING SWITCH script statement is executed, or through the Routing arrow of the Workload Management panel in WUI.

This maintenance is comprised of the following stages:

- Stage 0 - Pre-switch validation

  GDPS CA checks whether the workloads are in a switchable state. If workloads are found not to be in a switchable state, an exceptional message is issued for these workloads. Workloads detected to be switchable continue to be processed. We also need to check the end2end_latency from data replication, typically less than 1 minute.
- Stage 1 - Quiesce batch

During this stage, a WTOR is issued to allow operations to quiesce any existing batch activity before the switching continues.

- Stage 2 - Prevent new connections

  Connections to the workloads are quiesced to prevent any new work from being initiated against the workload on this site.

- Stage 3 - Deactivate existing persistent connections

  After waiting for a short time for active transactions to complete, this stage deactivates any active connections. In addition, active threads to any Db2 data are canceled. Additionally, Db2 tables are fenced to prevent any non-Qrep updates. For IMS workloads, any transactions that are not complete are dequeued and no fencing is performed for IMS data objects. During the switch, workloads are effectively fenced by preventing any programs or transactions from being initiated.

- Stage 4 - Allow replication to complete and start routing on the other site.

  Finally, any ongoing data replication is allowed to complete for a short period (a few seconds). If the replication does not complete within the timeout, a WTOR is issued to give you the opportunity to wait for longer or abort the action. After the replication has successfully stopped, the following actions are performed:

  - Db2 tables that the workload is using are unfenced.
  - IMS transactions and programs are made available for use.
  - Routing to the updated workloads is started in the newly active site.
  - Routing to any associated query workloads is reactivated.

Results:

- RPO can achieve zero.
- RTO can be the switch duration (seconds or minutes depending on the number and size of workloads).

Site2 is active for service.

Go home to Site1 in GDPS CA solution

After Site1 completes the maintenance, we must make Site1 the primary site again.

At a high level, the process for this scenario is as follows:

1. Ensure Site 1 maintenance complete and no workload or batch runs there.
2. Resynchronize Site1 data from Site2 by replication. If the accumulated data is too huge, a reverse initial load might be needed. When starting the reverse software replication, we must specify the earlier timestamp as the previous stop point.
3. When the Site1's data catches up the Site2, trigger the planned site switch from Site2 to Site1.

Results:

- RPO can achieve zero.
- RTO can be the switch duration (seconds or minutes depending on the number and size of workloads).
- Site1 is active for service.

3.4.4 Use case: unplanned outage

GDPS CA provides advantages for continuous availability in the event of an unplanned outage. Workloads benefit from a very low RTO (recovery time objective).

Site failure in GDPS CA solution

Figure 3-5 on page 112 illustrates an unplanned outage in one of two sites.

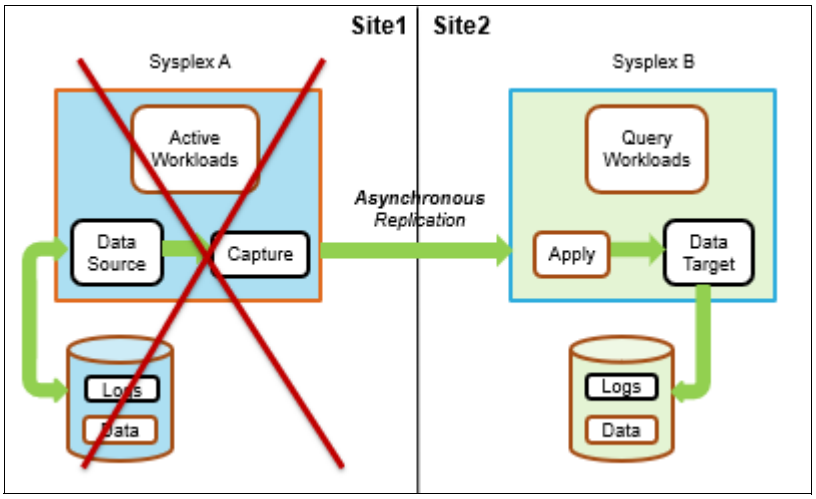


Figure 3-5 Site1 failure in GDPS CA solution

For a site failure, messages are issued for each workload that was active in that site. We must carefully examine the information that is being presented, along with alerts in SDF and other indicators. That way, we can understand why the switch prompt was issued.

After we have this understanding, and by using the site-specific recovery procedures, we can take these actions:

- Reply to the outstanding WTORs for each workload that is impacted with the appropriate response to carry out the switch.
- Recheck the conditions of the standby site, or take no further action.

The **`UNPLANSITE_tgtsysplex`** switch scripts are intended for execution when GDPS performs an unplanned site switch. When it initializes an unplanned site switch, GDPS redirects the routing of the workloads to Site2 based on the workload type.

Be aware that for an unplanned site failure scenario, the switches for update workloads make no attempt to change or restore replication between sites.

Check the workloads summary to confirm that all of them have been switched into Site2.

Results:

- RPO can be the END2END latency of software replication.
- RTO can be the switch duration (seconds or minutes depending on the number and size of workloads).
- Site2 is active for service.

Disk failure in asymmetric GDPS CA ZDL solution

Figure 3-6 on page 113 illustrates a disk failure in a GDPS ZDL solution.

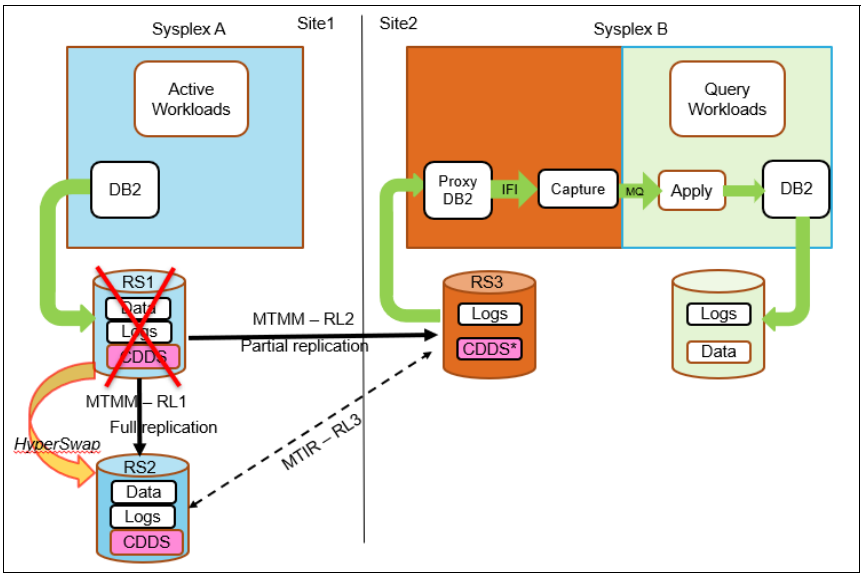


Figure 3-6 Disk failure in GDPS CA ZDL solution

In case of primary disk RS1 failure, the only viable leg for HyperSwap is RL1, which is enabled for HyperSwap. So, the HyperSwap can be initiated. After HyperSwap is completed and primary disk is RS2, the resynchronization to RS3 takes place as soon as possible.

Results:

- RPO = Zero data loss
- RTO = HyperSwap duration in second level
- Site1 is still the active site

Site failure in asymmetric GDPS CA ZDL solution

Figure 3-7 on page 113 illustrates a site failure in a GDPS ZDL solution.

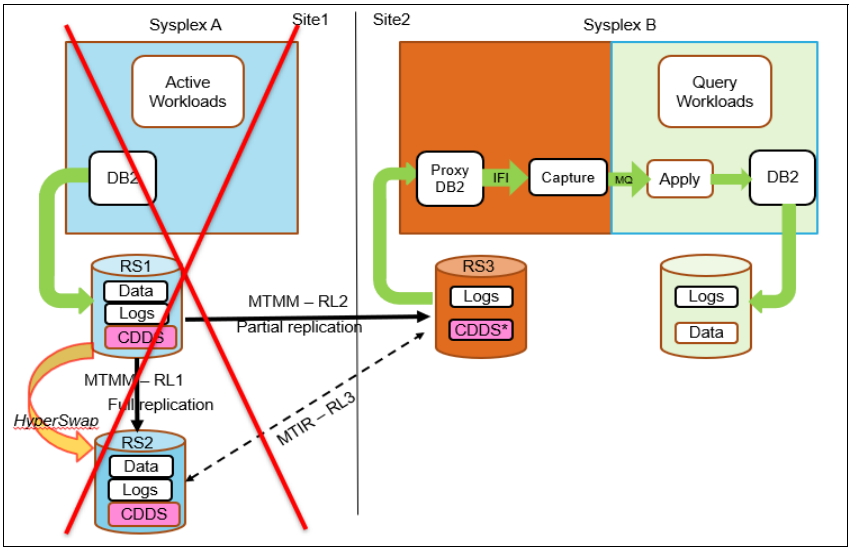


Figure 3-7 Site failure in GDPS CA ZDL solution

In case of site failure:

- Action taken: Trigger the UNPLANSITE_SYSPLEXB switch script, which switches active site into Site 2.

  Here, the **UNPLANSITE_*tgtsysplex*** script is intended for execution when GDPS performs an unplanned site switch. ***tgtsysplex*** identifies the target sysplex to which GDPS CA has switched failing workloads to.

  For example, we might have sysplexes (SYSPLEXA and SYSPLEXB). We would then code two *Switch* scripts:

  - **UNPLANSITE_SYSPLEXB**, which GDPS would initiate automatically if it performs an unplanned site switch from SYSPLEXA to SYSPLEXB.
  - **UNPLANSITE_SYSPLEXA**, which GDPS would initiate automatically if it performs an unplanned site switch from SYSPLEXB to SYSPLEXA.

Results:

- RPO can achieve zero data loss.
- RTO can achieve switch duration (seconds or minutes depending on the number of workloads).
- Activate workloads into Site2.

3.4.5 Resilience optimization and next steps

This section raises additional considerations for your test and verification efforts.

GDPS Continuous Availability is based on software replication and managing at the level of workloads. Consider including your application development services in your GDPS test environment. That way, you simulate the production workloads and define the similar application middleware subsystem. Hence, your stress tests and performance tests can reveal threats to successful deployment in Production. Also, be sure to test multiple switch scenarios, including these factors:

- Special emphasis to performance tuning and optimization of software replication.
- Maintenance policy with more sites involved.
- Manual responses, which are recommended because of the importance of switch decision.
- Disaster planning, which involves the local recovery, workload switch, and site switch. And we should also plan how to diagnose and compliment the data in extreme situations.

4

Planning for IBM Z resiliency

Building on the different layers described in the previous chapters, this chapter discusses some of the key industry best practices and considerations for resiliency. These factors will help you achieve higher levels of availability with your IBM Z infrastructure, and minimize disruption to your business.

For resiliency, every organization should have a business continuity plan (BCP). The effectiveness of the business continuity plan depends on many factors, such as these:

- Have you considered all possible failure scenarios like these?:
  loss of electrical power, damage to the site infrastructure, network outage, sabotage, cyberattack, natural disasters, and so on.
- Have you reviewed your BCP? Is it current, valid, and up to date?
- Has it been tested? How often do you test and measure the effectiveness?
- Does the BCP cover all the business requirements, RTO and RPO objectives, regulatory requirements, and the scope of recovery?
- Is everyone in the organization aware of the BCP and their role in the recovery process?
- Are the service provider contacts readily available when you must contact them?
- To what extent is your recovery plan dependent on key personnel?
- Can the recovery be performed remotely?

You can have the best resiliency technology, but you still must manage it properly. Technology is not a substitute for good planning and testing. Having a recovery plan on a piece of paper is not good enough. You must put it to the test again and again. Consider the following points to regularly assess the effectiveness of the BCP:

- Practice failover testing.
- Conduct surprise tests (without warning). In emergency conditions, people might be allowed to take with them only what is in their pockets, and no cell phones to access the system.
- Randomly have key people “not there” as in a real event. How will their absence affect the process?
- Consider automating as much of the recovery process as possible to minimize the dependency on your staff. The recovery procedure and documentation should always be available at multiple places.
- As much as possible, make DR simulations resemble a ‘real’ disaster scenario.

© Copyright IBM Corp. 2020. All rights reserved.

4.1 Considerations for the infrastructure layer

This section discusses the best practices and considerations that are related to the IBM Z platform and other peripherals that are connected to it.

4.1.1 Conduct a component failure impact analysis

A component failure impact analysis (CFIA) is a technique to raise the level of system availability by finding ways to shorten recovery times when a system component fails. It can also be used to:

- Assess a system design.
- Analyze proposed hardware and software configurations.
- Analyze the effect of placing files on various storage devices.
- Find critical paths and locate weak links in communications networks and distributed systems.
- Justify the acquisition of additional hardware and software to meet increases system availability criteria.

A CFIA looks at the flows of transactions end to end, starting with the end user, through the network, getting to the data, and back out again. Each hardware and software component should be listed. Find areas where a failure would cause productivity or venue losses, any Single Points of Failure (SPOFs), and redundancies. Then, document the owner of specific components and the action to take if there is a failure. Not all resources require a backup system. Procedural requirements depend on the level of risk and probability of failure.

Example CFIA analysis questions that you typically ask include these:

- Documentation
  - Component name / function
  - Where failover procedure is documented
  - Recovery method for data
  - Person responsible
  - When it was last tested
- Resiliency
  - Single points of failure
  - Business impact
  - Risk factors
  - Obsolescence of equipment
  - Operational considerations
- Recoverability
  - Monitoring
  - Failover automation
  - Adequate procedures
  - Trained staff
  - Defined recovery rolls
  - System support
- Does the recovery time meet SLA requirements?
- Any security risks?

The CFIA report consists of:

- A grid with system components as columns and applications as rows with each intersection showing the immediate impact of a condition or component. List everything that a transaction touches as it flows end to end, including these components:
  - System software (z/OS, JES, Middleware, monitoring and reporting tools, user exits)
  - Hardware (Server, disk, console, network routers)
  - Environmental (Power, A/C, printers)
- For each application, identify the impact of a failure, such as amount of lost revenue.
- For each intersection of an application and a component, answer this question: *What is the immediate impact on the application if this component becomes unavailable?* The answer does not consider any recovery actions. Example answers include these:
  - This application fails.
  - An alternative device is available.
  - Recovery actions are needed.
- Every recovery action is documented in a section that includes detailed information on the condition or failure, recovery action alternatives, and so on. This information might include the following points:
  - Component name / function
  - Probability and risk of component failure
  - Failover procedure
  - Where recovery plan documented
  - Recovery time
  - Recovery method for data
  - Steps required for future diagnosis of the problem
  - Person responsible
  - When was the scenario last tested?

Tip: To eliminate single points of failure, consider doing Component Failure Impact Analysis (CFIA) at regular intervals. Also, consider doing CFIA whenever you add or remove components in the infrastructure.

4.1.2 IBM Z platform

Configuring IBM Z for high availability is one of the most important tasks for platform administrators. Continuous application availability that is based in IBM Z and z/OS cannot be achieved without Parallel Sysplex. At the same time, Parallel Sysplex alone cannot provide a continuous availability environment for business applications. High availability or continuous availability depends on your design, implementation, and management of the Parallel Sysplex environment.

Note: Throughout this chapter, all references to sysplex refer to Parallel Sysplex. Sysplex and Parallel Sysplex are used interchangeably.

IBM Z offers multiple features that help to manage the platform without requiring an outage. These same features can help you handle outages, including the following key features:

- Concurrent management of I/O configuration.
- Multiple I/O drawers (domains) for redundancy.
- Concurrent processor drawer addition.
- Concurrent firmware driver upgrade.

- Licensed Internal Code (LIC) controlled number and type of processors, and memory size.
- Capacity Backup (CBU), a quick, temporary activation of central processors (CPs) in the face of a loss of customer processing capacity. Such losses might be the result of an emergency or disaster/recovery situation.
- Capacity on demand (CIU, CUoD, OOCoD, eBoD, and so on) offers the flexibility to rapidly increase or decrease your computing capability as workload or business requirements change. Such changes might represent a permanent capacity increase for planned growth or a temporary capacity increase for seasonal or unpredictable peak periods.

For more information, see *IBM Z Capacity on Demand User's Guide*: https://www-01.ibm.com/support/docview.wss?uid=isg28c04fa949ae402ed85258194006c7b9b&aid=1

In constructing an order for IBM Z, customers should consider questions like these:

- Should we enable non-disruptive upgrades to IBM Z components?
- Should we ensure that each IBM Z has enough capacity to handle the entire workload in case of an outage?
- Are we able to handle unexpected workload spikes?

To avoid unplanned outages, consider how you should configure redundancy into all the components, that support mission critical workload.

IBM recommends that you connect IBM Z to redundant input power in the data center (independent dual power feeds). All peripheral hardware devices that IBM Z connects to might also be connected to redundant input power. Or they might have UPS backup systems in the data center.

If your organization uses a single-system configuration, consider implementing sysplex with at least two LPARs. Also, consider deploying business applications on multiple LPARs in the sysplex to avoid outage due to LPAR failures or due to planned outages that do not affect the entire sysplex.

To achieve the highest level of availability, consider implementing Parallel Sysplex configuration with more than one IBM Z platform. When you create a Parallel Sysplex environment, consider setting up at least one z/OS LPAR on each IBM Z platform. (And the Z platform should be enabled with Server Time Protocol.) This approach for the overall environment ensures that the applications do not suffer an outage when one of the IBM Z platforms is unavailable due to planned maintenance, upgrade, or outage.

Tip: Consider configuring two z/OS images on each CPC in a sysplex. This might give you the ability to continue to utilize all the available processors on the CPC even if one of the images has an outage (planned or otherwise).

The two LPARs do not have to be located in the same Parallel Sysplex. And each LPAR can play distinct roles: Dev, Test, another Sysplex, balancing CBU with system management, and so on.

4.1.3 Coupling facility

A coupling facility is a special logical partition that provides locking, caching, and list services between coupling-capable z/OS images. Coupling-facility links connect a coupling facility to

the coupling-capable processors. The coupling-facility control code (CFCC) provides the coupling-facility functions and runs in a CF LPAR on the IBM Z platform.

Even for single system configurations, having two CFs can improve availability. Some types of CF failures can affect an entire CF image. By having a second CF, you make it possible for the structures to be rebuilt while z/OS availability is maintained. Note that this configuration is still exposed to single points of failure as far as the IBM Z platform is concerned.

Some CF structures, for example Db2 IRLM lock structures, require failure isolation. Without adequate failure isolation, loss of such CF structures might require a group-wide subsystem restart (for example, group-wide restart of Db2 subsystems) for recovery. This might lead to an outage that you would certainly like to avoid. In a multi-system configuration, you can avoid placing CF structures on the same system where the subsystem (or application) is executing. Consider placing such CF structures on a stand-alone CF or alternatively duplex the CF structure so that a copy of the structure is always available on another CF (on a different CPC). The subsystem(s) that are still executing can continue to do so by using the structure copy that is unaffected by the failure.

Tip: Traditional synchronous system-managed duplexing of coupling facility lock structures can affect performance. Asynchronous CF Lock Duplexing is a practical alternative that you should consider. See “Asynchronous Duplexed CF Lock Structures” on page 126.

With a stand-alone coupling facility in a sysplex, you might choose to *not* implement CF lock structure duplexing. Instead, you can implement asynchronous CF lock structure duplexing (discussed later in this section). And asynchronous configuration also improves availability, because it is resilient to IBM Z platform outage (planned or unplanned).

IBM recommends that you connect each LPAR in the sysplex to all the coupling facilities by using at least two paths. Also, consider connecting each CF to other CFs in a sysplex, over CF-CF links. For a production sysplex, consider having dedicated physical units (PUs) characterized as CFs.

For mission-critical applications that cannot withstand an outage, consider having a third coupling facility. This provides a fallback for the sole coupling facility that remains while one of the coupling facilities is down for maintenance (or outage). In this way, you comply with the general goal that structures that are defined in the CFRM policy should not have a single point of failure.

Tip: Consider configuring CF structures so that they do not have a single point of failure.

Consider the proper sizing of CF LPARs. Ensure that they have sufficient CPU and storage for these tasks:

- Rebuild CF structures in the event of a failure or maintenance downtime.
- Handle additional workload.

For a detailed study of various CF configuration options, refer to the *CF Configuration Options* white paper at: https://www.ibm.com/downloads/cas/JZB2E38Q

Also be aware of the IBM Z Coupling Facility Structure Sizer Tool (CFSizer). CFSizer is a web-based application that returns structure sizes based on the latest CFLEVEL for the IBM products that exploit the coupling facility. For more information, see: https://www-01.ibm.com/support/docview.wss?uid=isg3T1027062

4.1.4 Connecting IBM Z to storage servers

When you connect IBM Z to storage servers, consider the following issues:

- Connect FICON features (in the IBM Z I/O drawer) to the storage server ports so that redundant paths pass through at least two different Fibre Channel (FC) switches. This configuration helps you to manage FC switch outage (planned or unplanned).
- Connect each logical control unit (LCU) defined in the I/O configuration to at least two different channel paths. Choose the FICON feature corresponding to these channel paths from different domains (I/O drawer) to provide redundancy, and to avoid single point of failure.
- Configure the fibers from the FC switch to terminate on redundant ports on the storage server.

For more information on storage server high availability best practices and considerations, see *IBM DS8880 Architecture and Implementation*:
http://www.redbooks.ibm.com/abstracts/sg248323.html?Open

4.1.5 Connecting IBM Z to the network

When you connect IBM Z to the enterprise network, consider the following points:

- Each network subnet might be configured to terminate on at least two different Open Systems Adapter-Express (OSA-Express) features in a different domain (I/O drawer).
- Each LPAR might be connected to at least two different network subnets for redundancy.
- For single system configuration, consider implementing virtual IP addresses (VIPA). VIPA eliminates the dependency of TCP/IP stack (and the application that uses this stack) on multiple physical network interfaces. As a result, network interface failure does not affect the application availability. Using either ARP takeover or a dynamic routing protocol (such as OSPF implemented by OMPROUTE), static VIPAs can enable mainframe application communications to continue without being affected by network interface failures. As long as a single network interface is operational on a host, communication with applications on the host persists. Dynamic routing protocols enable traffic routing through alternate links in case of a link or node failure.
- For better availability in a Parallel Sysplex environment, consider implementing dynamic VIPA (DVIPA). DVIPA can be defined on multiple stacks and moved from one TCP/IP stack in the sysplex to another automatically. One stack is defined as the primary or owning stack, and the others are defined as backup stacks. Only the primary stack is made known to the IP network.
- To achieve the highest level of application availability and to distribute IP workload across systems in a sysplex, consider implementing Distributed DVIPA, Sysplex Distributor.

For more information, see *IBM z/OS V2R2 Communications Server TCP/IP Implementation: Volume 3 High Availability, Scalability, and Performance*:

http://www.redbooks.ibm.com/abstracts/sg248362.html?Open

4.1.6 Cryptography features/RoCE features/zEDC features

If you are using optional features like cryptography, zEDC, or RoCE, IBM recommends that you have at least two features of each type active and operational on every IBM Z platform. You must also ensure that every LPAR that uses these features is configured to access at least two such features, to eliminate single point of failure.

4.2 Considerations for the operating system layer

The three widely used operating systems on the IBM Z platform are z/OS, z/VM, and the Linux distributions for Z. These operating systems provide several features and functionality that products and applications can leverage to ensure high availability and system stability. This section discusses those features and functionalities.

4.2.1 General considerations for Parallel Sysplex environments

In a Parallel Sysplex environment it is possible to avoid single points of failure by exploiting parallel processing. Consider configuring your Parallel Sysplex environment in such a way that all the participating members have concurrent access to critical applications and data. Such a configuration not only enables you to share workload across sysplex members, but also make the systems resilient from possible hardware or software failure. Peer instances of a failing subsystem can detect failures and take appropriate recovery actions for the resources held by the failing subsystem instance.

Consider these approaches as you design and manage a Parallel Sysplex environment:

- Clone the z/OS LPARs and application processing subsystems so that other components instances can process the work/transactions that a failed instance is currently unable to process.
- To maximize redundancy, consider configuring two or more z/OS LPARs connected to two or more coupling facilities (CFs) over redundant coupling links between each LPAR and CF.
- Consider spreading z/OS images from all the sysplexes in your enterprise across all the systems, except for the systems that host pseudo-standalone CFs.
- Consider configuring Virtual IP Addresses (also Dynamic VIPA and Distributed Dynamic VIPA) so that the applications hosted on different LPARs can be accessed through a single IP address from outside of the sysplex environment.
- Consider duplexing of CF structures in your sysplex environment to make the structures resilient to CF outage (planned or unplanned).
- Dynamic transaction routing/workload balancing and elimination of static affinities to particular regions.
- Enable data sharing (IMS or Db2 data sharing, VSAM RLS data sharing, zFS file system sharing, and so on). That way, the work/transactions can access to the data that they require from anywhere in the sysplex.
- Use aggressive sysplex automation and alerting for automated recovery / restart.
- Conduct performance and stress testing with production-like work/transaction volumes.
- Set up a Dev/test sysplex that is separate from production sysplexes. That way, you can test sysplex-related changes without exposing production workloads to problems, such as from rogue transactions.
- Manage workload affinities effectively so that any work can run on any of the multiple supporting subsystem instances. Affinities are static bindings where work can run in one system or in one subsystem instance only. Certain types of workload affinities can affect even the most resilient sysplex infrastructure, if the subsystem instance that hosts the affinity is down. Yet it is often possible to work around affinities without time-consuming application changes, for example, by using the CICS/IA tool with CP/SM. See IBM CICS Interdependency Analyzer for z/OS:
  https://www.ibm.com/us-en/marketplace/cics-interdependency-analyzer

- Have a well-defined maintenance strategy to stay up to date and avoid defect rediscoveries. Take action on health check warnings and exceptions. Consider taking advantage of the IBM RSU (Recommended Service Upgrade) process. See the *Understanding the RSU and PUT process* web page for more information: https://www-01.ibm.com/support/docview.wss?uid=swg21618290&mhsrc=ibmsearch_a&mhq=rsu%20
- Availability management:
  - Ensure that operational procedures are in place and understood by the operations team. That way, they can take quick action when critical sysplex messages are raised to operations.
  - Modify recovery operations procedures to attempt recovery at the most granular level possible, as in these examples: attempt hot/warm starts, subsystem recycles, “group restarts,” and other granular actions before you resort to system IPL (or multi-system/sysplex-wide IPL). Minimize the scope of the impact of a problem when it occurs. And the more that you automate this type of recovery, the better.
- Consider implementing these architectural features:
  - Sysplex dynamic workload routing/balancing. That way, any work can be routed to any active region that can process it. (For example, Sysplex Distributor, IBM MQ shared queues, IMS shared queues, and so on might be used.)
  - Db2 Group Buffer Pool user-managed duplexing for Db2 Group Buffer Pool cache structure recoverability.
  - BCPii and System Status Detection (SSD) protocol for identifying systems that are in a known, remotely recognizable “down” state as quickly as possible, through hardware APIs.
  - Pseudo-standalone CF images (no co-resident z/OS images from the same sysplex). In this way, you provide failure-isolation without the use of system-managed CF duplexing.

For Parallel Sysplex best practices refer to *System z Parallel Sysplex Best Practices*, SG24-7817 at: http://www.redbooks.ibm.com/abstracts/sg247817.html?Open

4.2.2 Sysplex Failure Management (SFM) Policy

You must give careful consideration to effective management of sysplex failures. Based on the SFM policy, z/OS can take swift actions, without any operator intervention. Actions include avoidance of sympathy sickness in a sysplex. This problem requires removal from the sysplex of a ‘sick, but not dead’ application instance or a middleware instance or a system out of the sysplex.

SFM acts against sympathy sickness, which typically starts when one sysplex member is unable to fully participate in shared sysplex activities. Examples include the following scenarios:

- Not releasing enqueues and locks within the stipulated time on resources that have sysplex scope.
- Not sending or receiving XCF signals.
- Not updating heartbeat in the sysplex couple data sets in a timely manner.

SFM also implements fencing. Fencing isolates a system so that it cannot access shared data. Thus, SFM makes it safe for the survivors to release serialization of the shared resources.

In summary, an SFM policy can isolate or remove a 'sick' system from sysplex in a timely manner, and can fence the survivors. In this sense, such a policy can be considered to prevent sympathy sickness in the other LPARs.

Note: Consider automatically responding to critical messages related to XCF signaling through use of an automation product. XCF messages are important for availability. And automation can be used at least to highlight these messages and alert operations staff to quickly take action.

Consider using the z/OS AUTOIPL feature to start a stand-alone dump (SADMP) or an IPL of a z/OS image. Sysplex failure management (SFM) might take some time to perform fencing isolation on the failed system. AutoIPL might get delayed for several minutes before it actually initiates the SADMP or the re-IPL. During this time, the failed system appears to be hung. (Be aware that AutoIPL is not appropriate for a GDPS environment.)

For more information on how sysplex failure management policy can help you achieve high availability, see these resources:

- z/OS Parallel Sysplex Configuration Overview (SG246485): http://www.redbooks.ibm.com/redbooks/pdfs/sg246485.pdf
- Improve your Availability With Sysplex Failure Management white paper: https://www.ibm.com/downloads/cas/O2RKMVOY

4.2.3 Automatic Restart Manager (ARM) Policy

You cannot maintain High Availability for shared data in failure scenarios without reliable, rapid restart for middleware components. The z/OS Automatic Restart Manager (ARM) is a z/OS recovery function that can improve the availability of applications and middleware products. When a started task or authorized batch job fails, or the system on which it is running fails, ARM can restart the process without operator intervention. Depending on the ARM policy, a failed task or a group of tasks can be automatically restarted either on the same system or on a different system in the sysplex, in a predefined order. When a task is restarted on a different z/OS image, it is referred to as a cross-system restart. You might also be able to restart the applications or products by using automation products such as IBM System Automation for z/OS.

Note: Many automation products can automate the in-place restart of a product if it fails. However, the power of ARM is that it also automates the database manager restart when an LPAR failure occurs. This capability ensures that in-flight units of work are backed out as soon as possible, releasing the retained locks.

Consider setting up an ARM policy for every sysplex to maximize the availability of the applications and tasks that get registered with the ARM feature. For example, consider implementing automated same-system and cross-system restart procedures for Db2/IRLM and other critical subsystems. You can use either ARM or IBM System Automation for z/OS. Sysplex recovery for some data sharing subsystems is restart-based, with sysplex-wide retained locks not released until the subsystem restarts. This approach often causes sympathy sickness until resolved.

For more information on how sysplex failure management policy can help you achieve high availability, see *z/OS Parallel Sysplex Configuration Overview*, SG24-6485: http://www.redbooks.ibm.com/redbooks/pdfs/sg246485.pdf

4.2.4 IBM Health Checker for z/OS

IBM Health Checker for z/OS is a base component of z/OS. It is used to identify potential problems before they affect the system. IBM Health Checker for z/OS checks current, active z/OS and sysplex settings and compares them with those suggested by IBM or defined by you. Customers can also write their own checks with IBM Health Checker for z/OS. As a result, they extend the reach of the checker to environment-specific settings.

If IBM Health Checker for z/OS finds a deviation for a health check, it generates an exception message. IBM Health Checker for z/OS is a prerequisite for Predictive Failure Analysis (discussed later in this section).

Consider implementing IBM Health Checker for z/OS on all LPARs. For more information, see *Exploiting the IBM Health Checker for z/OS Infrastructure*:
http://www.redbooks.ibm.com/abstracts/redp4590.html?Open

4.2.5 Placement of data sets

Consider separating system-critical primary and alternate data sets onto different volumes, logical control units, and channel paths. Examples of such data sets include couple data sets, CFRM data sets, JES checkpoint data sets, SMS control data sets, and so on.

Consider implementing two types of separation:

- Placement on different DASD volumes.
- Use of different channel paths to connect IBM Z to the storage servers.

Similarly, critical data sets for your business applications can be stored on different DASD volumes to ensure high availability.

4.2.6 JES2 emergency subsystem

How many situations have you seen in which users and system programmers are unable to log on to TSO because the spool is full? These situations are difficult to handle because runaway jobs or started tasks can fill up the spool data sets within a matter of seconds. The problem persists, even if you keep adding more and more spool volumes. The system neither processes existing work nor accepts new work. Quite often, this situation is interpreted as an ‘outage’. Recovery from JES2 resource shortage can take considerable time when no one is able to log on to TSO to access the system management environment.

To address this situation, z/OS 2.3 introduced a new feature called JES2 emergency subsystem. This emergency subsystem ensures that the system management environment is always accessible — even at times of JES2 resource shortage for jobs — to TSO users and stated tasks that are identified as privileged.

Emergency JES2 subsystem goal is to maintain the normal systems management environment available when there is a JES2 resource shortage. That was created after a considerable number of cases where the exhaustion of certain critical resources (such as spool space) might prevent such activities as TSO LOGONs, execution of diagnostic jobs, and JES2 commands. The solution is based on having an amount of critical resources, such as percentage of SPOOL, jobs, output elements, and BERTs reserved for privileged job (STC, TSU, JOB) consumption. This assures enough resource to log on, perform analysis, submit jobs, and resolve the root cause of resource exhaustion. It is not meant for running high-priority workloads.

Consider setting up a JES2 emergency subsystem and configure the ESM product (such as IBM RACF) to grant privileges to the TSO user IDs of z/OS system programmers. With these privileges, the TSO users can access the emergency subsystem. The next time that you must deal with a JES2 resource shortage such spool full, the system programmers can log on to TSO and fix the problem quickly.

For more information, see these resources:

- JES2 Emergency Subsystem IBM Knowledge Center: https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.hasa300/priv_emer_subsys.htm
- Privilege support and the emergency subsystem IBM Knowledge Center: https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.hasa300/priv_emer.htm

4.2.7 Predictive Failure Analysis

Predictive Failure Analysis (PFA) is a base component of z/OS. It is designed to predict potential problems with your systems. PFA goes beyond failure detection to predict problems before they occur, thus taking system availability to the next level. PFA uses the IBM Health Checker for z/OS to collect data about the systems. Using this data, PFA creates a model of the future behavior of the z/OS systems and compares the actual behavior with the future behavior. If the behavior is abnormal, PFA generates a health check exception.

Consider implementing predictive failure analysis feature on all z/OS systems. This early warning system helps organizations achieve higher levels of system availability.

IzODA is an assembly of open source and proprietary technologies that allow data scientists and application developers to analyze and visualize the large volumes of data hosted on IBM Z.

The solution is designed to simplify data analysis. It combines open source run times and libraries with analysis of z/OS data at its source, to reduce data movement and increase the value of insights gained from leveraging current data.

IzODA contains two of the most popular code stacks used by the data science community - Apache Spark and Anaconda, to analyze modern enterprise-level volumes of data. The Optimized Data Layer provides unique data abstraction capabilities that allow data scientists to reach key data sources on and off the mainframe through a common interface.

One of the key features of z/OS IzODA Spark (built on Apache Spark), is to perform in-memory computing. Spark allows caching of intermediate results in memory rather than following the traditional approach of writing it to the disk. This drastically improves the performance of iterative processing.

z/OS IzODA Anaconda includes Python and Anaconda Python packages for data science, which provide data scientists with a comprehensive solution for integrating computations to the data.

The third component of IzODA, z/OS IzODA Mainframe Data Service (Data Service or MDS) provides integration facilities for both IBM Z data sources and other off-platform data sources. The Data Service provides your Apache Spark application with optimized, virtualized, and parallelized access to a wide variety of data.

z/OS IzODA Livy (built on Apache Livy) is a REST service used in conjunction with Spark enables users to submit Spark jobs without having the Spark client installed.

For further details and IzODA reference architecture refer to: https://izoda.github.io/

For more information, see the *z/OS Predictive Failure Analysis* white paper: https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.hasa300/priv_emer.htm

4.2.8 Runtime Diagnostics

Runtime Diagnostics is a base component of z/OS. It analyses a 'sick but not dead' system and helps the system programmer analyze a potential problem. These soft problems are often difficult to detect. Over time, multiple problems can emerge, including performance degradation. Runtime Diagnostics looks for evidence for these types of soft failures while it analyzes the system. The analysis that is performed by Runtime Diagnostics is quite comprehensive and is similar to analysis done by an experienced system programmer. Typically, Runtime Diagnostics completes the analysis within 60 seconds and then suggests the next steps.

Note: Runtime Diagnostics analyzes the operating system only.

Consider setting up RTS on all z/OS systems. That way, you can start it whenever you want to analyze that are related to the operating system.

For more information, see *Runtime Diagnostics* in IBM Knowledge Center: https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.e0zk100/rtdin.htm

4.2.9 Asynchronous Duplexed CF Lock Structures

In a duplexed CF structure, you maintain two copies of a structure in separate coupling facilities. If the primary CF structure is lost, the secondary structure can process the workload without any disruption. Despite the benefits of using CF structure duplexing, you incur significant performance overhead with synchronous system-managed duplexing of lock structures. This concern is especially relevant in environments with heavy data-sharing workload. The larger the distance that separates the CF structures (primary and secondary structures), the higher the performance overhead.

Asynchronous CF Lock Duplexing is an alternative to synchronous system-managed duplexing of coupling facility lock structures. With this feature, you consider the command to be completed as soon as the primary structure is updated. Updates to the secondary structure happen asynchronously from the primary structure updates. A 'sync-up' protocol must ensure that the secondary structure has all the necessary updates from the primary structure before any transaction is committed.

Asynchronous duplexed CF lock structures give you a practical alternative to system-managed duplexing of the lock structure. In a heavy data-sharing workload environment, this alternative avoids significant impact to the system performance.

For more information, see *Asynchronous CF Lock Duplexing with SAP Performance Report from IBM zSystems*, at: https://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102720

The concept of duplexing Asynchronous CF lock structures applies to any application that runs on IBM Z.

4.2.10 z/VM Single System Image (SSI) and Live Guest Relocation (LGR)

A z/VM SSI cluster consists of up to four z/VM systems (referred as z/VM SSI members) in an Inter-System Facility for Communications (ISFC) collection. The cluster is self-managed by CP, which uses ISFC messages that flow across channel-to-channel connections between the members. All members can access shared resources, including DASD volumes, Ethernet LAN segments, and storage area networks (SANs). System management becomes easier, because the members can be serviced, managed, and administered as if they are one integrated system.

z/VM SSI clustering helps to meet horizontal growth of z/VM workload. With SSI live guest relocation (LGR), Linux virtual servers (guest virtual machines) can be relocated to another member in the cluster.

That way, organizations can roll out hardware and software maintenance, and upgrades without causing disruption to the business. This capability makes a significant contribution to continuous availability.

To achieve higher levels of system, application, and data availability, organizations can consider transitioning from stand-alone z/VM systems to z/VM SSI configuration.

You can also consider implementing Linux Health Checker to identify issues before they cause problems. It helps to detect configuration errors, deviations from best practices, single point-of-failures, unused accelerator hardware, hardware that is running in degraded mode, and so on.

For more information, see *An Introduction to z/VM Single System Image (SSI) and Live Guest Relocation (LGR)*:
http://www.redbooks.ibm.com/abstracts/sg248006.html?Open

4.3 Considerations for the middleware layer

In this section, we discuss commonly used IBM middleware program products and considerations for high availability.

4.3.1 IBM MQ for z/OS

Availability of a queue is critical for availability of an application. To make IBM MQ queues (and applications that use these queues) highly available, consider configuring IBM MQ to exploit high availability features of a Parallel Sysplex environment.

Restarting a queue manager

In “IBM MQ for z/OS” on page 39 we discussed the importance of backing up queue manager resources like log data sets, bootstrap data sets, page sets, and so on for a reliable recovery after a failure has occurred. The recovery procedures should be tested at regular intervals to validate the procedure and to measure its effectiveness. At the minimum, the queue managers should be restartable on a different z/OS LPAR in the event of an LPAR failure. To achieve this, eliminate any affinity between the queue manager and the LPAR on which it runs. The restart process for the queue manager is disruptive. To minimize the disruption, use an automation product to restart the queue manager on a different LPAR. The following automation products are valid choices: IBM System Automation for z/OS or Automatic Restart Manager (ARM) policy.

IBM MQ clustering

As described in “IBM MQ clusters” on page 41, clusters can provide resilience when multiple queue managers in the cluster contain a cluster queue with the same name. The MQ cluster workload management routine will send messages to each instance of the cluster queue regardless of whether there is backend service instance processing the messages on the queue. If there is no backend service instance processing the queue the messages on it will build up, delaying work. You should put an automated process in place to ensure that backend service instances are always running. Or use the AMQSCLM sample (https://www.ibm.com/support/knowledgecenter/en/SSFKSJ_9.1.0/com.ibm.mq.dev.doc/q024620_.htm) that is provided with MQ to redistribute the messages to other instances of the queue.

IBM MQ shared queues

“IBM MQ queue sharing groups” on page 41 describes how queue sharing groups provides high availability of messages. When you use a queue sharing group one key requirement is to back up the MQ coupling facility structures periodically, at least once per hour. You can run the **CFSTRUCT BACKUP** command from any queue manager in the group. This task takes a snapshot of the persistent messages on the structure and creates a point of consistency in the unlikely event that the structure must be recreated from the MQ logs after a failure.

Tip: Consider using the MQ command **BACKUP CFSTRUCT** at least once an hour, to minimize the time it takes to restore a CF structure.

For more information, see *High Availability in WebSphere Messaging Solutions*: http://www.redbooks.ibm.com/abstracts/sg247839.html?Open

4.3.2 IBM Db2 for z/OS

Db2 provides extensive methods for recovering data after errors, failures, or even disasters. You can recover data to its current state or to an earlier state. The units of data that can be recovered are table spaces, indexes, index spaces, partitions, and data sets. You can also use recovery functions to back up an entire Db2 subsystem or data sharing group.

Development of backup and recovery procedures is critical in preventing costly and time-consuming data losses. In general, ensure that the following procedures are in place:

- Create a point of consistency.
- Restore system and data objects to a point of consistency.
- Back up and recover the Db2 catalog and your data.
- Recover from out-of-space conditions.
- Recover from a hardware or power failure.
- Recover from a z/OS component failure.

Note: Application owners sometimes overlook the importance of a carefully defined point of consistency. Application developers might not be able to answer the following planning questions, whereas application owners can:
Which tables are used? How often to back them up? Where to back them up (into which medium)? Have tables been missed? Are multiple versions of tables being backed up and then improperly restored multiple times because they are referenced by multiple applications?

In addition, for a single site IBM Z configuration, consider setting up a procedure for recovery at a remote site in case of disaster or a CPC failure. Documented disaster recovery methods

are crucial in the case of disasters that might cause a complete shutdown of your local Db2 subsystem. Also, consider registering Db2 to the Automatic Restart Manager (or an automation product like IBM System Automation). This facility automatically restarts Db2 (on the same LPAR or a different LPAR depending on the situation and configuration) if it goes down as a result of a failure.

The long tradition of IBM Db2 for z/OS has established best practices and tips for you to consider in the subsequent sections.

Db2 data sharing

In a Parallel Sysplex environment, a group of Db2 subsystems that share the same Db2 for z/OS data concurrently is called a *data sharing group*. Every Db2 subsystem that participates in a data sharing group is referred to as a member of the data sharing group. Because the Db2 subsystems typically are located on different LPARs, the data that they share must reside on shared disks.

There are various reasons to implement Db2 data sharing and the stable data integrity that it brings:

- **Availability:** Data sharing group increases data availability, thus helping organizations to meet their service objectives during both planned and unplanned outages.
- **Capacity:** With more and more data being stored in Db2, the processing capacity that is required to process the data can exceed the capacity of a single system. Data sharing allows organization to meet their data processing capacity needs without having to split databases across systems.
- **Growth:** Data sharing group helps organizations to overcome scalability and data growth-related challenges, thus making the organization future ready.
- **Workload balancing:** Data sharing groups balance the workload on the Db2 subsystems and they are extremely effective in managing unpredictable spikes in the workload. They also help to achieve very high transaction rates.
- **System management:** Consolidation of data makes system management very easy.

Db2 data sharing is by far one of the most critical considerations for ensuring data and application availability in any enterprise.

There are also many methods and rules of thumb that can be implemented to increase the concurrency of the shared data in a data sharing group as well as optimizing for performance and reduced CPU, including these examples:

- Applications should be coded to be well-behaved threads/applications that frequently commit. Having a low commit frequency will enhance concurrency of threads that run on a member or multiple members of a Db2 data sharing group. In addition to optimizing concurrency a low commit frequency will promote faster Db2 restarts/recoveries in case of a subsystem failure.
- Db2 BIND parameters such as RELEASE(DEALLOCATE) CURRENT DATA also play a role in Db2 availability and performance. In addition, there are various subsystem and buffer pool options that enhance Db2 availability such as CHECKPOINT frequency, local buffer pool settings, Group buffer pool settings to name a few.

For more information, see *Db2 for z/OS: Data Sharing in a Nutshell*:
http://www.redbooks.ibm.com/abstracts/sg247322.html?Open

Deletion of old coupling facility structures after a DR test

Consider deleting Db2 coupling facility structures from a DR test that is complete.

System programmers sometimes forget to delete such Db2 coupling facility structures after a DR test. In that case, if a real DR event happens, the stale Db2 structures might cause logical data corruption.

Warning: Deletion of structures from CFRM policy should be done carefully. CFRM policy and the structures that are defined in the policy should be consistent with the 'reality' in the system.

Consider purging data from existing CF structures before performing Db2 restart at the DR site to avoid logical data corruption. You might also decide to automate these actions to avoid manual intervention.

You can use the following command to list the Db2 structures that belong to a particular Db2 group:
`D XCF,STR,STRNAME=db2grp*` *(db2grp* is the group name)

You can force the LOCK1 and SCA structures by using the following command:
`SETXCF FORCE,STR,STRNAME=strname` *(strname* is the name of the structure)

For failed persistent group buffer pools you can use the following command to force the connections:
`SETXCF FORCE,CONNECTION,STRNAME=strname,CONNAME=ALL` (strname is the name of the structure)

After the connections are forced, the group buffer pool will be deallocated.

Note that XCF commands are the recommended way to make the changes suggested above. Use of CF commands will bypass CFRM, and it might lead to issues related to cross-system coupling (XCF) service.

Hold on to Db2 log-based recovery procedures

Do not discard standard Db2 log-based recovery procedures like taking image copies for forward log recovery to recover data to a previous point in time (or to the point of failure). Although very rare, a DR test (or even worse, a real DR event) might detect inconsistency in the data, failed utilities, or issues related to coupling facility and so on. In these circumstances, you must be able to recover the Db2 objects to a point of time that you choose.

A word about the Db2 restart light

Consider using the Db2 restart light mode if you are restarting a Db2 subsystem on an alternate LPAR. Typically, such restarts are required by an LPAR failure or CPC failure. With Db2 restart light, the system comes up quickly, backs out in-flight units of work, and releases retained locks.

The concept behind the restart light is to bring up Db2 with a very small footprint during a cross-system restart of a failed Db2 member on an alternate LPAR. By using a small-footprint instance of Db2, you avoid disturbing the existing Db2 member that is running on the alternate LPAR.

In environments where there are two-phase commit activities with Db2, RESTART LIGHT(NOINDOUBTS) should be used so that Db2 can release the RETAINED LOCKS and terminate faster. This will have less impact on other surviving Db2 members in the data sharing group.

For more information, see the *Restart light* topic in IBM Knowledge Center for IBM Db2 for z/OS:
https://www.ibm.com/support/knowledgecenter/en/SSEPEK_11.0.0/dshare/src/tpc/db2z_restartlightds.html

4.4 Considerations for the applications layer

In this section, we discuss considerations for the IBM CICS and IBM IMS applications that allow you to manage the myriad of transactions that a modern business requires. An additional consideration in this section is the use of IBM WebSphere Application Server, which provides a Java API if required for your programming efforts in system management.

4.4.1 CICS Transaction Server

Continued availability of CICS region is critical for the many applications deployed within CICS, and the data that is accessed by those applications. If CICS regions become unresponsive or fail, entire suites of applications can be compromised leading to significant service outages.

CICS is designed to be highly resilient and capable of continuing to service application requests under very high transaction volumes. However, to achieve this level of resilience requires planning and monitoring to ensure that good practices are followed and any issues are isolated and rectified quickly.

In general, you should implement the following practices to ensure that CICS regions maintain the highest levels of availability:

- **Efficient application design**:
  - Analyze applications that can benefit from being updated to be threadsafe. When you make an application program threadsafe, you can use the CICS open transaction environment, potentially improving performance by avoiding TCB switches. In addition, making applications threadsafe can improve the scalability of a single CICS region. Scalability increases because you reduce the requirement to use the QR-TCB, and therefore extend the capacity of a single CICS region beyond the capability of a single CPU. For further details, see *Threadsafe Considerations for CICS*, SG24-6351, at: http://www.redbooks.ibm.com/abstracts/sg246351.html?Open
  - Design applications to be as short running as possible, and to store state in temporary data area between user invocations. Traditionally this style of application was termed pseudo-conversational.
  - Design applications to take frequent sync points to reduce the time that database records remain locked.

    Note: CICS applications often run in a "pseudo-conversation" mode. A transaction starts, then generates state information that is needed by a later transaction, and then ends. The way the subsequent transaction gets this state information can cause affinities by constraining where the second transaction run. This can place constraints on dynamic routing, which is discussed further in Clustering and dynamic routing.

  - Take advantage of CICS facilities to protect critical storage areas, via storage protection and transaction isolation. For further details refer to , “Storage protection” on page 48.

Tip: For good resilience, ensure that storage protection is enabled for data and executable code:

Enable transaction isolation wherever possible. As an alternative, consider using transaction isolation to improve the effectiveness of application testing.

If you have existing CICS application code that is not reentrant, consider changing application code to make it reentrant.

- Design or migrate application interfaces to use structured containers within a channel interface, rather a single COMMAREA. This promotes better application reuse and more efficient processing for payloads larger than the 32 KB COMMAREA limit.

- **Clustering and dynamic routing**:
  - Consider configuring clusters of specialized regions in a multi-region CICSplex, routing work between themselves by using CICSPlex SM dynamic routing. This configuration provides both for both redundancy and scalability.
  - In order to optimize the CICSPlex SM routing process, it is highly recommended to enable CICSPlex SM sysplex optimized WLM. This was originally made available with CICS TS V4.1 and takes advantage of the Coupling Facility to store near real-time status of the target regions. The stored region status is then used to make more efficient routing decisions
  - Isolate and reduce transaction affinities to enable efficient routing between CICS regions. Consider using CICS Interdependency Analyzer (IBM IA) to identify and manage affinities. There are several types of affinities:
    - Inter-transaction affinity:

      1.A transaction terminates, leaving state data for a subsequent transaction.

      2.A transaction creates data for a second transaction that is running at the same time.

      3.Two transactions synchronize which each other by using mechanism such as an ECB or ENQ.

    - Transaction-system affinity:

      1.Dependency on a global user exit program.

      2.Use of INQUIRE and SET commands.

      3.Dynamic use of memory storage areas.

    - Affinities are characterized by these traits:

      1.Relationships - Instances of all transactions must execute on the same CICS region, z/OS, or Parallel Sysplex.

      2.Duration - The affinity lasts, for example, the lifetime of the unit of work, the process, the transaction, the user ID signs off, the region terminates, and so on.
  - To achieve the highest level of availability for a CICSplex, consider configuring CICS regions on LPARs that are located on different physical CECs. Workload distribution and workload balancing in a GDPS Continuous Availability environment is achieved by the IBM Multi-site Workload Lifeline for z/OS (also known as Lifeline) product and the external load balancers. For more information, see *IBM GDPS Active/Active Overview and Planning*, SG24-8241, at:
    http://www.redbooks.ibm.com/abstracts/sg248241.html?Open

- **Infrastructure**:

- Automate restart and recovery of CICS regions and related servers. CICS regions and any related servers should be registered with ARM or an automation product like IBM System Automation for z/OS to facilitate automated restart after failures.
- Enable alerting and automation for key CICS system resources such as task limits, virtual storage, data space usage, and so on. CICS System Policies, OMEGAMON XE for CICS and CICSPlex SM RTA all provide flexible facilities to monitor CICS systems and applications.

  Note: CICSPlex SM Real-Time Analysis (RTA) is stabilized. The strategic replacement for CICSPlex SM RTA is CICS System rules or dedicated monitoring products, such as those provided by the IBM OMEGAMON XE Family.

- Configure realistic system limits and priorities:
  - Use z/OS Workload Manager to define the performance goals for CICS regions and related subsystems so that when CPU becomes constrained subsystems do not become overwhelmed with new work requests. Allocate each CICS job a service class and then specify target response times for the service class. Typically, production regions and test regions are placed in different service classes, because response times for production regions are more critical than for test regions. In addition, TORs and routing regions might be separate categories from AORs and data-owning regions.
  - Configure and monitor CICS system limits, in particular max tasks (MXT), transaction class limits and priorities, sessions, TCB and thread limits, network backlogs and CICS dynamic storage areas (DSAs).
  - Use prioritization techniques, such as CICS transaction classes, or WLM service classes to ensure that high priority requests will still function correctly when the system is heavily loaded.
  - Do not set system limits that are unachievable given the current system resource allocations. Remember when the system is overloaded it is more efficient to delay or even reject requests so that they queue outside of z/OS rather than internally.

- **Data sharing**:

  Consider using sysplex data sharing technologies to ensure that there are no single points of failure for access to data. Sharing access to data also enables efficient workload distribution across a Parallel Sysplex.

  - Consider using VSAM Record Level Sharing (RLS) for better scaling capabilities and high availability.
  - Consider IMS and Db2 data sharing to enable shared access to data across the sysplex
  - Consider using a CICS temporary storage server and a coupling facility data table server, for better availability of data tables and TSQs across the sysplex.
  - Consider using IBM InfoSphere Data Replication for VSAM to produce copies of your VSAM data sets and maintain current data in near-real time, typically on geographically dispersed sysplexes.
  - Consider exploiting Multi-Site Workload Lifeline for a new Continuous Availability solution with CICS. Workload distribution and workload balancing in a GDPS Continuous Availability environment is achieved by the IBM Multi-site Workload Lifeline for z/OS (also known as Lifeline) product and the external load balancers. For more information, see *IBM GDPS Active/Active Overview and Planning*, SG24-8241, at: http://www.redbooks.ibm.com/abstracts/sg248241.html?Open

- **Connectivity**:

  Traditionally, workload was routed into CICS regions by using SNA via IBM VTAM. However, TCP/IP-based connectors are now the predominant connectors, examples being Web Services, CICS Transaction Gateway, Communication Server IP sockets, IBM MQ, and so on.

  - Consider high availability solutions for each of the connector technologies components to further reduce single points of failure. Options include Sysplex Distributor, MQ queue sharing, CICS TG dynamic server selection, and VTAM Generic Resource.
  - Ensure that there is a good balance between HTTP connection persistence and cycling of connections to ensure that connection balancing can continue to operate as web-owning regions are recycled.

Tip: To maximize the resilience benefits of MQ shared queue support, be sure to enable MQ group UR support by specifying RESYNCMEMBER(GROUPRESYNC): https://www.ibm.com/support/knowledgecenter/SSGMCP_5.4.0/reference/resources/mqconn/dfha4_attributes.html

In addition, connect CICS regions in multiple LPARs to data resources.

4.4.2 IMS

Consider backing up IMS logs at regular intervals. Consider exploiting dual logging or multi copy data sets options wherever applicable as it offers protection against data loss. The data sets might be placed on different storage volumes to make the system more resilient. Images copies can be taken at regular intervals to reduce recovery time.

Note: Consider exploiting dynamic refresh enhancement for pseudo-wait-for-input (PWFI) or wait-for-input (WFI) regions to introduce application updates into running systems.

Before IMS 14, when application program changes were made in IMS PGMLIB, you had to find all the regions to stop. The following time consuming manual step was required: *Stop all PWFI regions or regions that run WFI=YES transactions where the program was scheduled so that the new changes could be obtained when the program was scheduled again.*

Consider using Database change accumulation utility to streamline the recovery information you provide to the Database Recovery utility. Using it periodically speeds any database recovery that becomes necessary. Consider testing your backup and recovery procedures to make sure that they work as desired. Consider registering IMS regions with ARM for automated in-place or cross system restarts. Alternatively, use automation products like IBM System Automation for z/OS to restart failed regions automatically.

Tip: Consider using Fast Path to improve performance for simple transactions. When data communication requirements are for a high transaction volume with rapid database updates and inquiries, the Fast Path facilities offer several advantages over full-function DL/I processing. For a DB·DC environment, Fast Path requires the Database Manager and Transaction Manager and becomes an integral part of the IMS online system. The control program manages concurrent processing of Fast Path and DL·I programs. The DCCTL environment supports Fast Path processing and transactions, but not Fast Path databases.

For more information on the design, definition, initialization, monitoring, or tuning of databases used with Fast Path, see IMS Version 15 Database Administration: ftp://public.dhe.ibm.com/software/IMS/documentation/ims15pdfs/IMS_15_Database_Administration.pdf

For information on Fast Path application programming, see IMS Version 15 Application Programming: ftp://public.dhe.ibm.com/software/IMS/documentation/ims15pdfs/IMS_15_Application_Programming.pdf

Consider configuring multiple IMS images to run in a Parallel Sysplex environment to balance workload, build scalability, and increase availability. Sharing of message queues and databases enables workload balancing and insulation from individual IMS outages. You might want to use sysplex distributor to distribute the workload between the different IMS images in the sysplex.

Note: A HALDB is a partitioned full-function database. By partitioning a database, you allow the use of smaller data sets, which are easier to manage. Multiple partitions increase the amount of available data if one of the partitions fails or is taken off line.

If your organization uses IMS Full Function databases, consider converting them into IMS High Availability Large Database (HALDB). HALDB is designed to increase the capacity and availability of full-function databases.

Tip: Consider using IMS High Availability Large Database Toolkit for z/OS to convert existing IMS Full Function databases into IMS HALDB databases. You can use the toolkit to simulate changes to partition settings to ensure that they are correct before you implement those changes.

Consider the use of supermember support for IMS Connect and shared message queues. This is a way to achieve higher levels of availability. Also, consider implementing IBM InfoSphere IMS Replication for z/OS and using Continuous replication or Site Transition replication models to address data mirroring requirements.

Based on your site's RTO and RPO requirements consider implementing effective backup, restore mechanisms offered by IMS components and utilities. Also consider using various IBM product suites like IMS System Management for z/OS and IMS Recovery Solution pack. Refer to 2.6, “Management layer” on page 81 for a short description of the features provided by these components and IBM IMS Program Restart Facility for z/OS, IBM InfoSphere IMS Replication for z/OS, and IBM IMS Recovery Expert for z/OS product offerings.

- For more information, see the *IMS and IMS Connect High Availability* topic in IBM Knowledge Center:

https://www.ibm.com/support/knowledgecenter/en/SS4SVW_3.0.0/highavailability/ims_sp_ha.html

- For more information, see the *Overview of IMS Replication* topic in IBM Knowledge Center: https://www.ibm.com/support/knowledgecenter/en/SS4T2J_11.3.0/com.ibm.swg.im.iis.clz.imsr.overview.doc/topics/iiyitoioverview.html

4.4.3 WebSphere Application Server for z/OS

There are two implementations of the WebSphere Application Server - traditional and Liberty. Irrespective of the implementation, consider configuring WebSphere Application Server in a cluster to achieve high availability for the applications hosted on the servers. WebSphere Application Server provides different options to configure a cluster based on the user requirements. The different clustering options are: static clustering, dynamic clustering, vertical clustering, and horizontal clustering.

Static and dynamic clustering, both provide workload balancing features, but with dynamic clustering you can further optimize the cluster performance.

Vertical clustering topology refers to adding more servers on the same node or system (as redundancy). In contrast, horizontal clustering topology refers to adding more servers on multiple nodes (preferably across systems) in a cell.

Depending on the requirement, a right mix of vertical and horizontal clustering can help you increase the availability of your applications that are hosted on these application servers.

Consider implementing High Availability manager feature of WebSphere Application Server with thoughtful consideration. A high availability manager consumes valuable system resources, such as CPU cycles, heap memory, and sockets. These resources are consumed both by the high availability manager and by product components that use the services that the high availability manager provides. The amount of resources that both the high availability manager and these product components consume increases nonlinearly as the size of a core group increases.

Depending on the requirement, a network deployment configuration with on-demand router, clustering (mix of vertical, horizontal, and dynamic clustering) with high availability manager can provide extremely high levels of performance, scalability, and availability in an enterprise.

- For more information, see the *Establishing High Availability* topic in IBM Knowledge Center: https://www.ibm.com/support/knowledgecenter/en/SS7K4U_8.5.5/com.ibm.websphere.zseries.doc/ae/welc6tophighavail.html
- For more information, see *WebSphere Application Server V8.5 Concepts, Planning, and Design Guide*: http://www.redbooks.ibm.com/abstracts/sg248022.html?Open

4.4.4 User-written application

Consider identifying application code that has affinity to any LPAR or resource. After you identify these instances of application code, consider managing the affinity so that the application could be deployed on any LPAR in the sysplex. The applications can be enabled to exploit data sharing and other Parallel Sysplex features and functionalities to avoid single point of failure.

Note: With processor affinity, application code binds a process to a single CPU (or it can bind the process to a specific group of CPUs). This approach can be much more efficient for some processes, especially when the processing typically starts and stops multiple times before is completed. At the same time, processor affinity coding might undermine load-balancing policies. For this reason, you must assess the effects of code that uses processor affinity.

Consider taking a closer look at jobs or started tasks. Identify the ones that regularly cause enqueue contention, leading to 'deadly embraces'. Study the schedule, architecture, or design of these jobs or started tasks or transactions in detail and try to minimize enqueue contention.

For more information, see *Parallel Sysplex Application Considerations*: https://www.redbooks.ibm.com/redbooks/pdfs/sg246523.pdf

You might also consider converting or developing threadsafe applications to maximize concurrency, and minimize task switching. Threadsafe applications provide relief on quasi-reentrant task control block (QR TCB) which not only can reduce requirements to clone regions, but also has potential for region consolidation.

4.4.5 Batch processing

Batch processing is pervasive in the IBM Z environment but is often not given the same attention as mission critical online transaction processing (OLTP). Yet there are vital batch jobs that need to complete successfully for business-related purposes.

Resiliency for batch processing can be defined in two ways:

- Reducing the window by allowing batch jobs to run together with online workloads to support a 24x7 always-on environment.
- Recovering batch jobs after system or batch processing failures

Improving performance of batch processing by reducing its elapsed time can also reduce the batch window, and thus improving availability for online workloads. There are many techniques to allow this, such as reducing the I/O response time, reducing the amount of I/O, parallelizing the batch job or job step, use of WLM-managed initiators, or just improving the code efficiency. For more information on these techniques: *Batch Modernization on z/OS*, SG24-7779 (http://www.redbooks.ibm.com/abstracts/sg247779.html?Open) and *Approaches to Optimize Batch Processing on z/OS*, REDP-4816 (http://www.redbooks.ibm.com/abstracts/redp4816.html?Open).

This section presents some techniques to help provide better availability for batch processing.

Non-database related recovery for batch files

In planning your overall availability strategy, consider general backup and recovery of z/OS files, z/OS UNIX files, volumes, and recovery of these from failures.

Backup of data is needed to guard against users accidentally losing or incorrectly changing their data sets and against losing volumes because of hardware failures. Do this at both the file and the volume level. Depending on how much data gets changed, it might be more efficient to do incremental backup (logical backup of those tracks that changed since they were last backed up). This minimizes processing time because you are not backing the entire

file. Logical backup, with the exception of logical backups to an object storage cloud, lets you restore data sets to unlike devices.

There are many issues that need to be considered:

- How do you identify data dependencies?
- No easy way to identify all of the data used by an application
- Need for data dependency knowledge to extract data for certain application workloads to ensure a successful move to other sysplexes or data center locations
- If a failure happens in the middle of a batch workload that has data dependencies, which files need to be restored?
- Which GDG member should be used?
- How do you identify missing backups and duplication of backups?
- Important to understand for disaster recovery
- How do you manage and optimize backup resource usage?
- Data sets being backed up needlessly use large amounts of CPU, storage, and time
- How do you manage and track information for batch jobs that are run outside the scope of the job scheduler utility?
- How do you recover from data corruption or accidental deletion of a file?
- Where is the data kept?
- Who is responsible? Application owners or storage administrators?
- Where is the process documented? How can you prove regulatory compliance?
- How do you prove you're recoverable, and who is responsible for recovery? Are you auditable?
- How can you test, and prove, the process works?
- Do you have the reporting needed to document your gaps as well as your successes?
- How can this be done in an automated solution that doesn't rely on mainframe and application expertise?
- How do you roll back batch in time and know all the data that's been affected by a failure?

Many of these issues can be addressed by IBM Z Batch Resiliency (IZBR). IZBR automates the analysis, backup, and restore of batch application data to provide operational resiliency and reduction of business risk. It provides an inventory of data usage and backups and has the automation capabilities to recover batch data quickly for operational or disaster recovery events. In addition, it can generate reports that detail the usage and relationships of batch applications and data.

Tip: Consider using IBM Z Batch Resiliency (IZBR) to automate recovery of the batch environment. For more information, see:
https://www.ibm.com/us-en/marketplace/z-batch-resiliency/details

As there can be thousands of batch jobs that run over the course of a week, it is important to maintain control over the batch environment. Because the online environment might depend on when the batch window completes, batch performance can be considered part of the overall availability considerations for the system. There are several considerations for this, including:

- Have an enforced naming convention. WLM management is based on the job name. So, it is preferable to make sure a given job gets the resources it needs to finish on time, but not take away resources from other jobs that have higher priority.
- Use WLM-managed initiators to prevent too many concurrent jobs, which can slow down the throughput.
- Focus on reducing the critical job path. There are several methods to do this including breaking up a job with separate steps into multiple jobs that can run in parallel, or even parallelizing a step.
- Use Flash enclosures on the external storage to minimize I/O delays.
- IBM Workload Scheduler for z/OS has a feature that allows to switch the job's WLM service class to another if the job is late or long running. The decision is based on knowledge of the job's estimated duration (defined by the user and updated at each run on real execution time) and the job dependencies from other jobs, time, or other resources. IBM Workload Scheduler for z/OS:
  - Detects that a critical job is running late.
  - Calls WLM to move the job to a higher performance service class.
  - Conveys additional system resources so that the job can complete in a shorter time.

  For more information on job scheduling and WLM, see:
  https://www.ibm.com/support/knowledgecenter/en/SSGSPN_9.5.0/com.ibm.tivoli.itws.doc_9.5/zos/src_man/eqqr1jschedwlmchapt.htm

Db2 batch processing

Considerations for better availability for Db2 batch jobs all relate to acquiring as few locks as possible and releasing them as soon as possible. For more information, see:
https://www.ibm.com/support/knowledgecenter/en/SSEPEK_11.0.0/perf/src/tpc/db2z_programapps4concurrency.html

While the first three rules of real estate are "Location, location, location," the first three rules of Db2 batch are "Commit, Commit, Commit." There are (more than three!) reasons for this, such as:

- Reduces lock contention, especially in a data sharing environment
- Reduces potential for deadlocks
- Improves the effectiveness of lock avoidance, especially in a data sharing environment
- Faster rollback / backout processing
  All locks acquired during the unit of work are released. LOB locators are freed.
- Reduces the elapsed time for Db2 system restart following a system failure
- Reduces the elapsed time for a unit of recovery to roll back following an application failure or an explicit rollback request by the application
- Provides more opportunity for utilities, such as online REORG, to break in

These reasons are equally valid with IMS and CICS environments.

Reduces lock contention

Locks are accumulated during the course of the batch processing. These locks are released when the Unit of Work completes. If a second job or transaction wants to acquire a lock held by the first job, it will need to wait until the first job is done with it. Most locks can be released earlier if a Commit is explicitly issued. This allows the second job or transaction to run sooner, while reducing CPU to manage the contention.

Reduces deadlocks

A *deadlock* occurs when two or more application processes each hold locks on resources that the others need and without which they cannot proceed. After a preset time interval, Db2 can roll back the current unit of work for one of the processes or request a process to terminate. Db2 must determine which process to roll back or terminate. In doing so, Db2 assesses many characteristics of the processes that are involved in the deadlock and chooses the one that, if terminated, will cause the least impact relative to the other processes. By choosing a process to roll back or terminate, Db2 frees the locks and allows the remaining processes to continue. This process is similar to IMS batch.

Consider an example where two Db2 batch jobs run different applications but access the same data, each needing to process 1,000,000 records. Without any explicit Commits being coded, each job will accumulate up to 1,000,000 locks as it processes the data. A deadlock is likely if Job1 tries to get a lock held by Job2, while Job2 is waiting for a lock held by Job1. However, if you change the batch logic so that each job issues a Commit after 1,000 locks, on the average each job holds 500 locks at any one time instead of 500,000. The chances of deadlocks are greatly reduced.

Rollback

An SQL ROLLBACK statement is called if a transaction cannot complete successfully. For example, a travel reservation transaction might involve booking an airline flight and then a hotel room. If a flight gets reserved but a hotel room cannot be reserved, the application process might want to undo the flight reservation as well. Db2 will undo the changes made by the transaction until the latest Savepoint or Commit. The more changes made, the longer this process takes, while the transaction holds on to the locks obtained during this time.

Faster restart

If the Db2 system fails, all in-flight units of work are backed out in a similar process as the Rollback. The more updates there were since the last Commit, the longer it takes to roll back out the changes and allowing new work to start.

Faster disaster Recovery

Db2 logs record transactions and Db2 batch start and end times, updates, and other activities into a (write-ahead) log buffer. The log buffer is written to disk either when it becomes full or when the transaction or batch job issues a commit. The commit is not successful until it has been externalized from the log buffer to the log files on disk.

Tip: Consider using the URCHKTH subsystem parameter or the URLGWTH subsystem parameter to identify applications that do not commit frequently. URCHKTH identifies when too many checkpoints have occurred without a Unit of Recovery (UR) issuing a commit. It is helpful in monitoring overall system activity. URLGWTH detects when applications might write too many log records between commit points, potentially creating a lengthy recovery situation for critical tables.

Even though an application might conform to the commit frequency standards of the installation under normal operational conditions, variation can occur based on system workload fluctuations. For example, a low-priority application might issue a commit frequently on a system that is lightly loaded. However, under a heavy system load, the use of the CPU by the application might be preempted. And as a result, the application might violate the rule set by the URCHKTH subsystem parameter. For this reason, add logic to your application to commit based on time elapsed since last commit, and not solely based on the amount of SQL processing performed. In addition, take frequent commit points in a long running unit of work that is read-only. By doing so, you reduce lock contention and provide opportunities for utilities, such as online REORG, to access the data.

Committing frequently is equally important for objects that are and are not logged. Make sure that you commit work frequently even if the work is done on a table space that is defined with the NOT LOGGED option. Even when a given transaction modifies only tables that reside in not logged table spaces, a unit of recovery is still established before updates are performed. Undo processing reads the log in the backward direction as it looks for undo log records that must be applied. The processing continues until it detects the beginning of this unit of recovery as recorded on the log. Therefore, such transactions should perform frequent commits. In this way, you limit the distance that undo processing might have to go backward on the log to find the beginning of the unit of recovery.

Include logic in your application program to retry after a deadlock or timeout to attempt recovery from the contention situation without assistance. Such a method might help you recover from the situation without assistance from operations personnel. You can use the following methods to determine whether a timeout or deadlock occurs:

- The SQLERRD(3) field in the SQLCA
- A GET DIAGNOSTICS statement
- Reduce the amount of locks acquired. Whenever possible, bind applications with the ISOLATION(CS) and CURRENTDATA(NO) options. These options enable Db2 to release locks early and avoid taking locks in many cases. ISOLATION(CS) typically enables Db2 to release acquired locks as soon as possible. The CURRENTDATA(NO) typically enables Db2 to acquire the fewest number of locks, for better lock avoidance. When you use ISOLATION(CS) and CURRENTDATA(NO), consider using the SKIPUNCI subsystem parameter value to YES so that readers do not wait for the outcome of uncommitted inserts.
- If you do not use ISOLATION(CS) and CURRENTDATA(NO), use the following bind options:
- ISOLATION(CS) with CURRENTDATA(YES), when data returned to the application must not be changed before your next FETCH operation.
- ISOLATION(RS), when data returned to the application must not be changed before your application commits or rolls back. However, you do not care if other application processes insert additional rows.
- ISOLATION(RR), when data evaluated as the result of a query must not be changed before your application commits or rolls back. New rows cannot be inserted into the answer set.
- Use ISOLATION(UR) option cautiously. The Resource Recovery Services attachment facility UR isolation acquires almost no locks on rows or pages. It is fast and causes little contention, but it reads uncommitted data. Do not use it unless you are sure that your applications and end users can accept the logical inconsistencies that can occur. As an alternative, consider using a SKIP LOCKED DATA clause if omitting data is preferable to reading uncommitted data in your application.

Use group names

You must ensure that any of your Db2 work can run on any member of the data sharing group in a Parallel Sysplex. As part of this process, you must make sure that batch jobs, as well as CICS regions, IMS subsystems, WebSphere Application Server, and MQ servers specify the Db2 group name rather than a specific Db2 subsystem name. If a specific subsystem is unavailable, then the work can still connect to another Db2 in the same group.

There might be cases in which you want to run Db2 work on a particular system, not because you need to, but because you want to. One example might be that you want to run all the Db2 batch jobs for a given application on one system in an attempt to avoid GBP dependencies. The most flexible way to address this is to use WLM Scheduling Environments. By turning the

Scheduling Environment on or off on separate systems, you can easily control which systems are eligible to run those jobs.

IMS batch processing

Issuing checkpoint calls is an important part of batch-oriented BMP processing.[1] In addition to committing the changes, the checkpoints free locks. Such locks are held to avoid contention with transactions and other batch jobs. And the locks provide a location to back out work to if there are any failures, and a location to restart from. If a program fails while it runs in a batch region, it must be restarted in a batch region. If a program fails in a BMP region, it must be restarted in a BMP region.

Tip: Code your checkpoints in a way that makes them easy to modify. Converting a batch program to a BMP or converting a batch program to use data sharing requires more frequent checkpoints.

Because they do locking, BMPs should issue Checkpoint calls more frequently than batch programs to release DB locks for online transactions.

Tip: To minimize the batch window and to simplify recovery, BMPs should be used instead of DLI or DBB batch.

Recovering a transaction-oriented BMP

BMP can perform an update for an MPP. You can design the BMP so that, if it terminates abnormally, you can reenter the last message as input for the BMP when you restart it. For example, suppose an MPP gathers database updates for three BMPs to process, and one of the BMPs terminates abnormally. You would need to reenter the message that the terminating BMP was processing to one of the other BMPs for reprocessing.

Like MPPs, with transaction-oriented BMPs, you can choose where commit points occur in the program. You can specify that a transaction-oriented BMP be single or multiple mode, just as you can with an MPP. If the BMP is single mode, a commit point occurs each time that the program retrieves a message. Backouts and restarts are done from that checkpoint.

Recovering a batch-oriented BMP

Checkpoint points do not occur automatically, as they do in transaction-oriented BMPs, so checkpoints / sync points call need to be explicitly done by the application. See the following presentation for details:
https://ibm-zcouncil.com/wp-content/uploads/2018/05/IMS-DB-Recovery-procedures.pdf

BMPs simplify program recovery because logging goes to a single system log. If you use disk for the system login batch, you can specify that you want dynamic backout for the program. In that case, batch recovery is similar to BMP recovery, except, of course, with batch you need to manage multiple logs.

If you are running sysplex data sharing and you either have IMS TM or are using DBCTL, you might want to convert your program. This is because the use of batch-oriented BMPs helps you stay within the sysplex data-sharing limit of 32 connections for each OSAM or VSAM structure.

[1] https://www.ibm.com/support/knowledgecenter/en/SSEPH2_15.1.0/com.ibm.ims15.doc.apg/ims_bmpprocessingbatchoriented.htm

If you use data sharing, you can run batch programs concurrently with online programs. If you do not use data sharing, converting a batch program to a BMP makes it possible to run the program with BMPs and other online programs.

Converting DLI/DBB batch to BMPs enables you to run them with the online system, instead of waiting until the online system is not running. Running a batch program as a BMP can also keep the data more current.

Basic and Symbolic checkpoints

BMP, JMP, and batch programs can use either symbolic checkpoint calls or basic checkpoint calls.

Issuing a checkpoint call causes IMS to:

- Inform Db2 for z/OS that the changes your program have made to the database can be made permanent. Db2 for z/OS and IMS makes the changes to the data permanent.
- Write a log record containing the checkpoint identification that is given in the call to the system log, but only if the PSB contains a DB PCB.
- Send a message containing the checkpoint identification that was given in the call to the system console operator and to the IMS master terminal operator.
- For transaction-oriented BMPs, a checkpoint call acts like a call for a new message. And the next input message is returned to the program's I/O area.

Consider the following when you issue checkpoint calls in multiple-mode programs:

- The program should issue checkpoints frequently enough to make the program easy to back out and recover. A general recommendation is to issue one checkpoint call every 10 or 15 minutes. One can also issue a checkpoint call after a certain number of root segments the program accessed, or after a certain number of updates made.
- To free database resources for other programs, batch programs that run in a data-sharing environment should issue checkpoint calls more frequently than those that do not run in a data-sharing environment.
- Programs should issue checkpoint calls frequently enough to avoid building up too many output messages.
- Depending on the database organization, issuing a checkpoint call might reset your position in the database.

Symbolic checkpoint

Symbolic checkpoints are similar to basic checkpoints, but programs that issue symbolic checkpoint calls can specify as many as seven data areas in the program to be checkpointed. When IMS restarts the program, the Restart call restores these areas to the condition they were in when the program issued the symbolic checkpoint call. Because symbolic checkpoint calls do not support z/OS files, if your program accesses z/OS files, you must supply your own method of establishing checkpoints.

The restart call, which you must use with symbolic checkpoint calls, provides a way of restarting a program after an abnormal termination. It restores the program's data areas to the way they were when the program issued the symbolic checkpoint call. It also restarts the program from the last checkpoint the program established before terminating abnormally.

Generalized Sequential Access Method

Generalized Sequential Access Method (GSAM) is an IMS access method that treats sequential files as if it was a database and thus support recovery processes. GSAM files can

be BSAM, QSAM, or VSAM. An IMS program can retrieve records and add records to the end of the GSAM database, but the program cannot delete or replace records in the database.

To avoid x37 ABENDs, the GSAM file needs to be large enough so you do not run out of space and need to back out and restart the program. The IBM Tivoli Advanced Allocation Management for z/OS tool is recommended to help stop x37 ABENDs

Basic CHKP calls cannot checkpoint GSAM databases. Your program must use symbolic CHKP and XRST calls if it uses GSAM. These calls can be used to reposition the data set at the time of restart, enabling you to make your program restartable. When you use an XRST call, IMS repositions GSAM databases for processing. CHKP and XRST calls are available to application programs that can run as batch programs, batch-oriented BMPs, or transaction-oriented BMPs.

The GSAM file can be a GDG data set, but if restarting a failed program, JCL changes are needed to point to the right GDG member.

Recovering DLIBATCH and DBBBATCH jobs

If DLI or DBB batch is run concurrently with other IMS regions, it must be in data sharing mode. With that, checkpoints are needed to free up IMS resources to allow other work to continue. These checkpoints optimally should be coded at logical locations in the application. But they are coded as described above to checkpoint after a specified time or after a specified amount of log records are processed. A disadvantage is that there are many jobs in an application, and it is time consuming to manually code checkpoints. One solution is to convert these jobs to BMPs.

In addition to the need to code checkpoints, from a recovery and resiliency point of view, there are several shortcomings to running DLI / DBB Batch:

- DLI/DBB program ABENDs require manual intervention prior to restart
  - JCL to close the batch log, JCL to run batch backout.
  - Databases remain unavailable until batch backout runs successfully.
  - IRLM locks remain in a data sharing environment
  - Application data remains unavailable until batch backout runs successfully.
  - Error-prone. Wrong logs can be used for backout or restart.
- Restarting an ABENDed program requires JCL changes
  - Look up last checkpoint ID from prior execution, modify EXEC parm to specify the checkpoint ID.
  - Add IMSLOGR DD statement to point to log file that contains last checkpoint ID
  - BMP changes are minimized (CKPTID=LAST) in most cases, but can still revert to pre-IMS V11 behavior if you wait too long to restart.
- U0777/U3303 ABENDs
  - Deadlocks and unavailable resources cause BMPs to abend, requiring unnecessary restarts

There are tools such as the IBM IMS Program Restart Facility (IMS PRF) solution that can significantly simplify the recovery process. IMS PRF automatically calls log termination and batch backout utilities as part of the job ABEND process. No JCL changes are required. Because the process is automatic, it is faster, greatly enhancing availability. If a system outage occurs, log termination and backout is done at job restart. In addition, no JCL changes are required. PRF keeps track of checkpoints. When a job is a restarted after an ABEND, PRF will dynamically modify the CKPTID EXEC parm with the correct checkpoint ID. And it dynamically allocates the correct file to IMSLOGR to pick up the restart checkpoint information. After user-specified ABENDs such as U0777 or U3303, PRF can automatically

restart these jobs according to user specifications such as, delay between retries, and how many tries.

IMS PRF can also be used to help manage BMP job failures. See , “IMS Program Restart Facility for z/OS” on page 90 for more information.

VSAM Record Level Sharing and Transactional VSAM

Batch workloads can be run under Db2 and IMS databases. Because the database managers do logging of updates, if there are batch job failures the Db2 or IMS can back out the updates. Because the database managers, with IRLM, do locking, the jobs can run simultaneously with online transactions. The batch job acts like a transaction, although it runs a bit longer.

In CICS environments, many batch jobs access VSAM files that are also used by CICS. But VSAM is not a database manager. It does not do logging or locking. Organizations want to prevent data integrity issues from running online together with Batch and want to handle backouts if there are problems. Many organizations follow a procedure like this one, at the beginning of the nightly batch cycle:

1. Quiesce all online activity to the files used by the batch jobs
2. Create “checkpoint” files to fall back to. Give it a name such as “file.NEW.”
   - If there are any problems with the batch jobs that modify “file.NEW”, then the original unchanged file is still available.
   - File.NEW can be created by invoking DFSMSdss with FASTREPlication defined as the default. If the source device supports data set FlashCopy then DFSMSdss optionally uses FlashCopy to provide a virtual concurrent copy.
3. DFSMSdss releases any serialization that it holds on the data sets and prints a message to SYSPRINT and the console that the concurrent copy operation is logically complete. If DFSMSdss was invoked through the API, DFSMSdss informs the caller through the UIM exit option, Eioption 24, and the application can resume normal operation.
4. Start the batch jobs, accessing the “file_NEW” copy.
5. If the job completes successfully, delete the original file, rename the “file_NEW” to the original name and allow the online work to resume.
6. If the job failed, then delete “file_NEW”. Debug and fix the job and run again.

As part of this, you need to verify that you have made copies of all files that are being updated by the batch workload. Different SMF records can be scanned to identify the files being used, as in these examples:

- SMF 15 records are written when non-VSAM data sets — VIO tape data sets that are defined by DD statements or dynamic allocation — are opened for OUTPUT, UPDATE, INOUT, or OUTIN processing by problem programs. It is written when a data set is closed or processed by the end-of-volume (EOV).
- SMF 30 records contain operation information such as the job and step start and end times, number of records in DD DATA, and DD * data sets processed by the job step. The record contains an entry for each data set defined by a DD statement or dynamic allocation. Each entry lists the device class, unit type, device number, the execute channel program (EXCP) count, and more.
- SMF 62 records are written when a VSAM component or cluster is opened. It identifies the VSAM component or cluster, volume, and job name and time.
- SMF 64 records are written when a VSAM component or cluster is closed.

There are variations to this, but the issue is that the online transactions need to stop during the nightly batch window. This is resolved by using VSAM Record Level Sharing (RLS) with Transactional VSAM (TVS).

VSAM record-level sharing (RLS)

VSAM record-level sharing (RLS) extends the DFSMS storage hierarchy to support a data-sharing environment across multiple systems in a Parallel Sysplex. This support is primarily for VSAM data sets that online transaction-processing applications use.

VSAM RLS is a data set access mode that enables multiple address spaces, CICS application-owning regions on multiple systems, and batch jobs to access recoverable VSAM data sets at the same time.

With VSAM RLS, multiple CICS systems can directly access a shared VSAM data set, eliminating the need to ship functions between the application-owning regions and file-owning regions. CICS provides the logging, commit, and backout functions for VSAM recoverable data sets. VSAM RLS provides record-level serialization and cross-system caching. CICSVR provides a forward recovery utility. The level of sharing that is allowed between applications is determined by whether a data set is recoverable, as in these examples:

- Both CICS and non-CICS jobs can have concurrent read or write access to unrecoverable data sets. However, there is no coordination between CICS and non-CICS, so data integrity can be compromised.
- Non-CICS jobs can have read-only access to recoverable data sets concurrently with CICS jobs, which can have read or write access.

If a transactional VSAM job fails, CICS will back out any updates. If there are checkpoints in the job, the updates will be backed out to the most recent checkpoint to maintain data consistency. The problem should then be fixed and the job rerun. Because the job is now being restarted from somewhere in the middle of processing, logic needs to be added so the job knows where to restart and key variables saved across invocations.

Depending on what functions the process provides, properly implementing restart might be far from trivial. Many large batch update jobs (which are the ones that might even be candidates for restart) do more than simply update tables. Audit trail reports are often created, sequential files might be created and so on. Everything must be resynchronized if a restart is to be implemented.

In most applications, there is no reason to build in restart procedures due to the speed increases in CPCs and disk. It is cleaner and easier to restart from the beginning.

Transactional VSAM (TVS) Auto-commit

TVS enables batch jobs and CICS transactions to concurrently update shared recoverable VSAM data sets. As a result, it reduces the batch window for CICS and other VSAM applications and allows 24x7 access to VSAM recoverable data. With TVS, multiple batch jobs and CICS transactions can process with the same recoverable VSAM data sets, and data integrity is ensured for both CICS and concurrent batch updates. Today, to fully exploit TVS, batch applications must be examined to determine the commit frequency. Then they must be modified to use RRS to issue sync point commits and potentially backouts to avoid elongated CICS response times.

It is sometimes possible to modify a batch job step to exploit DFSMStvs without making source code changes. However, that approach means that the entire job step is treated as a single VSAM transaction, which can cause locking, logging, concurrency, and backout issues. To avoid those issues, sync points need to be added to the source code in most cases. In some cases, such as jobs that perform only a small number of updates, adding and testing

sync points might require more effort than is necessary. For such programs, the DFSMStvs automatic commit function might be the answer.

The automatic commit function can provide automatic commits of transactions for eligible batch jobs that use DFSMStvs. Eligible jobs are those for which a single update to a single data set constitutes an atomic transaction or logical unit of work. For example, if a batch job modifies two data sets — depending on the parameters that are specified — DFSMStvs might issue a sync point between the two updates, and one data set might become out of sync with the other data set. If such updates need to be atomic, an automatic commit is not suitable for this application.

You can code the optional TVSAMCOM parameter on the JCL EXEC statement as follows:

//stepname EXEC positional-parm, TVSAMCOM=({minval|0}),({maxval|0})

The minval and maxval values specify the minimum and maximum for a range of numbers of update requests. DFSMStvs uses this range to determine when and if to call RRS services to issue a commit point on behalf of the batch application. DFSMStvs adjusts the commit frequency to a number between the two specified values, based on record lock contention for the current unit of recovery and CICS transactions 'timeout value. Additionally, you can specify a similar system-level commit parameter in the IGDSMSxx member of SYS1.PARMLIB.

DFSMStvs performs an automatic commit when any of the data sets accessed by the unit of recovery is closed. This is done to avoid leaving record locks held by the unit of recovery until the implicit commit is issued at end of task. Any of the units of recovery that are eligible for automatic commit could cause the lock structure to start becoming full as a result of obtaining record locks. In this case, DFSMStvs performs an automatic commit when the lock structure is approximately 80% full.

For more information, see: *z/OS DFSMStvs Planning and Operating Guide*, SC23-6877: https://www-01.ibm.com/servers/resourcelink/svc00100.nsf/pages/zOSV2R3sc236877?OpenDocument

4.5 Considerations for the management layer

A robust and well versed operational and system management procedure is key for a highly available or continuously available system. There are various aspects to management systems that help to keep the system healthy and running for 24x7 operations. Some of the critical system management components are listed here:

- Monitoring Operating systems, program products, applications, and more
- Maximum Automation
- Robust Change Control Procedures
- Capacity Planning for computing power and storage
- Data Backup and Replication
- Maintenance strategy
- Security

IBM offers various solutions in all the areas that are mentioned in this list. These solutions can help organizations to raise their highly available systems to a higher level. Chapter 2 discusses several products in these areas in detail. The current section discusses a few important considerations for system management procedures that affect high availability.

For more information, see *End-to-End High Availability Solution for System z® from a Linux Perspective*:
http://www.redbooks.ibm.com/abstracts/sg248233.html?Open

4.5.1 IBM Z Operations Analytics

The IBM Z Operations Analytics (IZOA) tool helps organizations to search, visualize, and analyze large amounts of structured and unstructured operational data across IBM Z environments. This data includes log, event and service request data, and performance metrics. The tool helps you identify issues in the workload, locate hidden problems, and perform root-cause analysis faster.

IZOA has pattern recognition that intercepts application and system problems before they cause disruptions. Its real-time, self-learning solution accurately represents your environment automatically. The tool also supports Linux on Z for enhanced management on that platform. In fact, the same graphical user interface is used for monitoring z/OS and Linux on Z.

Organizations can consider installing IZOA to leverage all the features for making their applications on IBM Z highly available.

Note: IBM Z Operations Analytics is available on these platforms: IBM Operations Analytics - Log Analysis, Elastic Stack, and Splunk. The user interface and available functions can vary depending on the platform.

For more information, see these resources:

A demonstration of how IZOA works:
http://zscala.ibmzoperationsanalytics.com:9182/ZLALiveDemo/guide.html

IZOA product details:
https://www.ibm.com/us-en/marketplace/operations-analytics-for-z-systems/details

4.5.2 Data backup and replication

For resiliency and availability in your system, you must have a good control (and knowledge) of the data that is generated and stored in enterprise storage servers. A strategy for storage backup that is well thought out often dictates what RPO and RTO can be achieved in case of a disaster or component failure. Typically, storage administrators work closely with different teams within the organization to gather their storage and backup requirements. Then, these administrators frame the data backup plan and data recovery plan. The storage administrator must ensure that all the required data is periodically backed up (or replicated). Then, in the event of a disaster, it will be possible to restore everything that leads to the business applications, according to the specifications of the RTO and RPO parameters. This restoration includes many types of locations for data storage: disks or tapes that are related to applications, operating systems, program products, product license keys, databases, transactions servers, middleware, and so on.

For tape storage, consider implementing the IBM TS7760 family of virtual tape solutions for mainframes in a grid configuration. The grid communication facility provides access to any host for all the data in the grid. As a result, you ensure superior business continuance. Its storage hierarchy is fully integrated and tiered, so you take advantage of both disk and tape technologies. Your system can deliver performance for active data and best economics for inactive and archive data.

To achieve the highest levels of availability, consider implementing IBM GDPS Product offerings. See Chapter 3., “IBM Z resiliency models” on page 93 for related best practices.

4.5.3 GDPS removes SPOF of people

In an average data center immediately following a basic system failure, all the phones are ringing. Every manager within reach moves in to find out when service will be restored. The support staff is frantically scrambling for procedures that are more than likely out of date. And the System Programmers are all vying with the operators for control of the consoles. In short, chaos!

The goal of High Availability is to identify and remove SPOFs of hardware and software. Ironically, the support staff itself becomes a SPOF!

As it turns out, automation not only helps to remove support staff as a SPOF. Automation can also speed up recovery. Automation of system management functions can help meet required service levels, contain costs, and make effective use of resources by simplifying the management of system operations.

Now, imagine a scenario where the only manual intervention is to confirm that one can proceed. From that point on, the system recovers itself by using well-tested procedures. It responds to messages at system speed. You don’t need to worry about your support team using out-of-date written procedures. The team can concentrate on handing calls and queries from the assembled managers. And the Systems Programmers can concentrate on pinpointing the cause of the outage, rather than trying to get everything running again.

And all of this complexity results from a system outage alone. Consider a disaster recovery situation. In recovery, you also need to handle a plethora of tasks: invoke Capacity Back-Up (CBU), remove failed systems from the Parallel Sysplex, switch disk to use secondaries, reverse the remote copy, clean up CF structures and switch policies, modify activation profiles to come up on the second site that uses the correct IPL volume, switch network resources, IPL failed systems, quiesce discretionary LPARs, and so on.

Training staff takes time. People come and go. You cannot be sure that the staff that took part in the last disaster recovery test will be on hand to drive recovery from this real disaster. In fact, depending on the nature of the disaster, your skilled staff might not even be available to drive the recovery. The use of automation removes these concerns as potential pitfalls to your successful and rapid recovery.

Automation to manage system and site events are provided by the GDPS suite of products.

Automation means customizing and using the system's capabilities to perform automatically some of the tasks that operators perform at a console. This frees the operators to handle the tasks that a computer cannot. Automation seeks to correct problems as close as possible to the source of a problem. It also offers improved service, simplifies growth, and increases productivity.

Automating operations ensures that messages receive an immediate response. All operations staff is involved during automation planning to help develop standard procedures for specific situations. In this way, the automated procedures are built by using the experience and knowledge of many experts.

For more information about GDPS offerings, see *IBM GDPS: An Introduction to Concepts and Capabilities*, SG24-6374.

4.5.4 A word about cryptographic keys

Having an encrypted backup is of little or no use unless you have the keys to decrypt them during a disaster. Your organization might use a hot-site (mirrored DASD that uses the GDPS solution). If so, you must ensure that the DASD volumes on which CKDS and PKDS data sets reside are part of the backup (or mirrored). You must also ensure that master key changes that are done at the primary (or production) site are also done at the DR site.

Consider splitting master key into multiple parts (typically two or three). Every part should have an owner and no owner can know more than one part of the key. This practice brings additional security, because it won't be possible for one person to know the complete master key or to change the master key.

Consider a scenario in which your organization does not use a hot site, and instead relies on encrypted tape backup only. In this case, you must ensure that the driver system at least has access to those keys that can decrypt the backup, so that it can restore the system data. After the DR system is up, application data can be decrypted by using the keys that are stored on the system volumes.

For more information, see the *Crypto And Disaster Recovery* presentation:
https://ibm.biz/Bdzyb6



SG24-8446-00

ISBN 0738457795

Printed in U.S.A.

Get connected



