Infor Sizing Whitepaper

Sizing Documentation

Publication Information

Release: Product Name Release level

Publication date: November 15, 2012

Document code: B0052C

Contents

About this guide **11**

Intended audience 11

Related documents 11

Contacting Infor 11

Chapter 1 Introduction **13**

Introduction 13

Audience 13

Document scope 14

Feedback on Sizing Whitepaper 14

Prerequisites 14

References 14

Document organization 15

Chapter 2 Sizing General background information **17**

Sizing definition 17

Sizing scope 17

Importance of sizing 18

Who does sizing 18

Sizing in relation to performance 19

When is a sizing carried out 19

Sizing methodology to use 20

Sizing result 21

Workload description 21

Hardware infrastructure description 22

CPU 22

Memory 22

Disk 22

Network 23

Chapter 3 Various sizing methodologies **25**

Client specific benchmark 25
Advantages 26
Disadvantages 26
Stress test 27
Advantages 27
Disadvantages 28
Sizing according to the sizing guide 28
Advantages 28
Disadvantages 28
Sizing based on single-user measurements and extrapolation 29
Advantages 29
Disadvantages 29
Sizing of upgrades 29
Advantages 30
Disadvantages 30
Overview: advantages and disadvantages of various sizing methodologies 31
Capacity planning 31
Measurements 33
Importance of capacity planning and timing 33

Chapter 4 Sizing process 35
Process, reasons to carry out a sizing 35
Reasons to carry out a sizing 37
Main steps during a sizing 42
Sizing project group 42
Estimate impact of change and determine risk 42
Select sizing method 44
Investigate workload and specific requirements 44
Measurements/benchmarks 44
Carrying out a sizing 44
Sizing evaluation 44
Future sizings/Capacity planning 45

Chapter 5 Infor Sizing model 47
Sizing model 47
Infor Loadfactor Model 48
BRUs, Benchmark Reference Users 49
Sizing example 50
User-based versus transaction-based model 53

Chapter 6 Sizing recommendations, definitions, and attention points ... 55
Various user types ... 55
Named users/Potential users ... 55
Connected users/Logged on users ... 55
Concurrent users/Active users ... 56
Ratio between different types of user ... 56
Impact of ratios on the sizing ... 57
Ratio based on experience ... 57
Various Infor software products ... 58
Non-Infor Software ... 58
Integration ... 58
Customizations ... 59
Multi-byte/Unicode ... 59
Sizing peak periods ... 60
Peak load margin ... 62
Growth ... 62
Margin ... 63
Central versus decentralized solution ... 63
Advantages of central solution ... 65
Disadvantages of central solution ... 65
Batch window ... 66
Backup window ... 67

Chapter 7 CPU sizing ... 69

Chapter 8 Internal memory sizing ... 71
Memory sizing general ... 72
Distribution of resources ... 72
Buffer cache ... 73
Paging and swapping ... 73
Memory formulas ... 75

Chapter 9 Disk sizing ... 77
Disk sizing general ... 77
Disk space ... 77
History ... 78
Archiving ... 78
Disk I/O Throughput ... 79
Spindle size ... 79
Number of disk spindles ... 79

Disk controller 79
Disk cache 80
Raid 80
RAID 0 81
RAID 1 81
RAID 5 82
RAID 0/1, RAID 01, RAID 0+1 83
RAID 1/0, RAID 10, RAID 1+0 83
Software RAID 84
Hardware RAID 84
SAN/NAS 84
Definitions 84
SAN 84
NAS 84
Advantages and Disadvantages 84
Speed 84
Multi-host file access 85
Complexity 85
Backup 85
Image backup 85
Disk sizing formulas 85

Chapter 10 Network sizing 87
Introduction 87
Network capacity sizing 87
Network bandwidth 87
Example 87
Network latency 88
Sizing formulas 89
Network sizing considerations 89
Network compression 89
Citrix/Microsoft Terminal server 90

Chapter 11 Background information necessary for sizing 91
Computer architecture background 91
Symmetric Multiprocessing (SMP) 91
NUMA (Non Uniform Memory Access) 92
Parallel architectures 93
Caching 94

CPU architectures ... 96
Multi core ... 96
Hyper threading ... 97
IBM iSeries characteristics ... 97
CPU Clock speed ... 98
Clock speed same CPU family ... 99
Clock speed comparison between different CPUs ... 99
System scaling ... 99
Server scaling in relation to sizing ... 100
Computer architecture ... 100
Sizing impact on the various applications ... 101
Infor LN ... 101
Operating system ... 101
IO and network resources ... 102
Database/Data setup ... 102
Single threaded versus multithreaded software ... 102
Two-tier versus three-tier ... 103
Definition of two and three-tier within Infor LN ... 103
Selection considerations ... 103
Hardware costs ... 103
Database license costs ... 104
Heterogeneous environments ... 104
High-availability cluster ... 105
Scalability ... 107
Flexibility ... 108
Tuning ... 108
Hardware re-use ... 108
Infrastructure architecture vision ... 108
Performance characteristics of two-tier and three-tier ... 108
Advantages and disadvantages ... 109
Important configuration issues for Infor LN ... 110
Master Application Server ... 110
Recommendations ... 111
Hardware vendor selection ... 112
Price/Reliability of hardware ... 112
Service ... 112
Relation /Trust ... 112
Knowledge/Experience ... 112

Support 113
Knowledge 113
Operating system selection 113
Hardware dependency 113
Scalability 113
Open source 113
Other 113
Database vendor selection 114
Performance 114
Price/Performance 114
License costs 114
Manageability/Ease of use 114
Other 114
Response time behavior 114
Response time definition 115
Response times OLTP 115
Response times Batch 117
Batches on multi CPU machines 118
Heavy versus light transactions 119
Number of CPUs 121
Response times in a multi-tier environment 121
Infor LN: Single bshell versus parallel bshell performance 122
Duration 123

Chapter 12 Sizing Tools 125
Sizing Guides general 125
Sizing Whitepaper 126
Sizing Portal 126
Sizing account 126
Infor LN 127
Sizing guide 127
Sizing Assistant 127
E-Sizing Guide 127
Sizing Questionnaire 128
Infor LX 128
Sizing guide 128
IBM Workload Estimator 128
Questionnaire 128

Chapter 13 Sizing checklist 129
Sizing process checklist 129
Sizing content checklist for new sizings Infor LN 130

Appendix A Definitions, acronyms, and abbreviations .... Error! Bookmark not defined.

Appendix B Use case 1: Consolidation 139
Introduction 139
Sizing approach 139

Appendix C Use case 2: Sizing based on single-user measurements and extrapolation ... 143
Introduction 143
Sizing approach 143
Carrying out a sizing 144
Investigate the various user profiles 144
Investigate how the system is used during the day 144
Test database 145
Key users carry out application functionality and measurements 145
CPU 145

About this guide

The Sizing Whitepaper focuses on sizing details and the sizing process, which is not covered in the sizing guides. This Sizing Whitepaper has necessary sizing knowledge to improve the quality of the sizing.

The only significant change with the previous version is the update of the document template according the new standards.

Intended audience

This document is intended for technical consultants of Infor, partners and customers who are responsible for sizing and implementing Infor software.

Related documents

You can find the documents in the product documentation section of the Infor Xtreme Support portal, as described in "Contacting Infor" on page 11.

Contacting Infor

If you have questions about Infor products, go to the Infor Xtreme Support portal at www.infor.com/inforxtreme.

If we update this document after the product release, we will post the new version on this Web site. We recommend that you check this Web site periodically for updated documentation.

If you have comments about Infor documentation, contact documentation@infor.com.

Chapter 1 Introduction

1

Introduction

Hardware sizing is a critical activity during the implementation and further lifecycle of enterprise applications. The size of the hardware infrastructure has an impact on the productivity and satisfaction of end users and the timeliness of business activities and processes. Therefore, it is important that sizing is carried out correctly. Sizing can be a complex activity; a lot of background knowledge is required in different areas such as hardware, hardware architectures, operating systems, networks, databases, and application architectures.

Necessary background information is provided in the sizing guides. However, the information published in the sizing guides is mainly focused on the specific application characteristics and resource consumption of the specific application. Crucial sizing related subjects such as high availability, operating system selection, architectural choices, approach and many other important choices are not covered in the sizing guides. PBC, Infor Performance and Benchmark Center has much experience with sizing and guiding technical consultants in the area of sizing. PBC noticed that the same questions are asked, and that people run into the same problems. Many areas have been identified as lacking important fundamental knowledge. PBC decided to write a Sizing Whitepaper to close this knowledge gap. This Sizing Whitepaper has necessary sizing knowledge to improve the quality of the sizing, and to decrease time PBC spends on sizing support.

The Sizing Whitepaper focuses on sizing details and the sizing process, which is not covered in the sizing guides. Sizing is not a one-time activity, and must be carried out several times during the lifetime of a solution. The hardware architecture must be adjusted regularly to the dynamics of the enterprise. This is described in the sizing process. PBC noticed that this vision on sizing is not clear to everybody. It is important to make this vision clear and explain the sizing process because it can prevent customers from many performance problems and increase customer satisfaction.

This first version of the Infor Sizing Whitepaper is a first step. Future versions will cover more functionality and will be improved based on feedback.

Audience

The target audiences of this document are the functional and technical sizing consultants who are involved in sizing, and the involved project leaders and management. For the latter, it is not necessary to read the entire document. For the document overview, see section ‘Document organization’, Chapter 1. Based on the overview, you can select specific chapters and go into details, or limit yourself to the high-level chapters which describe the sizing process part.

Document scope

The Sizing Whitepaper only covers sizing related topics. The topics are described with enough detail for you to make the most important choices during the sizing process. This document is a supplement to the sizing guides, but does not replace them. Generic topics such as sizing process, vendor selection, and high availability are covered, which do not belong to the scope of product sizing guides. Product specific details such as resource consumption figures are not covered in this document.

Feedback on Sizing Whitepaper

This document was written to help you better understand the sizing process, to provide background information on several sizing related topics, and to share experiences. We hope this document fulfills the need. There are new versions of this document planned. We would appreciate your input to improve future versions. If you have questions or remarks, or you want to have certain subjects explained in more detail, please let us know. Send your feedback to sizing@infor.com.

Prerequisites

To understand the generic chapters one to four, basic sizing knowledge is required. For the other chapters, detailed technical background is required. For these chapters you need knowledge of computer architectures, networks, operating systems, databases, and integration technology.

Before reading this document, it is recommended to read the sizing guides first. This Sizing Whitepaper builds on the information in the Sizing Guides and many references are made to the sizing guides.

References

The following references have been used as input for the Sizing Whitepaper. They are referenced in the following text as indicated below.

Reference	Description	Author	Date	Doc #
B0045xx US	Sizing Guide Infor LN			

Document organization

Chapters one to four are important for those involved with sizing. These sections are important for involved management and project leaders and provide a high-level overview and explain the importance of sizing.

Chapters 5 to 13 contain more detailed information.

Document reading Guide	
Section number	**Content**
Audience: All sizing project members including project management.	
1	Overview information.
2	Generic sizing background information, such as definition, importance of sizing, who carries out a sizing, and sizing methodology selection.
3	Various sizing methodologies explained with advantages and disadvantages.
4	Sizing process is explained. Sizing is explained as a continuous process during all phases of the lifecycle of an implementation.
Audience: Technical people	
5	Infor has developed its own unique sizing model. This chapter explains the sizing model.
6	Various sizing issues are explained, such as different user types, integrations, and third-party software. All subjects mentioned in this chapter can have influence on the size of a configuration.
7	CPU sizing explained.
8	Internal memory sizing explained.
9	Disk sizing explained, both size and disk throughput.
10	Network sizing explained.
11	To carry out a proper sizing a lot of background knowledge is required. The chapter provides background information on important subjects such as computer architectures, system scaling and response times curves.
12	Various sizing tools and documentation are explained.
Audience: All sizing project members and project management.	
13	Checklist to check whether all steps are completed in

	the sizing process.
Appendix A	Definitions, acronyms, and abbreviations
Appendix B	Use case1: Server consolidation.
Appendix C	Use case 2: Sizing based on single-user measurements and extrapolation.

Chapter 2 Sizing General background information

2

This chapter provides a high-level overview on sizing and the background information to understand the subject sizing better. The target audience of this chapter is people who deal with sizing but do not necessarily need to know all the details, such as management and project leaders.

Sizing definition

Sizing is the process to predict future system workload and finding a suitable hardware infrastructure which can support this workload. During a sizing, the business requirements are investigated and translated into hardware infrastructure requirements. The quality of a sizing depends of the accuracy of the input and the time spent on the sizing. Sizing looks like an exact science, and till certain extend its, but one has to realize that during a sizing many assumptions are made.

There are various sizing methodologies. A sizing can be based either completely on a theoretical approach using sizing guides, on measurements, or on specific customer benchmarks. The selection of the approach is a tradeoff between costs, time, reliability, and risk. The investment in a sizing for a business critical solution is usually higher compared to a low risk application.

Sizing is not a one-time activity, and must be carried out several times during the lifetime of a solution. The hardware architecture must be adjusted to the dynamics of the enterprise. Therefore, sizing is a continuous process. During the lifetime of a solution, there can be many reasons to carry out a sizing.

The estimation of the hardware capacity is the most important step during a sizing, but not the only aspect. You must consider other requirements which have significant influence on the hardware infrastructure size, such as security, high availability, standardization, manageability, and consolidation (for example, to reach a certain availability level, you can decide for multiple servers instead of a single server).

Sizing scope

What should be covered by a sizing depends who you ask. A network analyst for example is mainly focused on the network requirements. At larger customers there are many people involved with their own area of expertise. In this document, a sizing covers all related hardware aspects to build complete hardware infrastructure with the view of a software vendor. Therefore, the infrastructure requirements of each layer will be described high-level. Details such as part numbers and component details are not provided. The sizing should provide sufficient information for the hardware specialists to finalize a sizing and to determine the exact components.

For example, the sizing gives the bandwidth and latency requirements for a network sizing, but nothing about needed routers and network topologies.

The server internal memory requirements are calculated. However, the type of memory, number of Simms, and what the best memory interleaving factor is, is not calculated.

Importance of sizing

The importance of sizing should not be under estimated. During an implementation project there is a strong focus on functionality. But a well-sized hardware infrastructure contributes to the success of the implementation. The size of the hardware infrastructure has an impact on the productivity and satisfaction of end users and the timeliness of business activities and processes. Therefore, sizing must be carried out correctly. Under estimated hardware can cause company bankruptcy.

The importance of a sizing increases when the business becomes more complex. The type of business can also make difference. Usually, high volume environments with a short ‘time to market’-cycle are very critical. In general one has to ask the question: ‘What are the costs, when my solution is down, and when I can’t deliver services to my customer on time’.   If downtime is unacceptable one need to invest more in the hardware infrastructure.

Who does sizing

Carrying out a sizing is team work. During a sizing the business requirements are translated into infrastructure requirements. The sizing definition already shows the multidisciplinary character. People who understand both the business and technical details are very rare. This makes team work necessary. In larger enterprises, responsibilities are distributed across multiple people. The following disciplines should be involved:

- Network infrastructure
- Hardware and operating system
- Database
- Application management
- Application customization development
- Business architects and business operations

A technical consultant usually takes the lead. This can be a consultant of Infor, an implementation partner, or hardware vendor.

The degree of participation varies with the importance of the sizing and the stage of the project. A pre-sales budgetary sizing requires less involvement compared to a final sizing just before a customer goes live.

Sizing in relation to performance

Correct sizing can guarantee optimal performance, but sometimes it cannot guarantee this.

The performance depends on many factors, not only on the hardware infrastructure size, but also on the following:

- Optimal configured network.
- Operating system tuning.
- Database tuning.
- Middleware.
- Technical and functional application configuration.
- Application usage.
- Customization optimization.

If one area is not functioning as it should, the performance suffers, but the sizing can still be ok. Often, you can try to solve performance issues with hardware upgrades. In many cases, tuning will help.

The involved software can contain performance bugs, despite Infor's thorough quality control. Therefore, the software should be optimized and the problem should be solved with additional hardware. In the functional area, people realize the consequences of bugs. In the performance area it is not always realized, and performance issues are frequently not classified as bugs.

For example, a database optimizer can produce the wrong query execution plan and cause full table scans. When this query is frequently used by many users it can impact the overall performance dramatically. In this example, the size of the hardware is too small. Based on the symptoms, this conclusion is correct, but the sizing is still acceptable. You should solve the root cause of the problem (the optimizer issue) rather then resize the configuration and adding additional hardware to the configuration.

A well carried out sizing guarantees optimal performance. If the system is tuned and configured well, the application runs at its best and is used as assumed during the sizing. Sizing information published in the sizing guide assumes these circumstances.

When is a sizing carried out

Sizing is not a one-time activity, and must be carried out several times during the lifetime of a solution. The hardware architecture must be adjusted to the dynamics of the enterprise. Sizing is an ongoing process. There are many reasons to carry out a sizing.

Sizings are frequently carried out for pre-sales reasons at the beginning of an implementation to estimate the hardware budget. The sizing in this situation will be very rough and high-level, because not many details are known. During later stages of the implementation project more detailed information, which has influence on the sizing, becomes available. The customer's business process and how the software will be used become clear. These can have an impact on the size of the hardware infrastructure. Before a customer goes live, it is recommended to re-evaluate the sizing. Later, it is also advised to re-evaluate the size of the hardware infrastructure sizing based on the results of capacity planning requirements analysis.

Reasons for initiating a sizing can be:

- Budgetary pre-sales sizing.
- Sizing before customer goes live.
- Update of hardware.
- Os upgrade.
- Database upgrade.
- Application upgrade.
- Customizations added.
- New application functionality used.
- Business growth.
- More users.
- Higher transaction volumes.
- Change in business processes.
- New integrations with other applications.
- Performance problems.

These reasons are explained in Chapter 4 in more detail.

Sizing methodology to use

The various sizing methodologies which can be followed are described in Chapter 3 in detail.

The best methodology depends on the circumstances, importance of the sizing, and the reason why the sizing is being carried out.

The sizing methodology described in the sizing guides assumes a new customer installation. Based on questions and a sizing model, the customer load is investigated, estimated, and quantified. Based on the outcome of the modeling exercise, suitable hardware is selected.

This is a valid approach, which can also be used for existing customers. But for existing customers there are more possibilities and sizing methodologies available. In the sizing definition it is stated that, in a sizing, the functional business requirements are translated into hardware infrastructure requirements. The definition speaks about the following two steps:

- Business requirements and application usage must be investigated.
- Investigated business requirements and application usage must be translated into hardware infrastructure requirements.

For existing customers, you can use the load characteristic on the current system as input for the sizing. The resource consumption can be monitored for a long period of time. The measurement results provide valuable sizing metrics. You can see how the application load looks like based on the real customer user concurrency and transaction volumes. Together with the information published in the sizing guide, this method results in a very accurate sizing.

Without saying the sizing model published in the sizing guide is not reliable, it does not need explanation that each modeling and translation step loses some accuracy. Using measurements of the current situation minimizes this lose of accuracy.

For example, if a customer wants to upgrade from version A to B of a specific application, and the sizing guide of this application provides a load difference factor between the two versions, it is probably more accurate to measure the current load characteristic of version A on the system and apply the difference factor, rather than going through a complete theoretical sizing exercise.

For a customer who operates in a high volume, low lead time type of business, correct sizing in this type of critical environment is crucial. The cost of system downtime can easily exceed the hardware costs. A sizing using the sizing guide can be a valid approach, but to minimize the risk you can consider a sizing approach based on load simulation or a customer specific benchmark. The exact customer workload will be simulated on the target configuration. When the load simulation meets the expectations, the customer can be confident the configuration can handle the predicted production workload.

Sizing result

The sizing definition explains that sizing is the selection of a suitable hardware infrastructure for the expected workload. The main aim of a sizing is a description of this hardware infrastructure. Together with this description, a description of the workload and the work procedures the sizing is based on should be described. The hardware infrastructure should be used according to this description. When the application is used differently than originally planned, the hardware infrastructure can have insufficient capacity. For example, if the batches are started during the day and were planned to run during the night, severe performance problems can arise.

Workload description

When a sizing is made, you can start to investigate how the application solution will be used and how big the workload is. The detail of this information depends on the chosen sizing method. When a rough sizing method is chosen, this information is not available. The following are important in a workload description:

- Number of named, connected, and concurrent users.
- Used application functionality by the specific users.
- Batch load: number batches, frequency, and scheduled time
- Expected growth and peak load.

Usually, a workload description will correspond with a thoroughly filled in sizing questionnaire.

Hardware infrastructure description

The sizing scope explains that not every detail will be described by Infor. The view on the sizing is the view of a software vendor. Specific hardware details can be better completed by the hardware vendor specialists. A hardware infrastructure description consists of computer systems, disk subsystems, and network layout with all their own capacity requirements.

Usually system requirements can be described using the following four basis components:

- CPU
- Memory
- Disk
- Network

In addition to this vendor-specific information must be added.

CPU

To describe the CPU requirements, you must know the following:

- Number of CPUs in the server
- Type of CPU
- Clock speed
- Cache size(s)

Memory

To describe the memory requirements, the amount of memory needed is sufficient. The memory that a system can be equipped with goes in step, according to the size of memory banks. The hardware specialist will round the amount of required memory up to the value which fits in the machine.

Disk

For disk sizing, two factors are important:

- Disk size
- Number of disk(s) arms

The size of the disks is self-explanatory, but the number of disks needs an explanation. The number of IO-requests a disk can handle per unit of time is limited. When a lot a disk activity is expected, the data has to be spread across multiple disks to deliver the required IO throughput. Often, the number of disks in a configuration needs to be higher for performance reasons than for disk size reasons.

To protect data against disk failures, disks can be configured in various RAID configurations. Hardware specialists can fill this in and also the type and number of disk controllers and various cache sizes.

Network

If the hardware solution consists of multiple systems with various applications, or the various application tiers of an application are spread across multiple systems, network sizing becomes important. This is particularly true when certain network connections are WAN connections.

Network requirement can be described as a function of the following:

- Network bandwidth
- Network latency

For network sizing, it is important to know the minimal bandwidth and the maximum latency requirements.

Sizing of routers and firewalls do not belong to the scope of an Infor sizing, and specialists must work on this.

Chapter 3 Various sizing methodologies

3

This chapter is aimed at people who deal with sizing, but do not necessarily need to know all the details, such as management and project leaders.

There are various sizing methodologies in a sizing, and each methodology has advantages and disadvantages. They differ in accuracy and costs. In this version of the document, the method used in the Infor Sizing Guides is described in detail; the other sizing methodologies are described in high-level.

One sizing methodology does not exclude the other. For example, a quick sizing can be carried out to estimate the hardware requirements needed for a customer specific benchmark and to estimate budget.

Sometimes a stress test is used to verify a paper sizing and create a higher level of confidence. This depends on the customer' situation, the circumstances, and the requirements.

Client specific benchmark

A client specific benchmark simulates the major business functionality used by a customer. The data structure, application parameter settings, transaction volumes, number of users, and all significant driving customer characteristics are simulated in these tests. Therefore, the end-result is very accurate and the implementation risk is minimized with this type of sizing exercise.

The end-user activity will be simulated based on user simulation scripts. There are several tools available to create user simulation tests such as LoadRunner from HP/Mercury, or Rational/IBM. LoadRunner offers integration with Infor LN and Infor Baan IVc and Infor Baan 5.

These load testing tools offer functionality to capture keystrokes and mouse clicks of real end-users. This information can be stored in a script and replayed, so the end-user activity is simulated to include realistic think times and type rates.

The scripts can be replayed to simulate a multi-user environment so that the system will be pushed to its limits. The response times of pre-defined transactions can be measured.

The simulation must only be done for the most frequently used and most performance critical business processes. During client specific benchmarks, all used business functionality can be simulated, but this is not necessary. Functionality which is carried out rarely does not contribute much to the overall system load. Note that from load perspective, light functionality can contribute significantly to the overall load when this functionality is frequently used and carried out.

A client specific benchmark costs a lot of time and is not an easy job. It takes quite an effort to develop simulation scripts and make them robust. Scripts must be parameterized because when a captured script is replayed, the application will probably react with an error: 'data already' exist. The scripts must be adjusted to work on bigger ranges of data. For example, when the script is captured with an activity on a certain item, it is important that during replay not all users use the same item due to contention problems.

The data setup and the scripting must be such that the test can be carried out multiple times and be reproduced.

The average time needed to develop a customer specific benchmark costs three months. However, this depends on the number of scripts which must be built and the business functionality simulated; experience is another factor. You can learn how to build scripts quickly. Carrying out a benchmark, tuning the system, and analyzing and improving is a specialized activity, which requires experience and knowledge.

A customer specific benchmark is quite an investment due to the hours and costs of the benchmark simulation tool. You should consider a customer specific benchmark only in business critical situations where the business does not allow any performance delay and where you are sure the system reacts quickly. The costs of a business delay can often exceed the hardware costs and the costs of the customer specific benchmark.

Customer specific benchmarks are also often used in environments with high transaction volumes. High volume environments are usually very customer specific and not covered with the standard benchmark in the sizing guides. Therefore, it is important to test this specific situation and ensure the system can handle the high transaction volume load within the reserved time.

In this situation, the customer specific benchmark is also a perfect tool to find the optimal configuration settings, setup, and architecture.

Some customers have customized the application heavily to serve their specific needs. Therefore, a customer specific can be very useful to test the customization on scalability and to find possible concurrency problems such as contention and database locking.

Advantages

- The results are very reliable because the simulated environment simulates the real production environment closely.
- Benchmark can be reused to test future changes in the configuration.

Disadvantages

- It is expensive to develop and carry out a customer specific benchmark.
- It takes a long period before sizing information can be delivered.
- A customer specific benchmark is complex to setup and requires a lot of technical experience.

Stress test

A stress test is an organized test where a group of users works on the system and carry out their regular daily activities. During this, the system is monitored and the resource consumption is measured. The measurement results are used as input for a sizing.

A stress test is often carried out at the end of an implementation project before the system goes live. All business procedures are known and it is clear how the application will be used. The application parameters are already set on their final values. The configuration already reflects the final production situation. This is necessary otherwise the sizing will be based on the wrong assumptions.

Ideally, you should carry out the stress test with all the users which will use the application when the system is live. This gives the best simulation of the final load. In practice, it is not feasible to free up the complete workforce for testing, as this conflicts with other activities. Often, not all users are trained with the new application in this stage of the project.

Regularly, a group of key-users is selected to participate in a stress test. Each key-user represents a function area. The mix of user functionality reflects the final production mix. Having a limited group of users results in extrapolating the final number of users load.

During the stress test, the users follow a pre-defined procedure and enter a pre-defined number of transactions. It is important that the users enter data with a realistic working speed, which reflects the final working speed during production. Fast working users cause a load on the system which is too high. The customer is left with a system which is too big when this high load is used as the basis of the estimation. Experience shows that users tend to work too hard during a stress test.

Sometimes, users are asked to work faster then they normally would during the stress test. The difference is that this is done in a controlled way. The fact they work faster is known. The reason why a stress test is carried out this way is to compensate for the lower number of users in the test compared to the final number of users in production.

During a stress test the end-users write down the estimated response times of pre-defined transactions. After the test, the measured response times will be matched with the pre-defined criteria and expectations. The stress test is a perfect method to check whether the solution matches the business requirements.

It is advised to check the application log file, event viewer, or spool files after the test. It is important to know that the test is carried out under clean circumstances.

During the stress test, the system should not be used for activities other than the stress test.

Another reason to let users work very fast is to test the system limits because it shows what happens when incidental higher capacity is needed.

Although a stress test is used to deliver input for a sizing, a stress is also used to verify the configuration based on a paper sizing and to check whether the application can handle the expected system load.

Advantages

- A sizing based on a stress test can be very accurate. Real simulated business activities form the basis of the sizing.

- Relatively low investment compared to a customer specific benchmark. Making a group of people available is a significant investment, but much lower than the costs of an extensive customer specific benchmark project.
- A stress test gives an idea of the response times, which pure theoretical sizing does not provide.
- A stress test provides quick results.

Disadvantages

- A stress is hard to reproduce. It is not feasible to organize a group of people multiple times.
- A stress test is not exactly reproducible. If a certain issue occurs, it is not guaranteed to show up again.
- Costs are higher than a paper sizing.
- Results are less reliable compared to customer specific benchmark. There is an extrapolation factor used if tested with a subset of users.
- Difficult to plan because all people need to be available at the same time.

Sizing according to the sizing guide

The Performance and Benchmarking Center of Infor has developed its own sizing methodology. The method is used in thousands of sizings and has been proven. Feedback from hardware vendors show the Infor sizing methodology is clear, accurate, and easy to use, which is why the methodology is used for various Infor applications. The sizing method is explained in the various sizing guides and in Chapter 5. There is an overlap with the explanation in the sizing guide, but the explanation in this whitepaper provides more background information than the sizing guides.

Advantages

- Sizing using the sizing guide is easy to carry out and can be done on immediately, which makes the sizing cost effective.
- The sizing guide is used in many situations, which means there is a lot of experience in the guide.
- Technical people with average technical background can do the sizing. For other sizing methods, specialized people are needed.

Disadvantages

- A sizing using the sizing guide can be too generic for special situations and business critical environments.
- Every customer is unique. A sizing guide is written for the average market and will not cover every situation.

Sizing based on single-user measurements and extrapolation

Certain customer situations are unique and sizing cannot be completely covered with the generic sizing guide. These situations can be approached with a stress test or a customer specific benchmark, as previously explained. Due to resource constraints such as money, time, and people, it is not always possible to organize a stress test or customer specific benchmark. In this case, a ‘Sizing based on single-user measurements and extrapolation’, which is described in this chapter, is a perfect alternative.

During this test, a particular user simulates their normal system activities. During this the resource consumption, mainly CPU and memory, are measured. This is repeated for all main business procedures and user profiles. The measurement results are used as input for a sizing. The results have to be estimated because they are measured from a single user.

This sizing methodology is worked out in an example/use-case, see Appendix C.

Advantages

- Low cost solution.
- No large group of people needs to be available at a certain time, which is easier to plan. The tests can be carried out on individual basis.
- The sizing can be carried out in a relatively short period of time.

Disadvantages

- Application concurrency not tested.
- The system load as function of the time is difficult to measure.
- IO throughput cannot be estimated based on this test.
- Method is difficult to use for multi-threaded applications.
- Experienced people who understand the system characteristics are needed to make the correct estimations.

Sizing of upgrades

Sometimes, existing customers must upgrade some components of their IT-infrastructure which require a sizing.

Examples of upgrade reasons are as follows:

- New server hardware.
- New version of operating system, database, or application.

When the sizing for upgrades is started from the beginning, the customer workload must be investigated. This is a time-consuming activity and it is easy to make mistakes. If the customer is running optimal without any complaints and the expected impact of the change is minor, it is more accurate to measure the current work load characteristic on the system and use this information as input for the sizing. Based on the load characteristics, you can see how many users are active at one time, and when the peaks occur during the day. This detailed information can be lost when a sizing is approached theoretically from scratch using the sizing guide and sizing questionnaire.

The sizing guides provide load difference factors for the most common upgrade paths. If a certain factor is not listed, this factor can sometimes be determined by comparing different version of the Sizing Guides. For example, an upgrade from an old version to a new version.

To see how an upgrade sizing works, see Figure 3-. On top of the measured load characteristic such as CPU, Memory, disk and Network, factors must be applied.

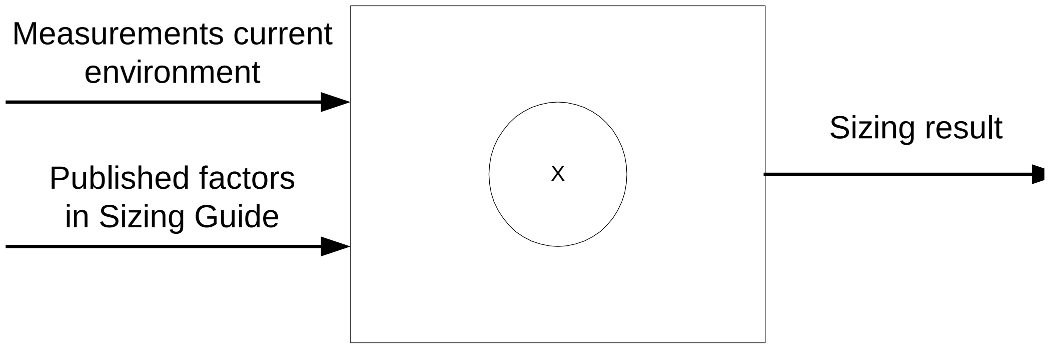
Measurements current environment
Published factors in Sizing Guide
X
Sizing result

Figure 3-1 Upgrade sizing

This type of sizing is chosen when the customer is up and running and satisfied with the current performance. If the customer has performance problems, it is not advised to the use this method, otherwise the load caused by the problems is used as input for the extrapolation. It is advised to use this method only for minor changes in the infrastructure. For large changes, no factors are available in the sizing guide.

The result of a sizing based on this approach is usually more accurate than a sizing based completely on the sizing guide. This does not mean that the sizing guide is accurate, but the customer load description is exactly known by the measurements.

Advantages

- Sizing results are very accurate.
- Many customers measure their environment on a regular basis and have historical performance data available. For these customers, a sizing can be carried out quickly.

Disadvantages

- If customers do not measure on a regular basis, they must start measurements.
- This method requires the knowledge to measure, to interpret, and extrapolate the results. The sizing must be done by a specialist.

Overview: advantages and disadvantages of various sizing methodologies

This section provides an overview table with the advantages and disadvantages of the various sizing methodologies. It can be useful to select the appropriate method.

Overview: Advantages and disadvantage of various sizing methodologies					
	Customer specific benchmark	**Stress test**	**Sizing based on sizing guide**	**Sizing based on single-user measurements**	**Upgrade sizing**
Time to carry out sizing	--	+	++	+	++ 1)
Costs	--	-	++	+	+
Accuracy	++	++	+	+/-	++
Easy to plan	--	--	++	+	+
Transaction Response time information	++	+	+/-	--	--

Table 3-1 various sizing methodologies compared

1) If measurement information already available.

Capacity planning

The sizing methods described before in this chapter are mainly focused on new installations and significant changes to the environment. The methods predict on forehand what the load on the systems will be, to make sure that there is sufficient capacity to run the initial workload after the change and the load in the near future.

In a live environment the resource consumption will grow over time. This can be caused by many reasons, some examples:

- Database growth, it cost more resource to insert, update and to search data.
- Database gets fragmented
- Users get used to the application and work faster. Of course this will not contribute to the total resource consumption, but the load per unit of time can increase.
- Users start to use more application functionality

- Business growth
- Installed OS and database patches
- Application service packs
- New integrations

The sizing methods described earlier in this chapter do not anticipate this. It is advised to use Capacity planning. Capacity planning predicts the future capacity requirements. The estimation is based on the current workload and workload in the past on the systems. The time will be predicted when the configuration runs against its limits. It is important to anticipate and ensure that before this point is reached, the right actions are taken. Actions can be to reorganize database or disks, tune operating system or database, tune application, hardware upgrade, new hardware, or change of business procedures.

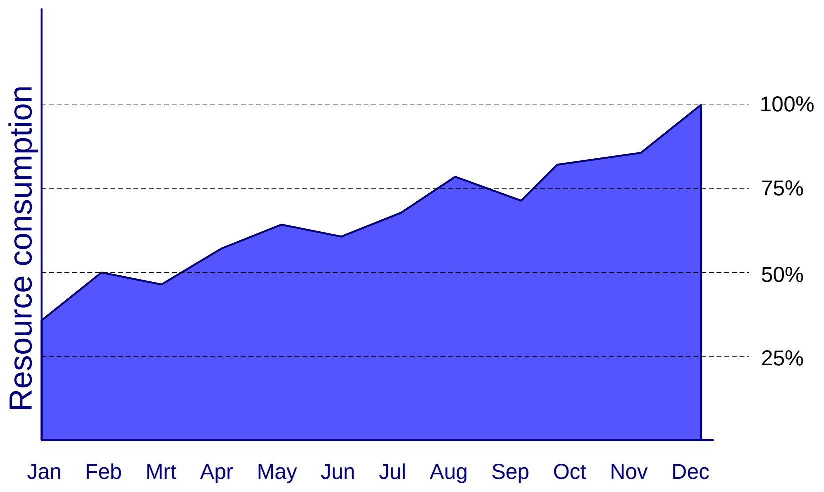

Figure 3-2 Capacity planning –Gradual resource consumption increase

Figure 3-2 shows the resource growth increase of one year. You can see that the configuration hits its limit in December. The responsible system managers should have known this beforehand. When a trend line is drawn, it is clear that there is a gradual need for more resources over time, because every month the resource consumption is increased. The growing trend in the first half of the year is almost the same as in the second half of the year. Therefore, in summer time, you can predict that the system reaches its maximum in December. This gives you six months to take action.

The graph shows a gradual load increase which is probably caused by data growth and unbalance of the database. Various trend lines are possible. Figure 3-3 shows a longer term trend line. There are various jumps in the load. After the Operating system was upgraded, it should have been noticed that the system was almost saturated, and the right action should have been taken.

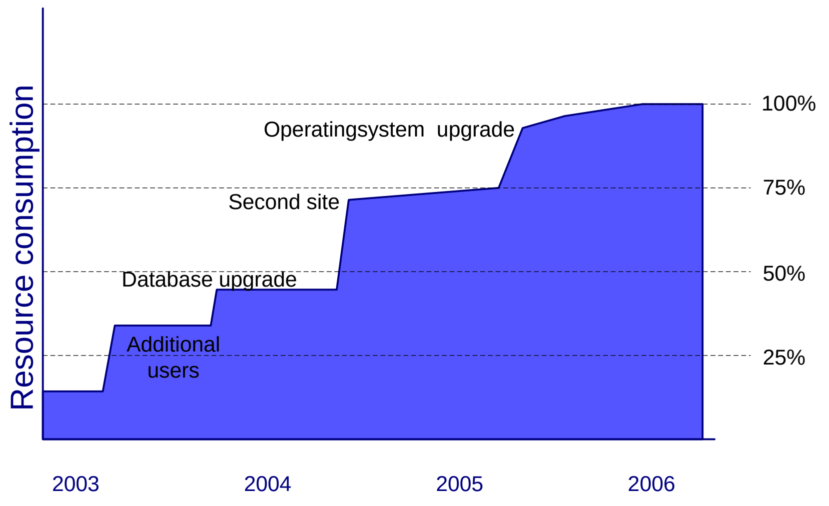

Figure 3-3 Capacity planning – Resource consumption increase based on occasions

Measurements

The above example shows that measurements are important for capacity planning. Without the right measurement data, it is not possible to determine trends.

It is important to measure the right and sufficient metrics, such as:

- CPU utilization
- Memory utilization
- Disk utilization
- Network utilization
- Response times
- Usage of licenses
- Batch duration
- Backup duration

Ensure that not too much detailed data is collected. It is hard to analyze large amounts of data, and when many metrics are collected you could loose the overview.

To determine trends, historical data should be preserved.

Importance of capacity planning and timing

Capacity planning is very important in business critical environments to ensure the hardware infrastructure keeps functioning with acceptable performance levels. Capacity planning ensures there is always sufficient capacity. However, it takes time to take the right actions before the system hits the limits. If new hardware is needed, there can be long delivery times, and short term decisions must be made. Otherwise, the installation of hardware cannot be scheduled at the best time, which can significantly impact the business operations.

Chapter 4 Sizing process

4

This chapter is aimed at people who deal with sizing but do not need to know all the details, such as management and project leaders. It also provides information about the project aspects of sizing

Sizing is not a one time activity. It must be done on a regular basis during the lifecycle of an implementation. Every time when a component changes significantly the impact on the technical infrastructure has to be evaluated. This chapter provides an overview of the events which can lead to a sizing. Every time when a significant change is made in the infrastructure you should decide whether a sizing is needed.

Process, reasons to carry out a sizing

There are many reasons to carry out a sizing. For an overview of these reasons, see the following Figure 4-1. All the reasons stated are explained in detail in the next paragraph.

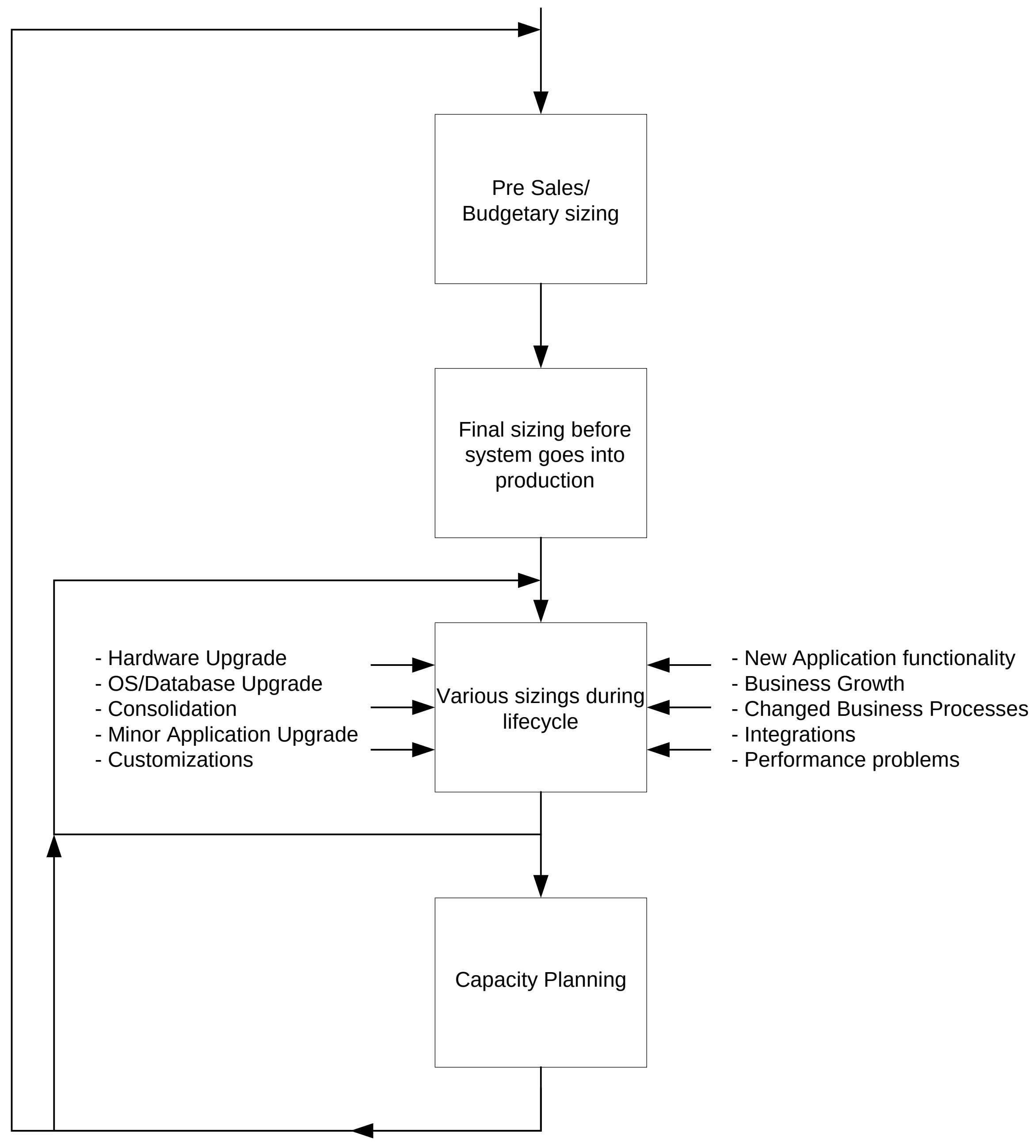
Pre Sales/ Budgetary sizing
Final sizing before system goes into production
Various sizings during lifecycle
- Hardware Upgrade
- OS/Database Upgrade
- Consolidation
- Minor Application Upgrade
- Customizations
- New Application functionality
- Business Growth
- Changed Business Processes
- Integrations
- Performance problems
Capacity Planning

Figure 4-1 Reasons to carry out a sizing

Reasons to carry out a sizing

The following are reasons to carry out a sizing:

- Budgetary or pre-sales sizing

  Although hardware costs have declined, they still form a significant part of the implementation costs. Before an implementation starts, customers like to have an indication of the expected hardware costs for budgetary reasons.

  When the application is supported on multiple platforms, this sizing is sometimes used to compare platforms and databases to determine the most cost attractive solution. The pre-sales sizing is also frequently used to make early decisions for infrastructure choice. For example, the OS and database will be selected, but also important choices will be made such as two-tier or three-tier, and central or decentralized.

  The budgetary or pre-sales sizing cannot be used as the final sizing because the assumptions the sizing is based on change during the implementation phase.

- Sizing before customer goes live

  It is important to ensure the system has sufficient capacity before the customer goes into production with the solution. This is the most important verification point within the implementation project plan. If the capacity in not sufficient, the implementation can fail.

  It is a risk to go live based only on a presales sizing. During the implementation, many things change which can influence the sizing. In this case, the sizing should be revised.

  The sizing must be carried out within such a time frame that there is time left to order new hardware if the pre-sales sizing does not match with the actual situation. You must ensure that no changes which have a significant impact on capacity requirements after the final sizing are made; otherwise, the sizing has to be re-evaluated.

  During final sizing you should look to capacity, but also important issues such as high-availability, security, backup and archiving. These factors can also have significant impact on the final hardware infrastructure.

- Hardware update

  Depending of the quality, hardware can be up and running for several years. Running the same hardware for a long period of time means the original hardware purchase cost can be spread across many years, which can lower the TCO of the implementation. However, running many years on the same hardware involves a risk. After some years the MTMF, Mean Time between Failure decreases, and the chance that hardware components start to fail increases. If the hardware infrastructure serves a business critical application, long downtimes are not acceptable. It is advised to replace the hardware after a couple of years. Hardware vendors deliver service contracts to deliver support on the infrastructure. The price of the contracts depends of the level of support delivered and the minimal response time in case of emergency. Usually, the prices of all contracts increase after the first years because the hardware reliability decreases over time. From the cost perspective it can be beneficial to replace the hardware with new hardware and close new support contracts.Other reasons to replace hardware can be due to insufficient capacity, or the hardware is no longer supported by Infor or vendors of other applications. Infor strongly advises you to work with a production environment with only supported components. It does not need explaining that unsupported components cause a high risk for the business.

When the hardware must be replaced, it is advised to carry out the sizing again. You must ensure the capacity requirements are not changed over time. Typically, the most suitable sizing method is sizing for upgrades, in section 'Sizing of upgrades', Chapter 3.

The capacity of the CPUs increase over time. New servers deliver the same or more capacity with less CPUs. It is strongly advised not to look only to the overall capacity. To deal efficiently with batch performance, sometimes a minimal amount of CPUs is advised, otherwise the OLTP users can be hampered by the batch jobs (see section 'Response time behavior', Chapter 11). For the same reasons, it is important to look to the capacity of individual cores of hardware threads when CPUs consist of multiple cores and hardware/hyper threads (see section 'CPU architectures', Chapter 11).

- OS upgrade

  The hardware/OS vendors release new versions of the operating system on regular basis. The new releases deliver new functionality, and include fixes. For support reasons, it is recommended to work only with supported versions of the operating system. An upgrade of the operating system can be a trigger to initiate a sizing. In practice, the load difference between operating system versions are not as big as they once were; but this in not a given. You should at least discuss the impact with the vendor. An example of a significant OS upgrade is an upgrade from 32 bits to a 64 bits operating system.

  In general, the most suitable sizing method in this situation is mentioned in section 'Sizing of upgrades', Chapter 3.

- Database upgrade

  The database is an important component within an infrastructure. The database claims a significant part of the overall available system recourses and is a significant factor for the overall scalability.

  From a supportability point of view, Infor advises you to work only with supported database versions. Therefore, it is important to follow new database releases.

  The difference in resource consumption between the various database versions can be significant. It is strongly advised to plan a sizing when a database upgrade is considered.

  New database versions deliver new or improved functionality. When this new functionality is used, you should evaluate the impact on the sizing. Certain functionality can have significant impact. For example, if it is decided to use the grid computing with the new database version, the performance characteristics change completely.

  New databases deliver a higher degree of auto tuning, which has many advantages, but can have a negative impact on the overall system load.

  Usually, the most suitable sizing method in this situation is mentioned in section 'Sizing of upgrades', Chapter 3.

- Application upgrade

Application upgrades can be minor upgrades to a new release, or major upgrades to a completely new generation of the software. Usually, new releases of the same version only include fixes, and these fixes do not have a negative impact on the overall load requirements. Sometimes, fixes can have a positive effect on the overall system requirements. The functionality of a sub version can also change. For example, business rules, tax rules, and regulations can change. Usually, the impact of these changes is minor, unless complete new business functionality is added. Typically, the sizing only has to be re-evaluated when new functionality is added.

However, an upgrade to a new generation of the software is different. This is not done frequently, and in most cases there are many years in between. Sometimes, versions are skipped and the latest and greatest version is selected by the customer. There can be a big gap between the current version and the new generation of the application. Previous manual tasks can be automated, and business procedures can be changed. A major upgrade is a trigger to re-evaluate the complete way of working. An upgrade to a new major application version is an implementation project in itself.

A sizing is necessary during major application upgrades, and not doing this is too much of a risk. There is no preferred method to follow. Based on the expected impact of the upgrade, you must select a method that minimizes the risk.

- Customizations added

  The Infor software is developed to serve a generic market. Specific application functionally, which is only important for a group of customers, can be brought about by certain parameters. There can arise a situation whereby the customer requires certain functionality not covered by the standard. This functionality can be so specific that it cannot be added to the standard application. In these situations, customers will customize the application.

  Sizing of customization can never be covered by a generic sizing guide. The implementation partner or hardware vendor must handle this. Depending on the modification and the expected transaction volumes or the number of users using the modifications, the impact can be significant. This can be a trigger to initiate a sizing.

  All sizing methods, except for the one based purely on the sizing guide, can be used for the sizing of the customizations. Which one to select is based on the risk level which is acceptable and the investment you want to make.

- New application functionality used

  The Infor applications have broad functionality, and not all customers start using the functionality from the beginning. The customer organization and users should be ready for this. Some customers gradually adapt new functionality to keep the implantation project manageable, a controlled versus an approach which changes everything at ones.

  Using new functionality can have an impact on the overall capacity requirements. Using new functionality is a trigger to initiate a sizing, but the importance depends of the expected impact. All sizing methods published in Chapter 2 can be used. The choice depends on the risk level which is acceptable and the investment you want to make.

- Business growth

The customers' business can be very dynamic and grow by acquisition or by organic growth. There can also be negative growth. Negative or positive growth impacts the overall capacity requirements. If the growth factor is significant, you should consider a sizing. Even if the growth is negative, it can be beneficial to re-evaluate the hardware requirements. Smaller hardware with a new support contract can lead to cost reduction.

Another related situation which does not belong to growth is outsourcing and off-shoring, which leads to less activity on the internal system. Significant changes in this area can make a sizing necessary.

Business growth can lead to the following:

- More users using the system.
- Higher transaction volumes.

As a result, the system resource demand increases. Business growth can make it necessary to reevaluate batch performance, batch and backup window.

- Changes in business processes

  The business circumstances and economic situation of the customer are very dynamic. To stay in business, the customer must constantly adapt their enterprise based on these dynamics. This has consequences on the internal processes and the way the system is used such as business using the internet versus direct business, project-based instead of flow production, and consolidated finance structure instead of decentralization financial organized structure.

  Changing business processes can impact the overall capacity requirements, which can be a trigger to carry out a sizing. The importance depends of the expected impact. All sizing methods published in section Chapter 3 can be used. The choice depends of the risk level which is acceptable and the investment you want to make.

- New integrations with other applications

  Integration is an important subject because applications within a company are more incorporated than in the past. This creates a higher level of efficiency and can save costs. Specialized applications to fulfill certain tasks, such as planning, become more common. These applications work together with the application backbone.

  More tasks will be automated and new technologies used, such as E-business, bar-coding, RFID, planning, and so on. These new technologies put a higher demand on integration.

  There is a trend that applications of business partners are connected. This leads to more data exchanges and can completely modify the load characteristic.

  Through the integrations, data will read, written, or complete databases will be transferred. This has impact on resource requirements and a sizing can be necessary.

- Performance problems

  Performance problems can make a sizing necessary. Infor advises you to be careful, and not to immediately start to solve performance problems with new hardware; first, it is important to investigate the problem. Often, problems can be solved by specific tuning, such as hardware, operating system, database or application tuning. When the system is not tuned or configured properly, new hardware will not solve the performance problems. A lot of money is invested without solving the problem.

Analog to functional bugs, there can be performance bugs. This is explained in ‘Sizing in relation to performance’, Chapter 2. There can be performance bugs in the standard software and also in the customizations. Many performance bugs arise when the application is used differently than expected. For example, a certain view on the data can be important for the customer, but is not expected by the programmer. The programmer has not created database indexes for good reasons on this combination. This is an example of a typical situation which must be solved by consulting or support channel.

The root cause of performance problems is not always related to problems with the software. For example, the system can run out of resources while the application load characteristic is normal, an earlier sizing could have went wrong, or the circumstances were changed without evaluating the impact on the hardware infrastructure. This is the situation where performance problems must be solved with a hardware upgrade.

There are situations where no thorough analysis is made, such as in the following figure when it was believed it was cheaper and easier to solve the performance problems with new hardware than to analyze the root cause. Figure 4- is the result of what happened:

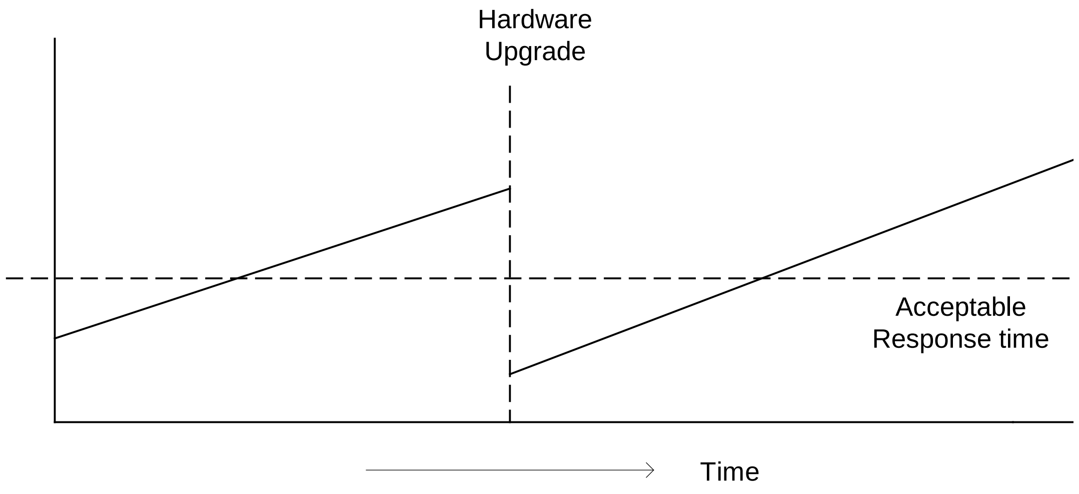

Figure 4-2 Computer Upgrade in combination with non-linear performance problem

There was a non-linear performance problem. The response times deteriorate over time. After the hardware replacement, the problem went away because the faster hardware can carry out the software faster. The problem came back after a while. The software still has the non-linear characteristic; this will not change on the faster hardware. In this case, the hardware upgrade was not the right solution. The bottleneck is moved, but not removed.

- Consolidation

  The network capacity of WANs is increased over time and network costs declined. This trend created the possibly to work more centralized. Users connect using a WAN connection remote to the application. Working centralized has many advantages from a management point of view because it decreases complexity and is easier to manage.

The size of servers grew over time. In the past it was often necessary to distribute the application across multiple servers and to work multi-site by separating the business into logical units. With the current hardware capacity, the size of the server is hardly an issue any more.

The increased network capacity and larger servers create possibilities for consolidation. A hardware consolidation can be a trigger for a hardware sizing.

- Capacity planning:

  The resource consumption on the systems can change over time. Infor advises you to monitor the system on a regular basis and store the measurement data. Compare the current measurement results with previous measurements and determine what the resource consumption curve looks like. If there is a growing trend, you can predict for how long sufficient resources are available and when the saturation point is reached. If a lack of resources is expected in the short-term, you should consider a sizing and extent or replace the hardware.

  The capacity requirements grow over time for many reasons; one being that the database grows overtime. This has an impact on the resource consumption.

  Capacity planning is explained in more detail in ‘Capacity planning’, in Chapter 3.

Main steps during a sizing

Figure 4-3 gives an overview of the main steps to complete when a sizing will be carried out. See also the sizing checklists in Chapter 13.

Sizing project group

Carrying out a sizing is team work. For more background information, see section ‘Who does sizing’, in Chapter 2.

One of the first steps is to establish a sizing project group. It could be that the first conclusion of the project group is not to carry out a sizing. Therefore, there seems little point in putting this project group together. However, a well considered decision can save costs. The decision is made by various people and viewed from various angles. In any case, Infor advices you to put together a project group.

Estimate impact of change and determine risk

Every time a configuration change is proposed or planned, a risk analysis must be made, and you must decide if a sizing is necessary. Section ‘Reasons to carry out a sizing’, in Chapter 4, lists various reasons to consider a sizing. In a production environment you should know the average system utilization, see section ‘Capacity planning’, in Chapter 3. The impact of the change should be estimated and you must determine if there is sufficient head room on the system to implement the change.

In business critical environments, you are advised to start a sizing exercise anyway.

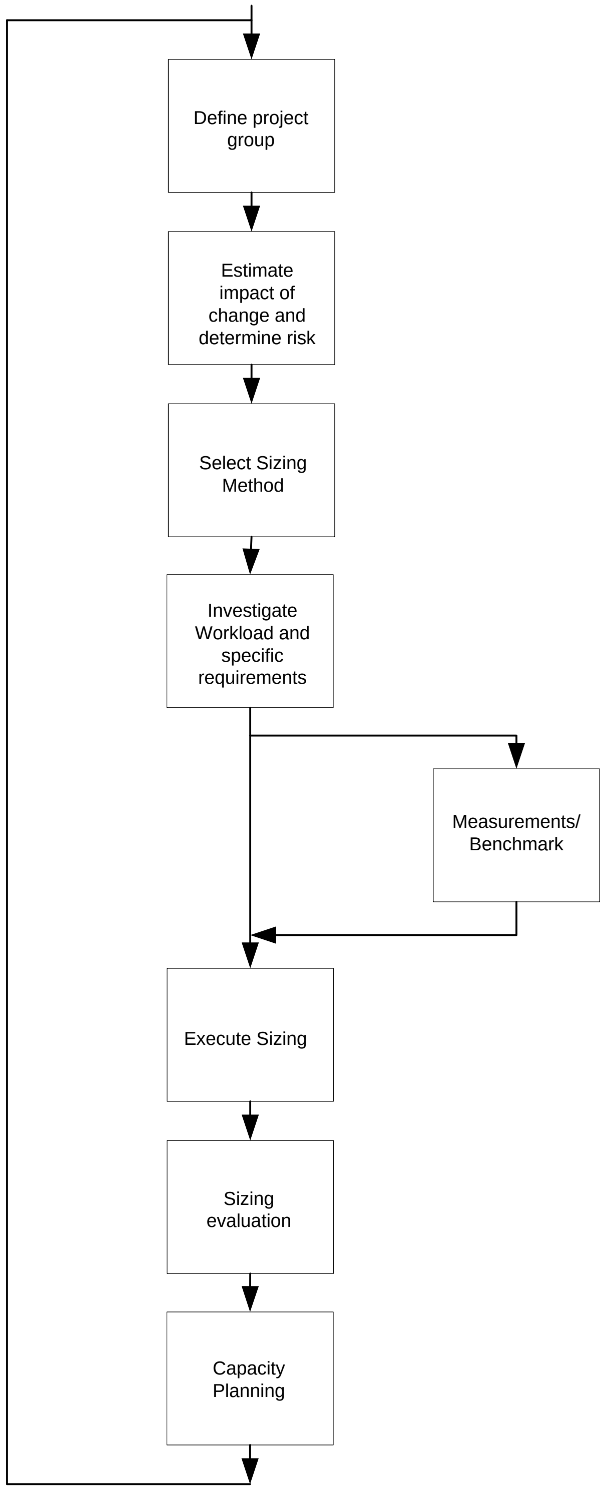

Figure 4-3 Main steps during a sizing

Select sizing method

Various sizing methodologies are explained in Chapter 3. One sizing method must be selected, and it depends on the situation which method is the best. It is a trade off between time, money, and accuracy. Section ‘Overview: advantages and disadvantages of various sizing methodologies’, in Chapter 3 gives information about the various sizing methodologies.

Investigate workload and specific requirements

An important step is the investigation of the expected workload and specific requirements. This investigation must be carried out thoroughly. You must realize that input which does not reflect the reality can lead to under-sizing or over-sizing. The sizing questionnaire and checklists can be used as a guide (see Chapter 13).

It is advised that you explain the questionnaire to the customer and why the information is important. If possible, the ideal situation is to fill in the questionnaire together with the customer and guide the customer as much as possible. Sometimes, customers do not always understand the questions, or have a different perception, and some do not see the importance of giving the right answers.

Measurements/benchmarks

If a non-theoretical sizing method is selected, benchmarks must be carried out and measurements done. For more information, see Chapter 3. From a project management point of view, this can be a time-consuming and costly exercise, but a sizing based on this approach can be very reliable.

Carrying out a sizing

Carrying out a sizing is the first step in making a sizing concrete. All the collected information based on measurements and questionnaire will be processed and the exact hardware capacity requirements will be determined. Together with specific requirements such as security and availability, concept configurations are made. These configurations will be discussed in the project group and the advantages and disadvantages will be discussed and weighted. Price is an important factor in this stage of a sizing process because it can cause requirement changes and force concessions. Also, the vendor selection is made during this stage. For this, see sections ‘Hardware vendor selection’, ‘Operating system selection’, and ‘Database vendor selection’, in Chapter 11. This makes the process very repetitious.

Sizing evaluation

After the sizing is finished, and the environment is up and running, it is advised that you evaluate. The following are reasons to evaluate:

- To check whether the load on the system is the same as expected and to determine the gap.
- To evaluate the process. What can be learnt for future sizings?

- To find out if the amount of headroom on the system is the same as expected. Otherwise, action must be taken before the implementation continues.
- To determine if response time guarantees are given and to check whether these are matched.

Future sizings/Capacity planning

Sizing is not a one time event, and you must implement a process to ensure that sizing is carried out on a regular basis. See sections ‘Sizing definition’, ‘When is a sizing carried out’, in Chapter 2 and Chapter 4. A sizing must be considered by every planned configuration change. However, sizing is also necessary without configuration changes. The load characteristics can change overtime. Do not wait until problems occur, but anticipate the problems.

Chapter 5 Infor Sizing model

5

The Performance and Benchmarking Center of Infor have developed their own sizing methodology. The model has already existed for several years and has been proven (see also section ‘Sizing according to the sizing guide’, in Chapter 3).

Sizing model

Figure 5-1 shows the Infor Sizing Methodology. The sizing model is a user-based model and not a transaction-based model. For more background information, see section ‘User-based versus transaction-based model’, in Chapter 5. From the Figure, it can be derived that a sizing is driven by three driving factors: OLTP-load, batch load, and Implementation choice.

OLTP-users interact frequently with the application; they search, browse, insert, delete, and mutate data. The response times are relatively short. There are no fixed criteria to classify a user as an OTLP user. To give an order of magnitude, OLTP responses are no longer than 60 seconds.

Batches/batch jobs are programs that run without user interaction. Multiple types of batch/batch jobs exist. Some batch jobs have a short duration; other batch jobs run for hours. Examples of critical batch jobs include Distribution Requirements Planning, MPS/MRP runs, end-of-year programs, and heavy print and query programs. Integrations that continuously exchange data are also seen as batch programs. There is no fixed value to classify a batch user. To give an indication, a batch is a program that runs for at least 10 minutes. Batches can run for many hours.

There is a gray area between OLTP users and batch users, which are the semi-batches. Examples of semi-batches are small reports. Semi-batches are classified separately in the sizing model. When there are many semi-batches in the customer environment, the semi-batches are counted and grouped as one or more batch/batch job(s).

The reason a distinction is made in the sizing model between OTLP and batch usage is that the system handles the various loads differently. The differences depend on the combination OS, Database, and application. These details will be described elsewhere in this document. Sometimes a batch can claim a complete CPU, or in the case of WMS, even more. See section ‘Response times Batch’, in Chapter 5.

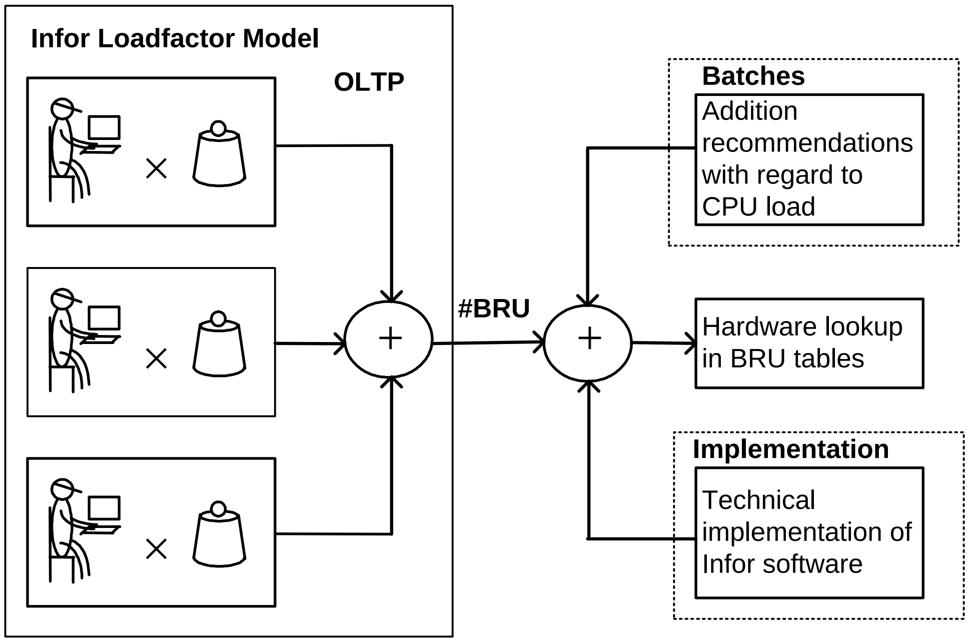
Infor Loadfactor Model
OLTP
#BRU
Batches
Addition recommendations with regard to CPU load
Hardware lookup in BRU tables
Implementation
Technical implementation of Infor software

Figure 5-1 Infor Sizing Methodology

The technical implementation is another important factor for the sizing. An example of an implementation choice is two-tier versus multiple tiers selection of database or hardware platform.

The combination of OLTP load, Batch load, and the implementation choice deliver sufficient information to describe the hardware capacity requirements. Based on this information you can search for a suitable hardware model.

Infor Loadfactor Model

The heart of the Infor Sizing model is the loadfactor model. This model calculates the load of an Infor implementation on a certain hardware configuration. The Infor Loadfactor Model is the part of Infor Sizing Methodology that qualifies the load of OLTP usage of a specific Infor application. The input for this model is the number of *concurrent* users.

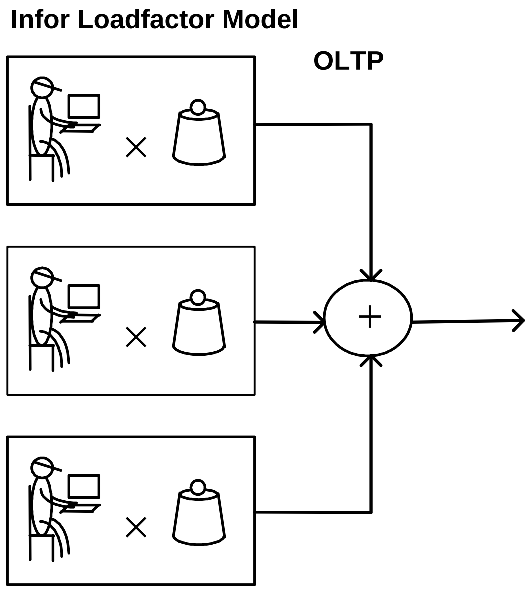

Figure 5-2 Infor Loadfactor Model

A number of different user profiles characterize the software. Various users characterize different user profiles and generate a certain load, as shown in Figure 5-2. These user profiles are related to the application areas, such as Manufacturing and Distribution. Batches are not included in the Infor Loadfactor model.

BRUs, Benchmark Reference Users

The load of the OLTP users on the system is expressed in the measurement unit BRU. A BRU stands for Benchmark Reference User. A BRU is a fictitious user with a loadfactor 1. All various loadprofiles, such as Sales user and Purchase users are expressed with a multiplier of this reference user. The heavier functionality a user represents, the higher the loadfactor is. The sum of the multiplied number of users per category multiplied by the loadfactor for this category determines the total required BRU capacity.

Note that a BRU does not correspond directly with a real user, although it is possible to have a real user with a loadfactor 1 also. The BRU is a unit which is independent of the underlying hardware. The translation from BRUs to hardware capacity for Infor LN can be found in the hardware tables. The hardware tables show how many BRUs a certain configuration can handle. In the hardware table, there are figures for the various implementations such as two-tier, three-tier application mode, three-tier database mode, and the various operating systems, hardware models and databases. For example, a certain application module with loadfactor 5 can claim on a certain configuration more system resources with database ‘A’ than with database ‘B’. In this case, the hardware tables show lower values for database ‘A’ than ‘B’. In both situations, the loadfactor is 5, because from a functional point of view the same functionality will be carried out.

The hardware tables show how many BRUs a certain configuration can handle. For Infor LX, the BRU is expressed in CPWs.

Note that a BRU belongs to a certain application version or release. An Infor Baan IVc BRU has a different load on the system than Infor LN. They differ although the two versions belong to the same product family. Remember that a BRU is separated from the technical infrastructure. A BRU is related to the functionality. For example, when the same functionality is carried out with Infor Baan IVc and Infor LN, the resource consumption on the system will differ, because the underlying software is different. This is why a BRU also differs between the various versions.

Sizing example

Business case: Sizing for Handy Tools Inc.

This example shows a sizing based on the Infor sizing model. The example is based on the fictitious application, SalesApp.

The company Handy Tools Inc. wants to upgrade to SalesApp 2.0b. Handy Tools asked Infor to determine the hardware requirements for the new environment.

Handy Tools Inc expects the following workload and number of users during their peak hours. See Table 5-1.

Application usage	
Application OLTP usage	**#Concurrent users**
Sales Order Entry	27
Business Partner management	3
Order Picking	15
Concurrent batches	
Finance End Period processing	1

Table 5-1 Load description of Handy Tools Inc

The information published in the following sections is taken from the SalesApp 2.0b sizing guide. SalesApp is not a real application, but in practice every sizing guide contains similar information.

Hardware table with BRU capacity copied from SalesApp 2.0b sizing guide

Table 5-2 lists various hardware models. From each model, the SalesApp 2.0b BRU capacity is given per implementation category. All figures are only applicable for DemoSQL DB on the Linux platform.

Maximum number of BRUs for Linux with DemoSQL*DB			
		3-Tier	
Model	**2-Tier server**	**Appserver**	**DB server**
SuperServer, Xpower, 2000MHz, 16 MB L2 Cache, 1 MB L1 Cache			
SuperServer G320 – 1 way	100	120	220
SuperServer G320 – 2 way	198	237	435
SuperServer G320 – 4 way	388	465	853
SuperServer, Xpower, 3000MHz, 32 MB L2 Cache, 1.5 MB L1 Cache			
SuperServer H340 – 1 way	150	180	330
SuperServer H340 – 2 way	296	356	652
SuperServer H320 – 4 way	581	698	1279

Table 5-2 SalesApp BRUs per hardware model

Loadfactor table taken from SalesApp 2.0b sizing guide

Table 5-3 gives the load factors per application module.

Loadfactors SalesApp 2.0b	
Application module	**Loadfactor**
Sales Order Entry	4
Business Partner management	1.5
Warehouse management	3
CRM connector	5
Order Picking	6
Invoicing	3
Cost price calculation	4.5

Table 5-3 loadfactors SalesApp 2.0b

Batch requirements copied from SalesApp 2.0b sizing guide

SalesApp is a single threaded application. Batches try to claim a complete CPU on the system in a 2-tier environment. The ratios of CPU resources spent in the application and database layer is not equally distributed. The following table gives an overview of the batch resource consumption. Identical CPUs are assumed for the application and database server.

Batch load			
		3-tier	
Batch	**2-tier**	**Appserver**	**DB server**
CRM Integration	1	0.7	0.3
Distribution Resource Planning	1	0.5	0.5
Finance End Period processing	1	0.6	0.4

Table 5-4 CPU per concurrent batch

CPU Calculation

This section shows how to calculate the CPU requirements for Handy Tools Inc.

All users must be categorized based on functionality. The number of concurrent users during the peak hours is determined (see Table 5-5). The number of concurrent users is multiplied with the associated load factor. The results of these calculations are summed together.

Application usage			
Application functionality	**#Concurrent users**	**Loadfactor**	**BRUs**
Sales Order Entry	27	4	108
Business Partner management	3	1.5	4.5
Order Picking	15	6	90
Total:			203

Concurrent batches	
Finance End Period processing	1

Table 5-5 Load description of Handy Tools Inc

The required OLTP capacity for Handy Tools is 203 BRU. The customer expects one concurrent batch during peak hours. In a 2-tier environment, 1 CPU will be reserved for this batch (see Table 5-4). In a client server scenario, this batch claims 0.6 CPU on the application server and 0.4 CPU on the database server.

System selection

Handy Tools Inc. wants to re-use an existing system and prefer a 2-tier environment. The customer owns a SuperServer, Xpower, 2000MHz, 16 MB L2 Cache, 1 MB L1 Cache. From Table 5-2, it can be derived that in 2-tier mode, this system can handle 198 BRUs. 203 BRUs are needed. So, a 2-way system is not sufficient. It is advised to use a 3-way system for OLTP. The OLTP load capacity for batches has to be reserved. In the previous section it is derived that 1 CPU is needed; in total, 4 CPUs are needed.

If the same calculation is made for the SuperServer, Xpower, 3000MHz, 32 MB L2 Cache, 1.5 MB L1 Cache, the required number of CPUs differs. A 2-tier system can handle 296 BRUs, which is sufficient to serve the required 203 BRUs. To handle the expected batch load, one CPU is needed. In total, a 3-way system is required.

3-tier sizing works in a similar way. A SuperServer G320 – 2 way can handle 237 BRUs as application server, and a SuperServer G320 – 1 way can handle 220 BRUs as database server. This is sufficient to handle the required 203 BRUs. From Table 5-4, it can be derived that 0.6 CPU is needed on the database server, and 0.4 CPU on the application for this specific batch. In total, 2.4 CPUs are needed on the application server and 1.6 CPUs on the database server.

User-based versus transaction-based model

The Infor sizing model used in the sizing guide is based on a user-based model, and not on a transaction-based sizing model. This section provides some background information on why Infor selected a user-based sizing model.

The decision to develop a user-based or transaction-based sizing model was crucial when the sizing model was developed. Both models are used in the industry and are valid options. Both sizing models have their advantages and disadvantages. Ideally, it would be good to have both models existing alongside each other. But it is too big of an investment to maintain two models. After long discussions, the user-based sizing model was favored.

Usually, the first reaction is that a transaction-based sizing model results in more accurate sizing results; in practice, this is not true.

After studying both models, it becomes clear there is some overlap. Primarily, both models have basically the same problem. For both models, the same questions must be answered from a different angle.

The key sizing question is “What is the load per unit of time?” In a user-based sizing model, this is driven by the number of concurrent users. In a transaction driven model, this is based on the number of concurrent transactions. For a user-based model, it is hard to determine the number of concurrent users. For the transaction-based model it is difficult to determine the number of concurrent transactions per unit of time. Often, customers have difficulties in providing this data. Typically, it is easier to get information about the total number of concurrent users in the company, than to get information about the total number of concurrent transactions. When customers lack this information, it is easier for them to estimate the number of concurrent users than to estimate the number of concurrent transactions. This is one of the reasons Infor chose a user-based model.

Infor believes both models deliver approximately the same level of accuracy for the majority of the customers. Other selection criterion is that a user-based model is easier to implement, less complex, and is a smaller investment.

Many application vendors have chosen a user-based sizing model.

The following is a summary of the considerations made during model selection:

- A user-based model is easier to measure and benchmark.
- A user-based model is easier to understand for the majority of customers.

- Many customers do not have detailed information available. In this case, the user-based model is best. The number of users is easier to estimate.
- Hardware becomes a commodity. Customers are less willing to invest in sizing because they want to be advised quickly. They are reluctant to answer many detailed questions. Hardware profit margins went down and do not always allow an extensive sizing exercise. A user-based sizing model is more suitable for a quick sizing than a transaction-based sizing model.
- Infor received feedback stating that the transaction-based model is too time-consuming for customers to come up with the detailed transaction data.
- The user-based model is perfect for small to medium sized customers, which is the majority of the install base. Typically, the minimal hardware requirements are more than sufficient for them. An extensive sizing exercise will lead to the same results.
- A transaction-based sizing model is usually more accurate than a user-based model. If the model is implemented correctly, a customer sizing can be carried out in more detail compared to a user-based model.
- A transaction-based model is more suitable for larger customers. Larger customers usually have a better view on their transaction volumes than smaller customers. They have a bigger IT-department and can effort to invest in management and control tools and systems. From an operations point of view, it is important that they have continuously up to date information available to control their business operations. However, the larger customers typically do not only rely on a sizing guide sizing, but go through an intensive stress test or customer specific benchmark cycle. The sizing guide is used to improve their confidence level and set targets for their own testing.
- A transaction-based sizing model connects better to the 'think world' of customers. For example, a customer understands perfectly what concurrent sales orders means, but it is vaguer for them to understand what concurrent users are doing.
- Transaction information is usually the only reference for new customers about their old environment. From their current used application they can derive transaction volumes. The number of users can also be measured with concurrent application, but this can change completely with the new application due to changed procedures.
- A transaction-based model gives the customer the belief you are working very accurately, which is true. However, the transaction characteristics can vary per application version and service pack. Application parameters also can impact the transaction performance characteristics. There are many application settings, and all these factors together can make a significant difference. There are too many different options to test and this makes it hard to maintain and keep it accurate, which a transaction-based sizing model totally provides.
- Each customer is unique. In every situation, the system reacts differently. This makes it difficult to come up with a detailed transaction model.
- It is difficult to base a memory sizing model on a transaction-based sizing model. The memory requirements are mainly driven by the amount of user application instances running. For accurate memory sizing, a user-based model is necessary.

It is possible to carry out a transaction-based sizing with a user-based model to a certain extent. The transaction volumes per user, which are used in the benchmark to populate the sizing guide, can be used for this. Benchmark transaction volumes are published in almost every sizing guide.

Chapter 6 Sizing recommendations, definitions, and attention points

6

This chapter contains a variety of important sizing subjects. The subjects in this chapter are important because many questions in the sizing questionnaire refer to these subjects

Various user types

In the sizing model, three categories of users are defined. It is important to understand the differences between these categories. For sizing, it is important that user amounts are chosen correctly. Also, the ratios between the various user categories should be determined carefully; otherwise, some formulas give unrealistic results which leads to under-sizing or over-sizing.

The various sizing user categories differ from user categories defined for licensing and pricing. There is no direct relationship assumed with licensing and pricing.

Named users/Potential users

A named user is a user who has a login account for the system, but does not necessarily work with the system at a given point of time. A named user has the ability to work with the system.

A named user claims only system resources when the user is logged onto the system; the only exception is disk space. Data entered earlier by this user needs to be kept on disk. The category named users is introduced for disk space sizing.

Connected users/Logged on users

A connected user is a named user who is logged on to the system and can actively work with the system or can be inactive. A user which is logged on and has, for example, a lunch break or is on the telephone also belongs to the connected user category. A connected user always claims memory on the system. On certain systems, the memory used by inactive users will be paged out. A connected user only claims CPU resources when they are active. The categories Connected/logged on users are introduced for memory sizing.

Concurrent users/Active users

A concurrent user works actively with the application or system simultaneously with other active users.

An active user claims CPU and memory resources. How many CPU resources are claimed depends of which application functionality is carried out, and the activity level and work speed of the user. In the Infor sizing model, a user is counted as a concurrent user when the user causes at least once some system activity within five minutes. There are users which run heavy processes with duration of at least five minutes. These users are inactive themselves during processing period, but from a resource point of view they are counted as a concurrent/active user.

It doesn't need explanation that a very fast working user causes a higher load on the system compared to a slower working user. The Sizing model does not distinguish between the activity levels of the various concurrent/active users. The sizing model is based on an average working rate. There will be fast and slow typing users at the customer site. It is assumed that at customers the average work speed has on average a comparable activity level with the benchmark, where the sizing model is based on. It is intentional not to make a distinction between the intensity of concurrent users, otherwise the sizing model becomes too difficult. It is too time-consuming to define the activity level for each user, and the overall behavior is too dynamic.

There is a difference in resource consumption between the application functionality carried out by the various users. This load difference is expressed with the load factor model. See section 'Infor Loadfactor Model', in Chapter 5.

Ratio between different types of user

Figure 6-1 shows that Concurrent users are a subset of the Logged on/Connected users, which is a subset of the number of Named users. The Named users are a subset of the total number of Employees, which makes the assumption that external people do not have access to the system.

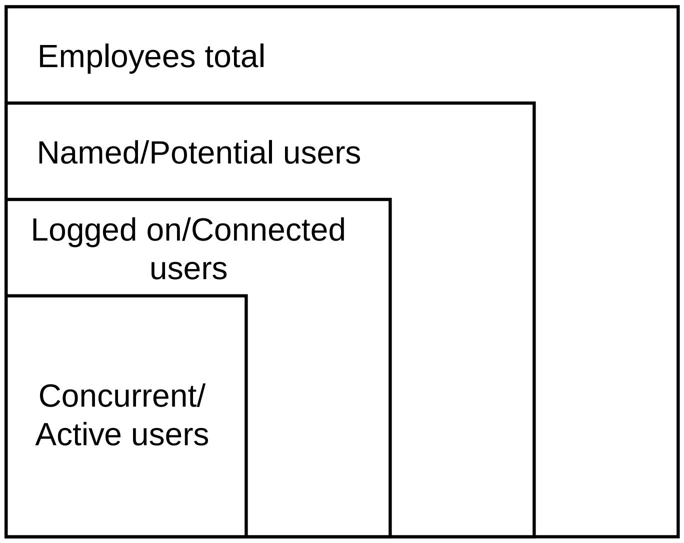

Figure 6-1 User distribution

Impact of ratios on the sizing

It is important that ratios are determined carefully because a mismatch can lead to under-sizing or over-sizing.

In the sizing model are the Named users input for the disk space sizing. When the number in Named users is too low, this leads to insufficient disk space. When the number of Named users is too high in the sizing, the amount of calculated disk space can be too high. This is less of an issue, as disks are relatively cheap. Too many Named users reserves future capacity, which is itself not a bad thing, but other sizing parameters take care of margins.

The connected/logged on users are used for internal memory size calculation. It is clear that a too high or to low value will directly impact the calculated memory size.

The amount of memory for a connected user used in the memory formulas is lower than the memory consumed by a real active/concurrent end-user. This makes sense, because when a user is inactive for a longer period of time, the system will page out inactive memory pages which belong to the user, to free up memory for other activities. Therefore, it is crucial to base the memory calculation on the number of connected users.

Some customers want to have the lowest number of licenses as possible, because this can save license costs. Inactive users are forced to log off. In this type of situation, the number of connected and concurrent users is almost the same, which will lead to memory under-sizing because the memory reserved in the formulas for a connected user is lower than for real concurrent user. Despite the fact that the number of connected users used in sizing matches with the real number in the customer situation, it is advised to apply the general ratio mentioned later in this section.

The CPU sizing is based on the number of Concurrent/Active users. Mismatches in this number will have an immediate impact on the calculated CPU in the sizing.

Ratio based on experience

Infor recommends to determine the exact number of users because this provides the most accurate sizing result. There are situations where this is not possible or is too time-consuming to determine the exact values. The consequences and possible risks are explained in the section above.

In this type of situation, it is recommended to estimate the number of concurrent users as good as possible. You can count the number of employees per department which actively work with the system. General rules based on experience are as follows:

Small and medium enterprises:

#connected users = 1.5 * #concurrent users.

#named users = 3 * #concurrent users.

Large enterprises:

#connected users = 1.75 * #concurrent users.

#named users = 4 * #concurrent users.

Various Infor software products

The product portfolio of Infor is broad. Infor delivers solutions for various logistical and financial areas and each product is specialized for a specific task. Many customers use, or are planning to use, multiple Infor products. Infor has sizing guides for the various products. These sizing guides approach the various products individually, which leads to at least one server per application. In many situations this is not ideal. During the sizing process, you must introduce a consolidation cycle and check whether products can be combined on fewer servers. Besides consolidation of resources you must check whether there are no constraints regarding operating system, database, and middleware usage.

Non-Infor Software

Sometimes, non-Infor software is installed besides the Infor software on the same server. This software fulfils specific tasks and can be delivered by Infor if it is software of partners.

This software can have a significant impact on the sizing, which is dependent on the load and frequency that the application is used. Both Infor and non-Infor software compete for the same system resources. Therefore, it is important to evaluate the load impact during system sizing.

If the software is not delivered by Infor, Infor cannot take ownership for the sizing. In this case, contact the software vendor for sizing information and load characteristics of the specific product. Key information is the impact on CPU, internal memory, external memory, and network. The person who is end responsible for the sizing must collect all the sizing information of the various products and combine it into one sizing.

If, for a specific product, no sizing information can be found, Infor recommends that you use one of the methods described in Chapter 3, other than the theoretical sizing method using the sizing guide.

Integration

Certain applications are developed for very specific tasks, such as Constraint Planning. These applications are used for their strength in their domain area. Their functionality is not covered by the backbone applications. These applications get their input from other applications and send the results back to the other applications. Data will be exchanged using an integration process. There can be much integration within a total solution. The load of integrations can be very intense and large amounts of data can be sent back and forth. Often, the integration process formats the data into an application specific format. The mapping contributes significantly to the load of integrations.

Integrations which exchange frequently a large amount of data can have a batch type behavior on the system (see ‘Response time behavior’, in Chapter 11). Integrations cannot be negotiated or considered during the sizing because the load of integrations is underestimated and even forgotten in the sizing.

If, for a specific integration, no sizing information can be found, Infor recommends that you use one of the methods described in Chapter 3, other than the theoretical sizing method using the sizing guide.

Key information is the impact on CPU, internal memory, external memory, and network.

Customizations

Most of the software delivered by Infor is written for a market and not necessarily for a specific customer. The software covers a lot of functionality sufficient for the majority of the customers and can be parameterized to the specific needs of the customer. Sometimes, when the customer needs are very specific, the software will be customized.

Customizations have an impact on sizing, but this depends to what extent the software is modified. Slight modifications, such as changing the menu structure of field changes, will not impact the sizing, but others may.

The impact of customizations can never be derived from a sizing guide because it is customer specific. You must always evaluate the impact of the load. Experienced sizing consultants can estimate based on their experience, or use one of the methodologies described in section Chapter 3, other than the theoretical sizing method using the sizing guide.

Multi-byte/Unicode

Many Infor applications are able to switch between languages. Switching between languages does not have a large impact on performance and sizing unless you have to switch between character set. Switching from an eight bit character set to a character set which needs more then eight bit to present the characters can have a significant impact on the resource consumption of the involved systems. Examples of languages which need more than eight bits are Korean, Chinese, and Japanese.

The various applications use various technologies and implementation to support these languages. For example, Infor LN uses multi-byte, which is its own implementation. The latest Infor LN versions also support Unicode. Unicode is a standard for multi-language support. The impact on sizing cannot be ignored because this leads to under-sizing.

The resource consumption on disk, CPU, and memory are higher. More disk storage is needed because multiple bytes are needed to store one character; the same is true for internal memory. To store data strings in memory costs more space. Also, the CPU resource consumption is higher. It is more intensive for the CPU to process long strings instead of short strings. Program functions and system calls such as string compare have more work to carry out.

For the impact on specific applications, see the various sizing guides.

Sizing peak periods

At most customer sites, the load on the system varies throughout the day. Figure 6-2 shows a typical load distribution.

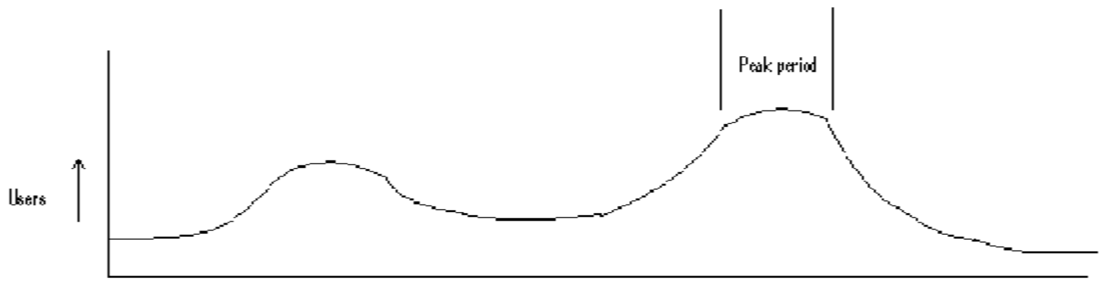

Figure 6-2 Distribution of user load

Typically, peak periods occur during the middle of the morning and afternoon. For correct sizing, you must identify these peak periods. Peak periods vary from customer to customer; you must take peak periods into account. A correct sized system can handle peak loads. As well as the daily peak periods, also consider weekly, monthly peaks, season peaks, or other periodic peaks. If the peak loads are small or infrequent, see Figure 6-3 and consider whether to accept temporary lower response times for the system. To size a system based on infrequent peak periods is not cost-effective and can result in an over capacity for most of the day. It is advised that you calculate with the average load of the regular peak periods.

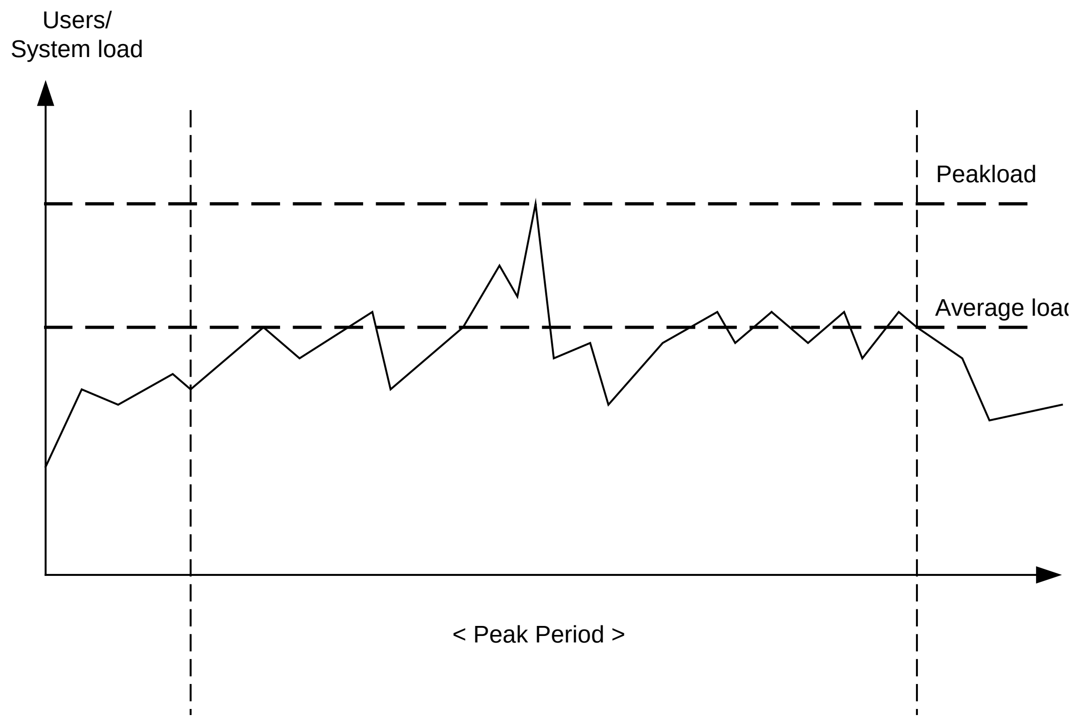

Figure 6-3 Load characteristic during peak period

Some businesses are season oriented. For example, a company which produces ice creams has the biggest selling peak in the summer, but the production starts in early spring (see Figure 6-4). Ensure the peak period used for the sizing is taken in the right season.

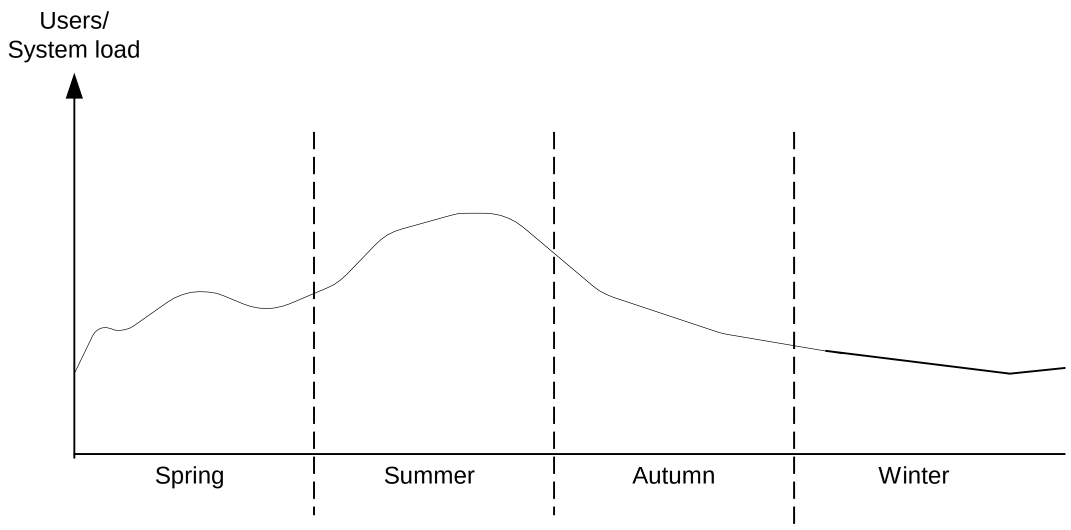

Figure 6-4 Seasonal Peaks

Peak load margin

The loadfactors published in the sizing guides are measured with a benchmark that has a flat load characteristic. Flat indicates that the variations on the CPU load are within the ten percent range. Customer situations differ. The load varies much more. You cannot predict the exact pattern, but you must take this variance into consideration. For this reason, Infor recommends that you do not size a system to the maximum capacity. A commonly-used industry standard is a peak load margin of 20 percent.

The peak load described in section 'Sizing peak periods', in Chapter 6 represents a long-term average. The 20 percent peak load margin is different from the long-term peak load average. The peak load margin represents short peak loads, such as if all users break for a meeting and then afterwards return to the system simultaneously.

Growth

It is important not to base a sizing only on the current situation, but also to look to the future. The system must be able to accommodate future growth requirements. How many years of growth to consider depends on the situation. A typical scenario considers three years. After three years, the hardware costs typically decrease, and CPU capacity increases so hardware replacement is often preferable.

If a customer buys an extendable system, they can add additional CPUs and memory as needed.

Discuss with the vendor the most effective scenario to deal with growth.

Margin

The sizing model itself does not include a margin. In certain situations, you must add a percentage on top of the sizing. The amount depends on the situation. Lack of time, amount of knowledge, priority, and accuracy can affect the sizing input. Not every customer takes the sizing process seriously, and therefore delivers inaccurate input. The customer must understand the importance of accurate and complete information on the sizing questionnaire.

Key information can be unknown when the sizing is carried out. In this situation, it is also advised to apply a margin.

Central versus decentralized solution

During a sizing exercise, the question is whether to work centralized or decentralized. In the past, each location had its own datacenter (see Figure 6-5). There is a clear trend that shows companies now tend to work more centralized. Two limiting factors are moving away. Server capacity has increased significantly over time, and more users can work on the same server. Also, WAN-capacity increased and network costs came down. Therefore, a central solution becomes more achievable.

The definition of centralized is not always used consistently in the sizing world. The term centralized indicates there is one or a few datacenters, instead of having many datacenters/servers on many locations. Some sizing experts say centralized is actually one single large server on a central location, (see Figure 6-6). Other experts see multiple servers involved, but all on one location, (see Figure 6-7).

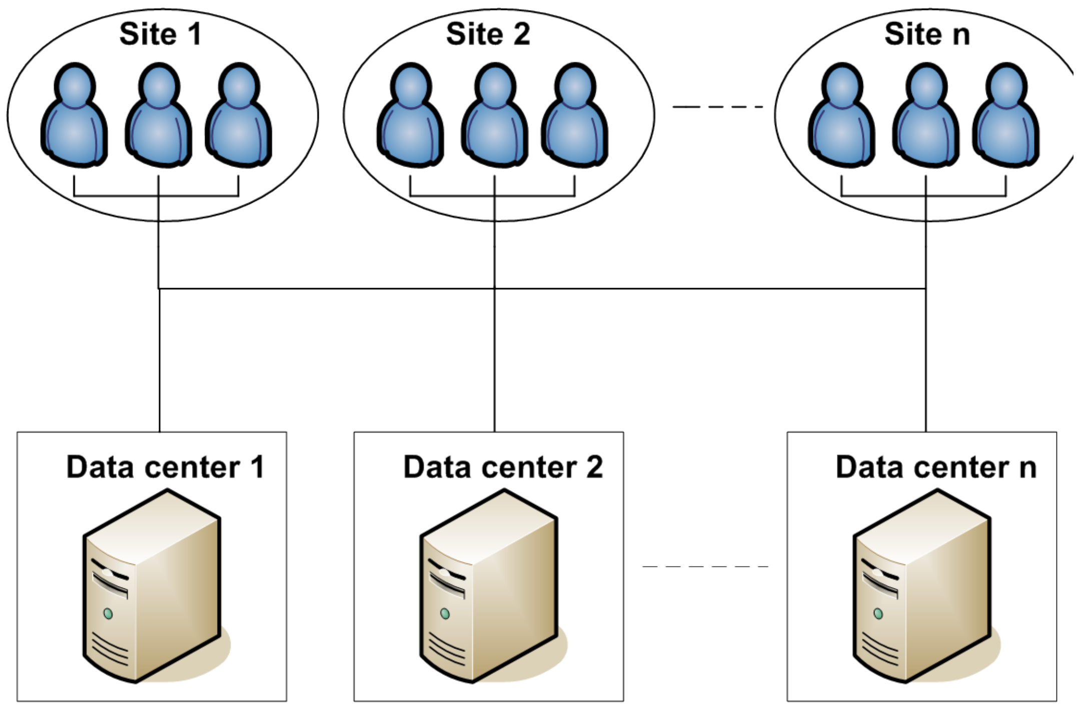

Figure 6-5 Decentralized solution

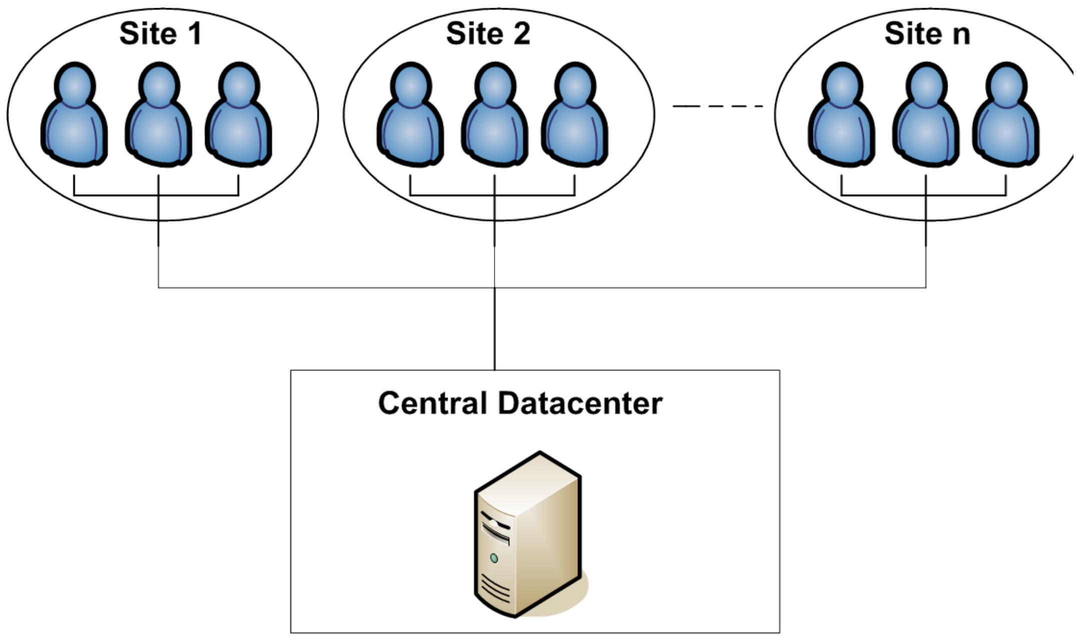

Figure 6-6 Centralized solution- single server

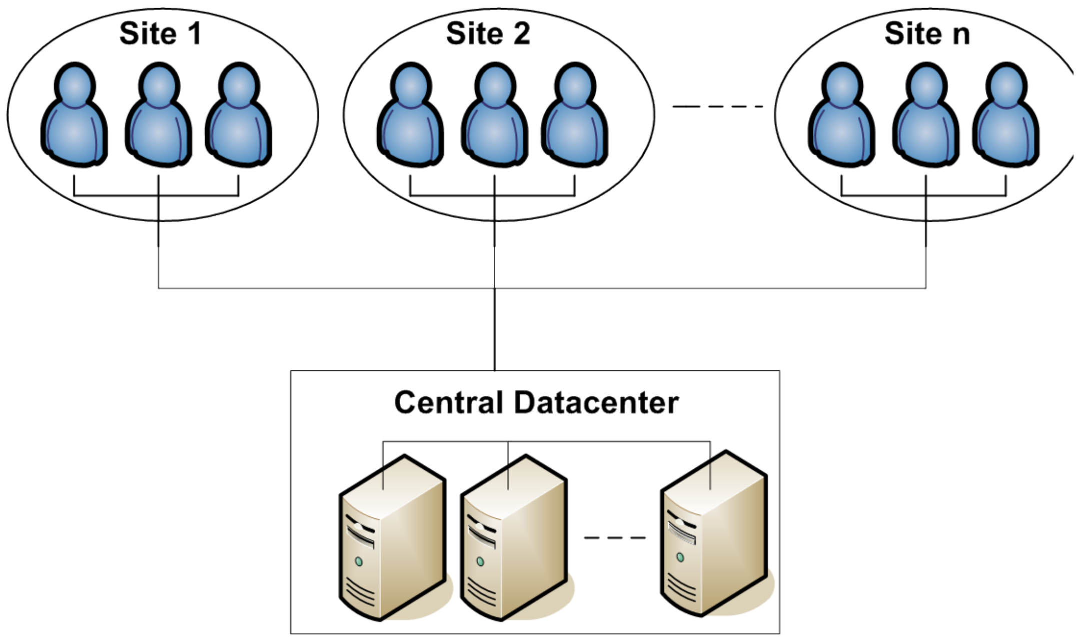

Figure 6-7 Centralized solution- multiple servers

Advantages of central solution

The following are advantages of the Central solution:

- Every customer uses the same software.
- No complex data exchange architecture between the various servers.
- Only one environment must be maintained.
- IT department can be centralized.

Disadvantages of central solution

The following are disadvantages of the Central solution:

- Higher network costs. Instead of connecting to a local server, users connect using a WAN connection remote.
- Response times can be worse. WAN connections have lower bandwidth and higher network latency than local connections.

- From an availability point of view the risk becomes higher. In case of a system failure, more users are down compared to a decentralized solution because there are more users working on the same server. This involves using high quality hardware.
- You rely on the network. It is advised to design the network infrastructure in such a way that network traffic to the central server can be routed using multiple ways.

Batch window

The batch window is the timeframe reserved to carry out the batch processing. Batches can also run during normal operation hours parallel with OLTP work, but this is not counted as batch window.

The length of the batch window depends on the type of business a company operates in. Companies which only work during normal office hours have sufficient time to process their batch jobs. Companies who have activities 24 hours a day have less time to process their batch job (see Figure 6-8).

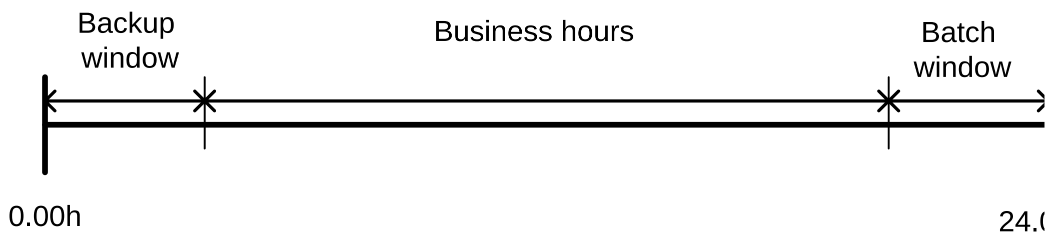

Figure 6-8 Time schedule

Centralization of environments has a negative impact on the batch window. More data must be processed and when sites operate in different time zones, the batch window can be very short. For more information about centralization, see section ‘Central versus decentralized solution’, in Chapter 6.

From a sizing point of view, it is important to know how long the batch window is. To ensure the batch processing fits into the batch window, take the following actions:

- Faster or better hardware; faster CPUs or more memory can speed up processing times. Parallelization: Run different functional batches in parallel as much as possible, instead of sequential; this can shorten the total batch processing time. Certain applications support parallel batch processing. Functional batches can be split into multiple sub-jobs and processed in parallel; this shortens batch duration. Infor LN supports parallel bshell, (see section ‘Infor LN: Single bshell versus parallel bshell’, in Chapter 11).
- Run batches without any network layer involved as much as possible. For example, batches running in two-tier are faster than running batches in three-tier.
- Do not process more data than necessary. Evaluate which level of detail is necessary for the batch processes. For example, when a range of projects must be processed, do not process all projects. Do not set planning horizons too far away if only short-term information is used.

Backup window

The backup window is the time available to create the backup.

How long the backup window can last depends of the type of business a company operates in. Companies which only work during normal office hours have sufficient time to create a backup. Companies who have activities 24 hours a day have less time to backup.

Centralization of environments is not beneficial for the backup time. Especially when sites operate in different time zones, the backup window can be very short. For more information about centralization, see section ‘Central versus decentralized solution’, in Chapter 6.

Usually, the backup window time is critical and the available time is always under pressure. The amount of data increases, so it takes longer to backup. For batch processing, see section ‘Batch window’, in Chapter 6. To ensure the creation of the backup fits into the backup window, take these actions:

- Incremental instead of full backup
  It is less time-consuming to only backup the changes made after a certain point, instead of creating a backup of all data. A full backup can be made in the weekend.
- Online backup
  Certain databases support online backup capabilities. This means that a backup window is no longer needed to backup data. Note that this has a negative impact on the normal operational system load.
- Faster media
  The technology to create backups evolves and the speed of the devices increases. It can be useful to invest in the latest technology.
- Disk to disk copy
  There is a tendency that disk subsystems are equipped with cheap disks besides the normal operational fast disks. These disks are used to create a fast backup of the data. The downtime to create the copy is very low. This backup to the slow disks is used as input for the backup to slower by less volatile backup media, like tape. The time to finish the backup is less critical in this situation.
- Third mirror
  Certain disk subsystems have the capability to configure a third disk mirror. This mirror can be split at a certain moment of time. The third mirror contains an exact copy of the data. The third mirror can be used as the input of the backup process. The time to finish the backup is less critical in this situation.
- Flash copy
  Certain disk subsystem offer functionality to create fast copies using flash copy. Only the changed data blocks will be copied. The copy is disk-to-disk. The copied data can be used as input for the backup. The backup time is less strict.

Chapter 7 CPU sizing

CPU sizing is an important factor in the overall sizing. Insufficient CPU leads to a solution performing badly. User response times will become unacceptable, and batch processing times will exceed their criteria.

If the CPU is undersized you can add CPUs or buy faster CPUs, but it is not always guaranteed that the model can support this. It can indicate that the system has to be replaced, which can have a big impact.

The various methodologies to size the CPU are described in Chapter 3. Infor has developed its own sizing model for CPU sizing (see Chapter 5).

The CPU sizing indicates that other decisions must be made, such as the following:

- CPU cache
- Clock speed
- Number of CPUs
- System architecture
- Hyper treading
- Multiple cores CPUs

Background information about these subjects can be found in Chapter 11.

Chapter 8 Internal memory sizing

8

Memory sizing is an important part of the overall sizing. Memory under-sizing leads to severe performance problems, and memory over-sizing makes the solution less cost effective.

This chapter gives an overview of the factors which affect memory sizing. Only basic background information is provided to understand the used memory formulas in the sizing guides.

Memory sizing general

Distribution of resources

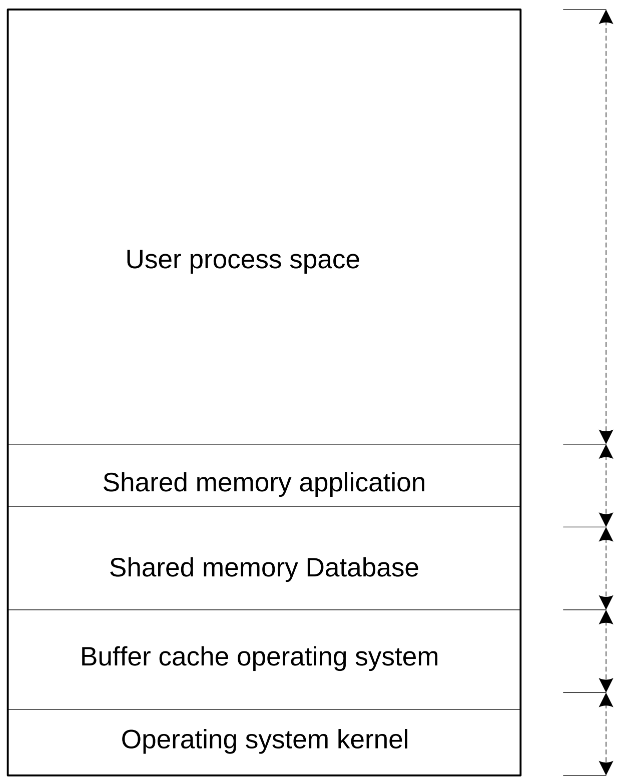

Figure 8-1 Memory distribution

Figure 8-1 gives a general overview of how the internal memory of a system is used. Naturally, it depends on which application is running and how an application is designed. Therefore, there are different memory formulas for the various products.

In every situation, memory is needed for the operating system. The size of the memory used by operating can vary depending on how the application works. When there are many processes active, and the programs have a lot of open files, the kernel size will be bigger. However, generally, the size of the kernel will not vary with large steps and is quite stable. The initial kernel size depends on the size of the system and how the kernel is configured. The maximum size depends of the maximum value the kernel parameters are configured. Many kernels are dynamic and cannot be configured. With these kernels, you have less influence on the memory size.

These are general rules:

- On small sized systems, the kernel size is 256 MB.
- On midrange systems, the kernel size is 512 MB.

Buffer cache

The buffer cache belongs to the operating system and is used to make disk IO more efficient. For more information about caching, see section ‘Caching’, in Chapter 11.

The buffer cache can be static or dynamic and can vary per operating system. On certain operating systems, you can configure static or dynamic buffer cache. Static means the size of the buffer cache is fixed. In the case of a dynamic buffer cache, the size can vary between a minimum or maximum configurable memory values. On certain operating systems, the buffer cache grows when adequate free memory is available.

The dynamic buffer cache can lead to confusion. On such systems, it looks like there is no free memory left because the buffer cache grows to its maximum value. In this case, the size of the buffer cache can be larger than necessary for good performance. A big part of the buffer cache can be seen as free memory because it does not contribute to a better overall performance. You must view the cache hit ratio of the buffer cache to know whether the buffer cache has an optimal size.

Sometimes the behavior of the dynamic buffer cache leads to little paging. Usually, paging can be an indication of insufficient memory. In this case, you must be careful. If the buffer cache has reached its maximum value and new processes are started, such as when a new user logs on, you can see some paging. The buffer cache shrinks to free up memory for the started user processes. Paging caused by this phenomenon has nothing to do with a lack of memory. For background information about paging, see section ‘Paging and swapping’, in Chapter 8.

The advisable size of the buffer cache depends on what is running on the system. If the application does a lot of file system I/O, a large buffer cache is needed. For example, a database can be created to work with raw-devices or file-systems. In case raw devices are selected, the buffer cache can be smaller because raw devices do not use the buffer cache. Some operating systems support options to bypass the buffer cache for file system I/O. This is called direct IO.

These are general rules:

- The average size of the buffer cache for a small and midrange system is ten percent of the internal memory.
- The average size of the buffer cache for a larger system is five percent of the internal memory.

Paging and swapping

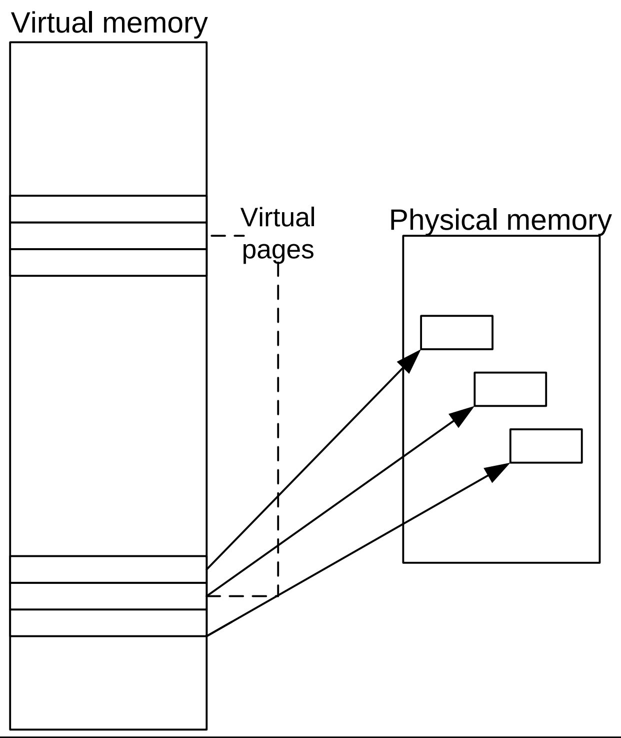

Figure 8-2 Mapping from virtual memory pages to physical memory pages

For a server to perform well it must be equipped with sufficient memory. If the amount of memory is insufficient, the system starts paging. If the system is under very heavy memory pressure it starts swapping. The performance of a system deteriorates significantly when it starts paging or swapping.

Paging is the mechanism whereby internal memory pages are mapped on the pagefile on disk, the virtual memory, (see Figure 8-2). The size of the pagefile is at least twice the size of the internal memory. If a system comes under memory pressure, and there is insufficient internal memory for the active processes, the system will move memory pages which are not currently used to the pagefile to free up internal memory. In many operating systems, this is based on a Least Recently Used mechanism. If a certain process needs the memory page later, the page can be read back from the pagefile into the internal memory. Memory accesses of pages located in the pagefile are resource intensive. The duration before a memory page is loaded is relatively long because disks are slow devices compared to the speed of internal memory. This works well for situations where pages can stay for a relatively long time in the pagefile. If pages are constantly moved back and forth between the pagefile and internal memory, the system gets very busy and the disk subsystem is loaded. Therefore, the system performance will become very bad.

Swapping happens when the system is under very heavy memory stress. Memory of complete processes will be moved in bulk to the pagefile. If swapping occurs, the performance is very dramatic.

The memory formulas are developed in such a way that the system has sufficient memory and does not start paging and swapping. The events described in this section show that memory sizing must be carried out carefully.

Memory formulas

The memory formulas used in the various sizing guides are based on the following format:

$I_{tot} = I_B + n * I_n$

Whereby the following applies:

I_{tot} = Total amount of memory needed.

I_B = Base amount of memory.

I_n = Amount of memory per user.

n = Number of **connected** or **concurrent** users.

I_B includes all the static memory components such as OS kernel, buffer cache operating system, initial shared memory database and initial shared memory application. There is one size given for all systems which is based on small to medium sized servers. For larger servers, it is advised to increase I_B.

The memory consumption increases if the number of users grows; this is expressed in $I_{n.}$

The memory formula is based on a linear approach; in practice, this will not be the case. A high number of users have a relatively higher degree of data sharing compared to a low number of users. This means the memory consumption per user will be lower in the higher user range. The factors in the formula are average factors. It is possible to introduce a more detailed model, but it has been decided not to do so because the model becomes very complex and difficult to maintain. A detailed model will also acquire a high level of accuracy, which is useless due to the dynamics of the users. The user variations are too dynamic to fit into a detailed model. Having a detailed model and using rough sizing input does not make sense.

The number of **connected** or **concurrent** users is an important input value. It is important for you to understand the differences between these two user types, which are explained in section ‘Various user types’, in Chapter 6.

The memory formula is based on an assumed ratio between connected and concurrent users of a factor 1.5. If the ratio at the customer site is lower, it is advised that you increase I_n.

Chapter 9 Disk sizing

9

Disk sizing is often recognized as the activity to determine the size of the disk space needed to store data. Besides disk space, the I/O throughput is another important factor. The disk subsystem must have sufficient power to catch up with the read/write requests it receives. From a performance point of view, a slow disk subsystem can be blocking for the overall system throughput.

Business data is vital for an enterprise. The disk subsystem must be reliable, because losing data is not desirable. In case of a disk failure, various technologies exist to increase the availability of the disk subsystem and restore data. This chapter gives some background information on these topics.

Disk sizing general

To design a disk solution, complete the following steps:

- Determine the needed disk space.
- Select a RAID configuration which delivers the correct level of data protection and operates with the acceptable level of performance. Ensure the disk solution can handle sufficient IOs.
- Ensure the disks can be accessed by multiple IO Paths for performance and redundancy reasons.
- Database logs are needed to restore the data. Separate database logs from the data onto a different physical disk.
- For disaster recovery reasons, replicate the data to disks on another location.
- Always create a data backup on a regular basis which can be restored rapidly.

Disk space

To determine the disk requirement is a complex activity, particularly for applications that have many tables and use complex data structures. Often, applications can be parameterized to adjust their functionality to the specific needs of customers, which affects the way data is stored on disk.

The type of business the customer operates in also has an influence on the data volumes. It makes a significant different whether the customer operates in a high volume commodity product market, or deals with low volume customized complex products.

Theoretically, it is possible to compose a model to calculate the exact disk requirements, but this becomes very complex. To feed the model, detailed customer information is needed. In the majority of customer cases this information is not available when the sizing is done, and collecting this information is a time-consuming activity. Therefore, Infor has decided to base the disk space requirements sizing on average values fine tuned based on experience and costs. Disk space becomes cheaper over time and makes a detailed analysis unnecessary.

History

The database will grow over time because new data will be constantly entered. A growing database has an impact on disk space and performance. With history, there are two main questions:

- What is the time to keep the data part of the active database before it will be archived?

  Historical data has an impact on performance. Growing tables have a negative impact on performance. It costs more system resources to retrieve and mutate data from large tables than small tables. Options are, if the application supports, to move certain data to history tables or having a dataset per period. The active database is normally located on high quality disks. You can consider locating historical data on less sophisticated and cheaper disks. Historical data is accessed less frequently and is backed up. The fast disks can be used as much as possible for the active part of the database.
  The time that data must be available online varies per business, and depends on company policy or government rules.

- How long should data be keep available online?

  After a period of time, there are no business requirements to keep data online. The data can be archived (see section ‘Archiving’, in Chapter 9), or removed/stored only as backup data.

Archiving

Archiving is the process to extract data from the live online database to a more offline environment, or to backup media. In section ‘History’, in Chapter 9, it is explained that historical data can have an impact on the performance and disk space. Actually, this is true for all large amounts of data.

Archiving helps to make the environment clean because old data will be removed; it also helps to keep the performance of the environment under control. Data retrieval and mutations cost more resources on large tables than small tables. Indexes become deeper, and table scans and range scans become heavier.

Archiving helps to keep an environment maintainable. All kinds of operations, such as backup and restore, building indexes, and determining table statistics can be done faster. Without archiving, these type of operations sometimes cannot be done due to time constraints because the database is too large.

Archiving will be used for these types of data:

- Volatile data:

Certain data, such as data used for production planning, is volatile. After a certain item is produced, the planning data will no longer be used.

- Data consolidation:

  Certain data, such as financial data, consists of a lot of details and can be consolidated into fewer data. Naturally, details are lost, but that is not always a problem.

- Historical data, see section ‘History’, in Chapter 9.

Disk I/O Throughput

Spindle size

The size of the disk spindles grows over time; disks with a size 200 GB are not uncommon. The speed of the disks and the number of IOs the disk can handle per unit of time has not increased with the same factor. This often leads to the calculated disk space fitting on one or two disks. Infor strongly advises you not to install a complete database on two or three disks. One single disk cannot handle the required number of IOs, and will block the overall performance. This makes sense because some years ago, for the same amount of data, five times more disks were needed with almost the same disk speed.

Number of disk spindles

Disk storage is one dimensional, and disk speed is another important dimension. In some situations, the data can fit on one disk, but the disk cannot handle all the data retrieval and mutation requests. The spindle size is not inline with the disk speed (explained in section ‘Spindle size’, in Chapter 9). On average, a modern disk can handle 100 IO operations per second. An IO does not always indicate that a lot of data is transferred; a small data transfer also causes an IO.

To deal with the number of IOs, it is advised that you spread the data across multiple disks. These disks can then work in parallel on the same data request. Overall, more IOs per unit of time can be handled. The various ways disks can be combined are explained in section ‘Raid’, in Chapter 9.

Note

- All Infor sizing guides advise a minimum amount of disk spindles; for performance reasons, it is strongly recommended you follow this advice.
- A database server which is used frequently should have at least 6 disks; for the exact details, see the specific sizing guide.

Disk controller

The disk controller is the hardware interface between the computer system and the disks/disk subsystem. The disk controller transfers the data to the disks and ensures that the data is written on the correct disks. There are various types of controllers such as straight forward controllers, fancy controllers, controllers which support hardware RAID, controllers with lots of cache, and fiber optical controllers.

For sizing and performance, it is important that a disk controller has sufficient bandwidth capacity to handle all traffic to the connected disks. Too many disks connected to one controller can cause a performance bottleneck. Check with the vendor how many disks can be connected to one controller.

Disk cache

Disks are slow devices compared to the other components in a computer system. If the processor must wait until an IO is finished, the overall system throughput degrades. Having a cache in front of the disks can improve the performance and throughput. This size of the cache determines the overall disk throughput, especially for random IO. Note that caches will not help in situations with a large amount of IO operations. For example, if a large piece of data must be written, the cache will quickly be filled with new data. Data must be flushed to disk before you continue to write data to the cache; in this situation, the speed of the IO operation iterates to the IO speed of the disk, because you don’t benefit from the cache because all data is new. The writer process cannot write before the cache is emptied.

Disk caches can be implemented on the controller or on the disks/disk cabinet.

For more information about caching, see section ‘Caching’, in Chapter 11.

Raid

Originally, RAID stood for Redundant Array of Inexpensive Disks; now, it is known as Redundant Array of Independent Disks. RAID stores redundant information about data. The redundant information enables data regeneration in case one of the disks fail.

Overview of Raid level definitions

RAID Level	Description
Level 0	Disk striping without any data protection.
Level 1	Disk mirroring.
Level 2	Data protection based on Hamming code for error detection and correction. Data is striped on bit level. It is not used in practice.
Level 3	Is based on byte level striping with a dedicated parity disk. RAID 3 is very rare in practice.
Level 4	Data blocks are distributed as with striping. Dedicated parity disk reserved.
Level 5	Data blocks are distributed as with striping. Parity data is distributed across all disks within the array.
Level 6	RAID 6 extends RAID 5 by adding an independently computed check data.

Table 9-1 Various RAID levels

The various RAID levels are not all used in practice. A RAID selection is always a compromise between disk space and performance overhead versus data recovery possibilities. Some levels have a better cost/benefit ratio. In practice, many RAID level combinations are used; the most common are explained below:

RAID 0

In a RAID 0 configuration, the data is split into small pieces and spread across the various disks. Raid 0 is optimized for speed. If large pieces of data are read or written, all disks in the configuration can work parallel on the same IO request. This means the IO request can be handled faster. The size of the blocks is called the stripe size. The optimal stripe size relies on the character of the work load. For OLTP type of workload, a smaller stripe size is more beneficial. Batch type environments benefit from larger strip sizes. A typical stripe size for OLTP is 64 Kbyte.

There is no data reserved for parity information, which means RAID 0 is not a very reliable solution. If one disk in the configuration fails, all data is lost.

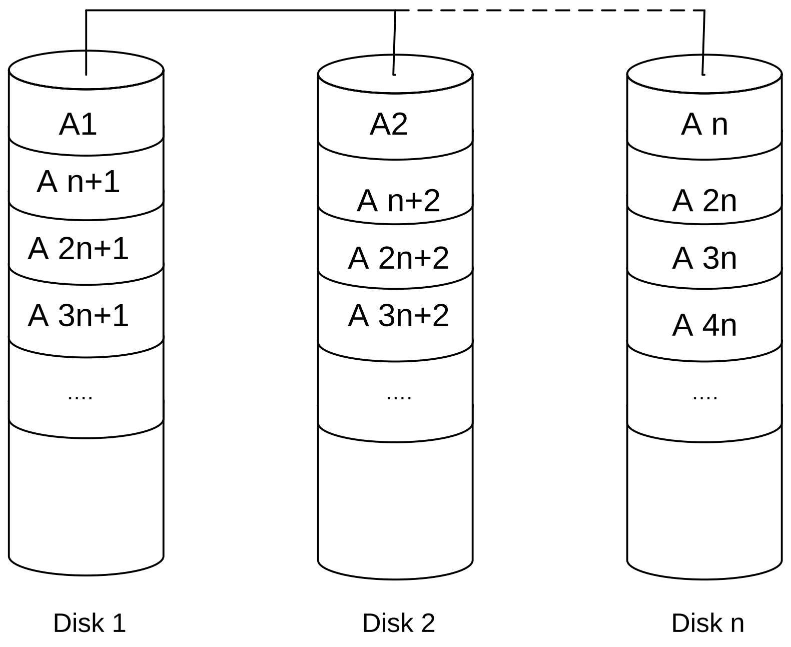

Figure 9-1 RAID 0

RAID 1

RAID 1 stands for disk mirroring. A RAID 1 configuration consists of two or more disks. The data is copied exactly to another disk or disk set. This means that only a maximum of half the total available capacity can be used.

RAID 1 can be beneficial for read intensive applications. In case of data reading, the disk that finished the read operation first presents the data. Write operations can last longer. Before a write operation is finished, the write operation to all disks must be finished because this increases the average write operation time.

From an availability point of view, this is a reliable solution.

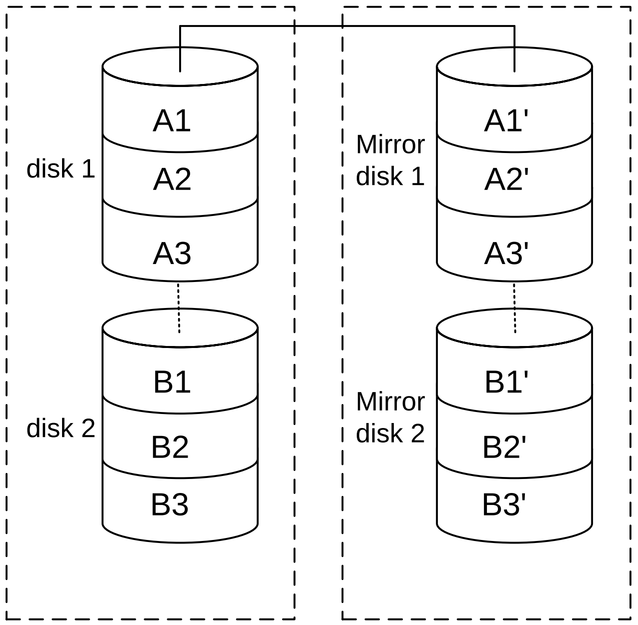

Figure 9-2   RAID 1

RAID 5

Raid 5 uses striping combined with parity data. Every time a data block is written, a parity block will be written for the same stripe set. From a storage point of view, RAID 5 is a reliable solution with low cost overheads, which is why RAID 5 is very popular. From a performance point of view, Raid 5 is not optimal. For every write operation, the parity block must be calculated and updated. For IO intensive environments, RAID 5 is not recommended.

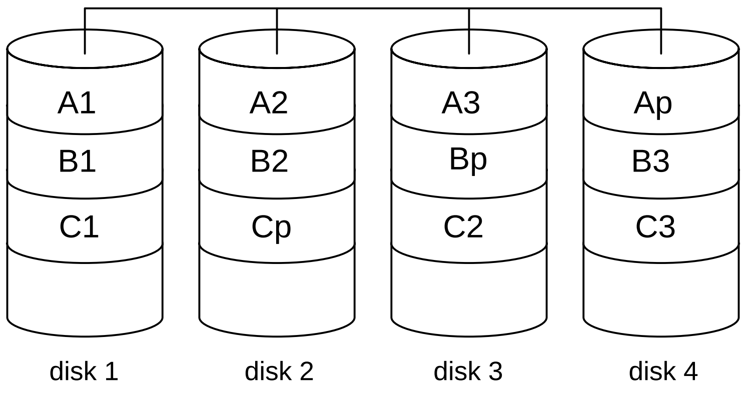

Figure 9-3 RAID 5

RAID 0/1, RAID 01, RAID 0+1

This RAID level combines RAID level 0 and RAID level 1. RAID 0/1 is a mirror of stripes. It has the speed benefits from mirroring and striping and the data reliability of mirroring.

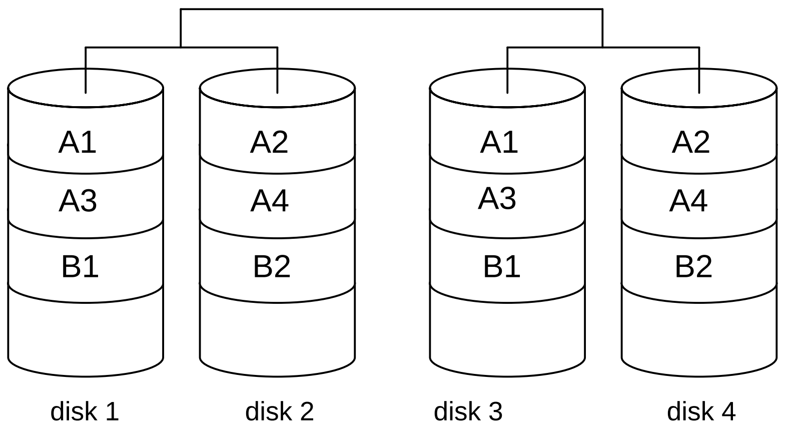

Figure 9-4 RAID 0+1

RAID 1/0, RAID 10, RAID 1+0

This RAID level combines RAID level 0 and RAID level 1. RAID 0/1 is a stripe of mirrors. It has the speed benefits from mirroring and striping and the data reliability of mirroring.

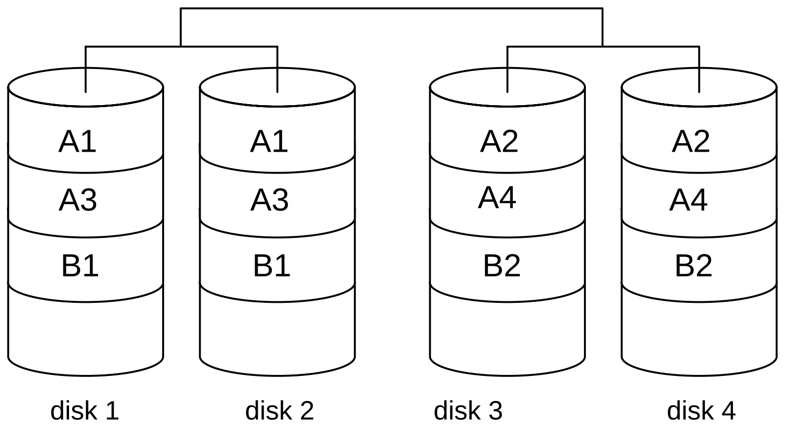

Figure 9-5 RAID 10

Software RAID

The various RAID levels described in the previous section can be implemented in software and hardware. In software, the server CPU does the necessary processing for the RAID functionality which introduces some overheads; fewer cycles can be spent for carrying out application functionality. The overheads vary per platform. Infor does not recommend software raid in general (see also ‘Hardware RAID’, in Chapter 9).

Hardware RAID

In hardware, the RAID functionality is implemented in hardware on the controller; which introduces less overhead for the server. Infor recommends hardware RAID above software RAID.

SAN/NAS

Definitions

The difference between SAN (Storage Attached Network) and NAS (Network Attached Storage) is not as clear as it was. The two fundamental different storage architectures are changed and they start to overlap functionality wise.

SAN

A SAN is a network designed to attach devices such as disks and tapes to a server. A SAN has its own management layer, which takes care of the security, storage, and connections. For the server, SAN storage looks like local storage. The communication protocol with a SAN is very low level and similar to SCSI or ATA for local disks. The disk access is on raw block level than file level. They are often connected to the server by a high-speed fiber connection.

NAS

NAS is a storage sub-system with one of more hard disks. A server can connect to a NAS using a network which is frequently TCP/IP based. Users or software communicates with a NAS on file protocol level. Examples of protocols are the NFS of CIFS Common Internet File Systems.

Advantages and Disadvantages

Speed

A SAN is usually faster than a NAS. The file system protocols based on TCP/IP create much more overheads than the latest SCSI protocols on top of a dedicated fiber. The I/O throughput between a single server and a SAN can be higher than the I/O throughput between a single server and a NAS.

This is true in the high-end, but note that the maximum throughput of a NAS is already sufficient for the majority of the applications. The difference in speed means that SANs, from origin, are frequently used in intensive database environments and NAS is used in fileserver type environments.

Multi-host file access

Multi-file access in a SAN environment is more difficult than in a NAS environment.

To update the same file on a SAN, a cluster based file system must be implemented; the newer SANs are improved to deal with this. NAS does not have any issues with this.

Complexity

SANs are usually more difficult to work with. The fiber channel, HBAs, and switches introduce additional complexity.

Backup

A backup to tape is more intensive in a NAS environment than in a SAN environment. Backups usually deal with large amounts of data. Due to the low-level protocol in SAN environments, a backup in a SAN environment can be performed much more efficiently; the same is true for backup recovery.

Image backup

Many SAN solutions offer image backup functionality to create fast backups, but many NAS systems do not. However, NAS evolves to SAN in this area.

Disk sizing formulas

The memory formulas used in the various sizing guides are based on the following format:

$I_{disk} = I_{base} + (2 * I_{tot}) + H * (n * C)$

Whereby the following is applied:

I_{disk} = Total calculated disk space.

I_{base} = Base amount of disk space.

I_{tot} = Internal memory.

n = Number of **concurrent** or **named** users.

C = Amount of disk space per user.

H = Number of years history.

I_{base} includes all the static memory components such as operating system, and database and application installation software. This value also includes temporary working space.

The factor $2 * I_{tot}$ is used to calculate the disk space needed for paging (see section ‘Paging and swapping’, in Chapter 8). On systems with large amounts of internal memory, you can choose to reserve less space for paging. Infor advises you to reserve the recommended values. Having insufficient paging space can lead to losing data. Because disk space is rather cheap, it is not worth taking the risk.

The number of **named** or **concurrent** users ‘*n*’ is a key input value. It is important to understand the differences between the two user types, which are explained in section ‘Various user types’, in Chapter 6.

C represents the dynamic data growth per user. It is an estimated factor based on measurements and experience. See also section ‘Disk space’, in Chapter 9.

H represents the number of years historical data has to be stored.

Chapter 10 Network sizing

10

Introduction

Systems operate in a more integrated way. User interfaces are more complex and graphical instead of character based. Solutions are more central than decentral (see section ‘Central versus decentralized solution’, in Chapter 6). Looking to the internet evolution, there is now more functionality which needs network capacity, which means that network sizing is still important.

Network sizing can be an extensive job because a lot of knowledge is required. In this whitepaper, only the crucial network capacity aspects from an application point of view are covered.

Network capacity sizing

The network capacity or network requirements can be characterized by network bandwidth and network latency. Whatever network infrastructure is build, the capacity can be described as a function of these two variables.

Network bandwidth

Network bandwidth is the amount of data that can be transferred over a communication link per unit of time. The measurement unit is usually bits per second.

The following example gives you a feeling of the bandwidth impact:

Example

A file of 1 MByte is transmitted. 10 bits are needed to transfer 1 data byte. The additional bits form the communication overhead. The package size is 1500 bits.

Line speed	Network card delay [ms]	Calculation	Response time [ms]
10 Mbit	0	1000*1024* 10 bits / 10*1000*1024	1000
100 Mbit	0	1000*1024* 10 bits / 100*1000*1024	100
1 Gbit	0	1000*1024* 10 bits / 1000*1000*1024	10

Table 10-1 Network example 1

The above example shows the bandwidth has a significant impact on the total transmission time. In the low-end, the absolute response times can differ significantly. In this case, the end user will experience the difference between 10 Mbit and 100Mbit. The difference between 100Mbit and 1 Gbit is also noticeable and is relative the same ratio as between the 10 Mbit and 100Mbit, but is as absolute response time and user experience less significant.

For all applications which require an LAN, Infor advises the highest speed possible.

Network latency

Network latency has a strong relationship with the network response times, but is not completely the same. The following example shows the impact of various line speeds in relation to response times. The example is based on the same assumption as the example in section 'Network bandwidth'.

Line speed	Network card delay [ms]	**Calculation**			**Response time [s]**
		Bandwidth delay [ms]	#packages	Total network card delay	
10 Mbit	0.5	1000*1024* 10 bits / 10*1000*1024 = 1 sec	(1000*1024* 10 bits)/1500 = 6827	6827 * 0.5 ms = 3.4 sec	3.4 + 1 = 4.4
100 Mbit	0.5	1000*1024* 10 bits / 100*1000*1024 = 0.1 sec	6827	6827 * 0.5 ms = 3.4 sec	3.4 + 0.1 = 3.5
1 Gbit	0.5	1000*1024* 10 bits / 1000*1000*1024 = 0.01	6827	6827 * 0.5 ms = 3.4 sec	3.4 + 0.01 = 3.41

Table 10-2 Network example 2

The Network delay mentioned in this example is a form of **network latency**.

Latency is the time needed to transfer a data package from A to B. Network latency can be caused by the following:

- Media Access: For example, wait for silence/collision recovery.
- Media Speed: Bits per second on the wire.
- Transmission Delay: Distance, media velocity of propagation, and repeater delay.
- Switching: Time for router or switch to decide, filter processing and cut through latency of switch.
- Process: Time for a host to process request and reply.
- Queuing: Time spent behind other packets waiting transmit.

In WAN environments, latency plays an important role.

The latency appears for every data package. Latency has relatively less impact on large data packages than on small packages.

Bundling network packages can be beneficial in high latency network environments.

The network latency is one of the root causes why page oriented applications perform better than field level oriented applications.

Sizing formulas

The Infor network sizing formulas used in the various sizing guides are based on the following format:

$I_B = n * I_u$

Whereby the following is applied:

I_B = Total calculated bandwidth.

n = Number of **concurrent** users.

I_u = Bandwidth requirements per user.

Many Infor sizing guides specify minimum bandwidth requirements per connection line. If I_B is smaller, then the minimum specified requirements go with the minimum requirements; it is highly recommended to follow this recommendation. Even if the bandwidth based on the calculation is not needed, the bandwidth has an impact on the transfer time. If the line speed is too low, the transfer from A to B takes too long for an acceptable performance.

The maximum or recommended latency is specified in the sizing guides, and it is highly recommended to follow this recommendation. Lines with higher latencies than advised can cause unacceptable end user response times.

Network sizing considerations

The topics mentioned in this paragraph impact the network sizing and need consideration.

Network compression

All network figures published in the Infor sizing guide are measured without data compression, unless mentioned. In practice, there can be data compression on various levels, and data can be compressed in routers or on application levels. Certain web servers can switch packet compression on or off. It is always an exchange because compression will cost processing power, but it will save network capacity.

Compression also reduces the response times, especially on the low-speed networks.

Compression can reduce networking costs because the network requirements are reduced. It is worthwhile investigating the compression possibilities.

Citrix/Microsoft Terminal server

Terminal server is a solution where client software which normally runs on the local client PC, runs on a server. Only the screen output is showed to the users. Terminal server can reduce network traffic significantly because only the screen changes are sent to the client PC. Citrix/Microsoft Terminal Server offers many advantages which are out of reach of network requirements.

Chapter 11 Background information necessary for sizing

11

To carry out a sizing thoroughly, detailed knowledge of various sizing related technical subjects is crucial. For example, you must understand how the applications behave on the various computer systems, architectures and operating systems. Background knowledge of how the application architecture is developed and how the application can be configured, helps to make the right decisions. Various sizing related important subjects are explained. This chapter is intended for the technical people involved in the sizing projects.

Computer architecture background

This section provides some background information about various computer architectures, but not detailed extensive theoretical background information about all possible architectures. Many books are written about this subject. This document stays high level, and only the architecture aspects which have a known impact on the sizing are described.

Symmetric Multiprocessing (SMP)

SMP is a computer architecture where two or more processors connect to the same main memory. All processors are connected to the same system bus, (see Figure 11-1   SMP architecture). Symmetric in this context means that processors are treated equally.

SMP is the most common multiprocessor architecture for the low-end and midrange systems. For high-end systems, other computer architectures such as NUMA and crossbar architectures are used. All processor-to-memory traffic goes through the same system bus in an SMP structure. If there are many processors, this system bus can become a bottleneck because there is too much traffic to handle; in this situation the scaling deteriorates. For more information about system scaling, see section ‘System scaling’ and ‘Server scaling in relation to sizing’, in Chapter 11.

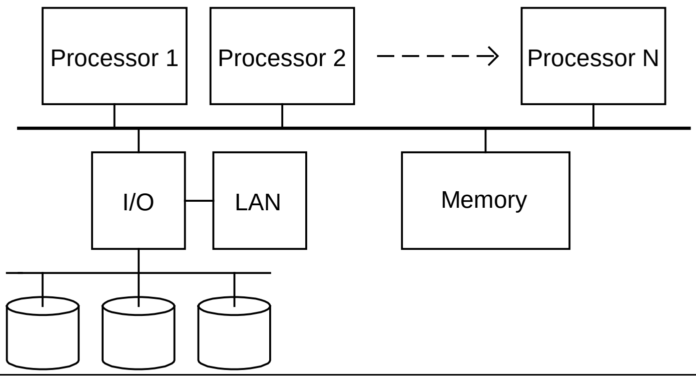

Figure 11-1 SMP architecture

It is difficult to specify how many processors the SMP architecture scales. Firstly, it is application dependent. An application which does a lot of memory access and does not use the cache effectively puts a higher load on the system bus compared to applications which make effective use of the cache. For more information about caching, see section ‘Caching’, in Chapter 11. The bus speed can vary per hardware model. Newer models work with higher bus speed. More traffic on the system bus can be handled per unit of time, but newer models are usually equipped with faster processors and memory. Faster processors and memory are more intensive for the system bus. In general, scale modern systems till eight processors without any problem; above eight processors the scaling flattens. For the majority of customers, eight processors are sufficient. Most hardware vendors have scaling information available for the hardware models they deliver and it is advised that you ask them for this information.

The hardware tables in the sizing guides also provide scaling information for that particular Infor application.

NUMA (Non Uniform Memory Access)

In section ‘Symmetric Multiprocessing (SMP)’, in Chapter 11, it is explained that SMP has certain limitations with regards to scaling. NUMA is developed to reach high-end system scaling and solve the SMP scaling issue. Figure 11-2 NUMA architecture gives an overview of the NUMA architecture. NUMA architecture consists of groups of processors. The various hardware vendors have names for these groups such as quad, cell-board, and CEC. The distinguishing characteristic of the NUMA architecture is that groups are connected to each other through a high latency, relatively low speed bus. Each group of processors has its own memory and IO channel. The processors within the group are connected to each other using a low latency, relatively high speed bus. This makes it possible for all processors in the configuration to access all memory within the configuration. The bus which connects the cell-board has higher latency due to the longer distance.The latency of the local memory bus is much lower than the latency of standard SMP systems. The advantage over the SMP architecture is that this architecture has no bandwidth limitation on the local system bus. The high latency bus that the processor groups are connected with has a slightly higher latency than the standard SMP bus. From this perspective, NUMA has no improvements over SMP; the idea is that the majority of the requests are processed locally. This implies that the software operating system, database and application must be NUMA aware to work with this concept. In Figure 11-2 NUMA architecture, there is a cache between the controller and the local bus. This is to compensate for the high latency of the processor board interconnect.

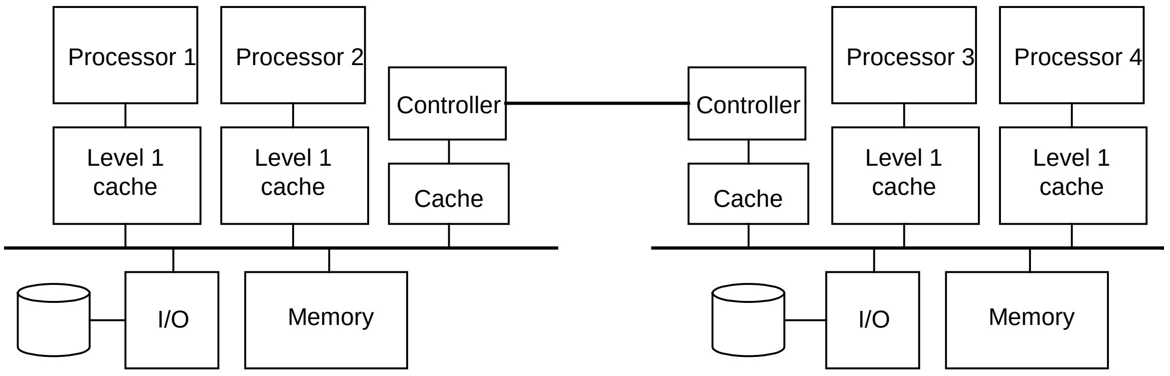

Figure 11-2 NUMA architecture

Parallel architectures

In the context of this document, a parallel architecture is a cluster of independent running servers. A cluster can be used for scalability/high availability. Examples are as follows:

- Three-tier configuration with multiple application servers.
- Database cluster running a parallel database.
- Two-tier configuration running in a cluster with cold standby.

There are various parallel computer architectures, and parallelization can be reached on multiple levels. For example, SMP and NUMA (see section ‘Symmetric Multiprocessing (SMP)’ and ‘NUMA’, in Chapter 11) are variants of a parallel architecture, because there can be multiple processing streams at the same time.

Usually, reaching high availability goals is not a problem. You must ensure the application software supports the high availability options and that the software is configured well. In many situations it is not possible to benefit from the scalability of a cluster, and even when it works from a functionality point of view the scaling can be very poor. There can be hotspots in the data, such as data which cannot be shared among multiple servers.

An example of when parallelization works quite well is a web farm with multiple web servers. If the software is stateless, adding an additional server will contribute to the overall capacity and scalability. However, not all applications are stateless and designed to be carried out in parallel.

Databases are often used in a parallel mode, running on a cluster of servers. If one server in the cluster breaks, the other servers take over the functionality of that server and cluster and carry on with its operation. From a scalability point of view, these type of solutions work perfectly in read intensive environments. In environments with a lot data mutation, the scaling can be poor, especially when the data working set is small. Unless the database is based on the ‘shared nothing’ architecture, all memory sections and caches of the servers in the cluster must be updated in case of data mutations and data locks. It introduces a lot of overheads to keep all server caches in sync.

The previous examples show that you must be careful with parallel solutions. Before you start using these, you should investigate thoroughly and ask whether it is used for scalability or high availability. In all situations, it is advised that you carefully test and check whether the solution reaches the expected goals.

Caching

Caches are used to optimize network, memory, and disk access; every server makes use of caches because it contributes significantly to the overall system performance. Certain Infor applications benefit from caching. Therefore, it is important to understand the basics of caching and to explain why certain applications benefit more than others. The following examples show caching in combination with a processor. The principles are the same for disk caches and network caches.

Figure 11-3 shows memory connected to a processor. A processor works at a higher speed and operates much faster than memory, and must interact with the memory frequently. The speed difference can be more than a factor ten, which means the processor must wait after every memory access, which subsequently degrades the overall system throughput. To increase the memory access speed, a cache can be placed between the processor and the memory (see Figure 11-4). A cache is a small piece of memory which works on a much higher speed than the main memory. The frequently used data is copied from the main memory to the cache. If the processor needs to work with the data, the data is read from the cache and retrieved much faster now. The processor can continue its work with less delay.

There are various statistical algorithms used to keep the cache up-to-date. Keeping caches updated is quite complex, especially in multi CPU and NUMA environments.

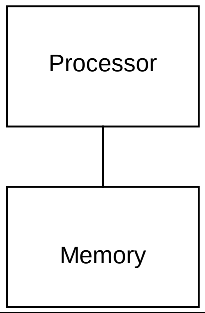

Figure 11-3 Memory architecture

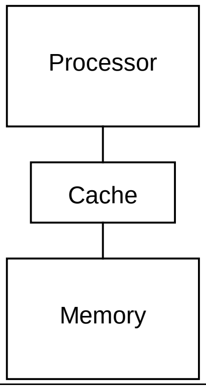

Figure 11-4 Memory cache

There can be multiple level caches to increase speed (see Figure 11-5). In this case, there is a level 1 and a level 2 cache implemented. There are even systems which have level 3 and level 4 caches. The higher the level of the cache, the larger the size is. Level 1 cache is often part of the processor and is implemented on the same chip.

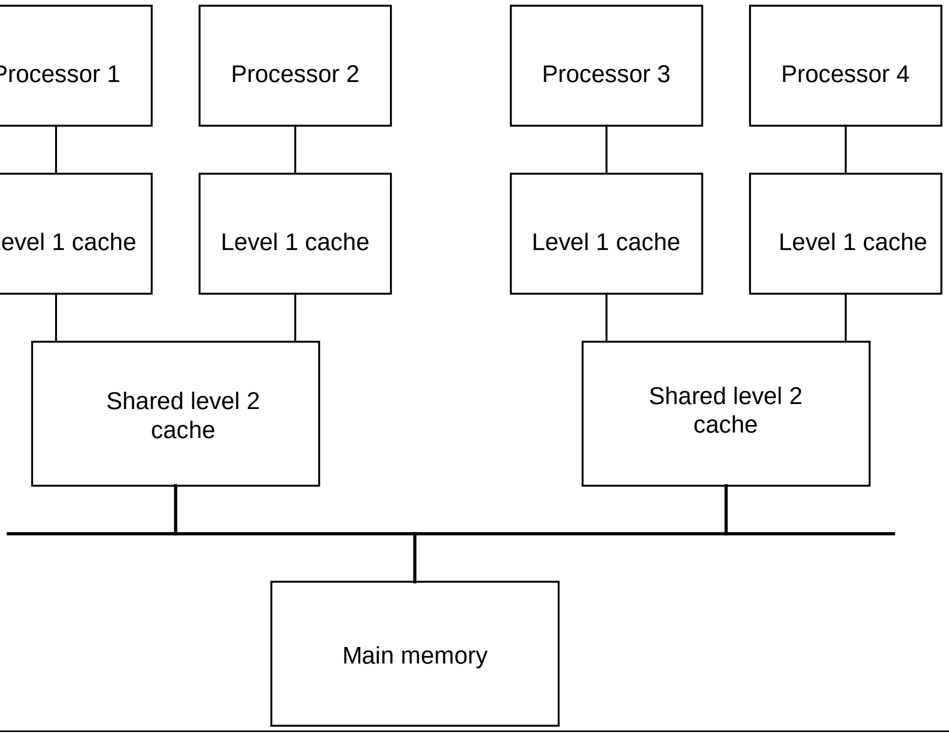

Figure 11-5 Multi level caches

CPU architectures

Previously, there was a processor with just a single chip with one processing unit. However, several new architectures have been developed, such as multi-core processors, hyper-threading, and hardware threading. In addition to this, combinations of these architectures are possible.

Multi core

A multicore processor consists of two or more independent processors on one die. The cores share the L2 cache. For an architectural overview, see Figure 11-6. Having multiple CPUs on one die has some advantage over single core processors, such as:

- The cache can be brought in sink on a much higher speed when they are on the same chip.
- Multi core CPU takes up less space in systems.
- The overall power consumption is lower than single-core processors. The communication between cores operates on a lower voltage.

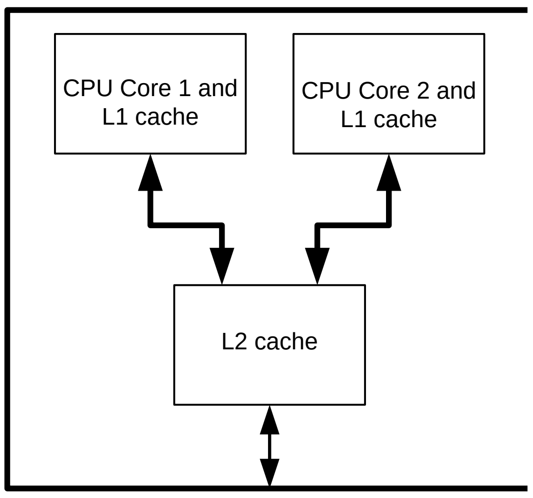

Figure 11-6 Dual core chip

Multi-core has some disadvantages, such as:

- Multi-core chips run on a relatively lower clock speed than single-core processors. Single-threaded applications can run slower (see also section ‘Infor LN: Single bshell versus parallel bshell’ and ‘Single threaded versus multithreaded software’, in Chapter 11).

Hyper threading

Hyper-threading is an Intel specific technology that improves processor processing capacity. With this technology, the units in the processor that store the processes' state are duplicated. However, the processing units are not duplicated. The benefit of hyper-threaded CPUs is that they can switch between processes relatively easy. If the CPU cannot continue, for example because of a cache miss, the CPU can easily continue with another process because all required registers are already loaded.

To the operating system, the various threads are presented as multiple CPUs. According to Intel, a hyper-threaded CPU shows a performance gain of 30 percent compared to a non-hyper threaded CPU. However, the performance gain depends on the application because some applications benefit more from hyper-threading than others.

Hyper-threading can increase the overall processing capacity. Be aware that the performance can slow down under certain circumstances. Single-threaded processes, such as batches and server processes of databases, can slow down significantly because the processing speed of an individual thread is lower than the speed of the CPU without hyper-threading.

As a rule, hyper-treading increases the overall capacity with a factor of 30 percent, but the speed of the individual threads is 65 percent of the performance of a non-hyper threaded CPU.

IBM iSeries characteristics

The performance characteristics of the IBM iSeries are very specific and differ from other operating systems such as Microsoft Windows and UNIX®. To carry out a correct sizing, you must understand the iSeries performance characteristics. This section provides background information on the IBM iSeries performance characteristics and is limited to the general behavior important for sizing.

IBM has multiple iSeries models with different capacities. The number of CPUs, the CPU speed, and the model's features determine the iSeries capacity. The load is expressed in Commercial Performance Workload (CPW). Two types of CPWs are batch and interactive (see Figure 5-3). Interactive CPWs handle the OLTP transactions and transactions that need immediate response. Batch CPW capacity handles batches, semi-batches, reporting, and low-priority activities.

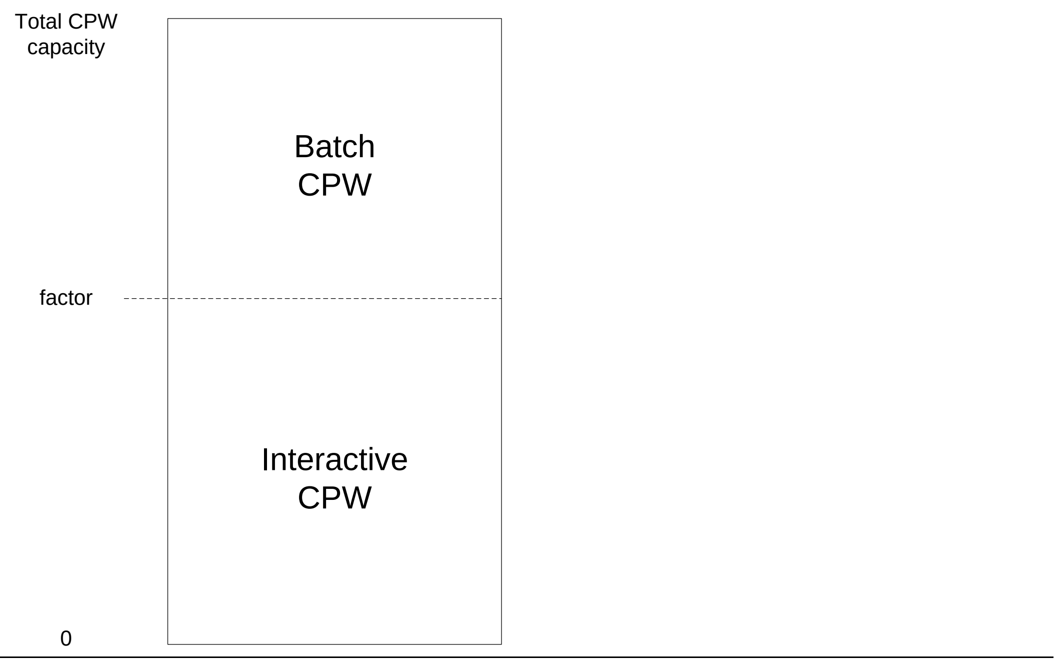

Figure 11-7 Distribution of Batch versus Interactive CPW capacity system wide

A customer who orders a system can select either of these options:

- Enterprise systems
- Non-enterprise systems

With a non-enterprise system, the amount of interactive CPW capacity is pre-determined and fixed to a certain percent of the available total capacity. The capacity depends on the system. Within an enterprise system, the ratio between interactive and batch CPWs can vary depending on the load characteristic at a given time. The amount of interactive CPW capacity can vary from zero to the total CPW capacity of the system.

You must determine the required batch/interactive CPWs because a significant price difference exists between the two CPW types.

The application software determines whether a program or transaction uses the interactive or batch CPW capacity. Typically, programs use a mix of batch and OLTP CPW capacity (see Figure 11-7). Certain software allows a task to be carried out in either interactive or batch mode.

CPU Clock speed

The clock frequency of a CPU is not the same as the absolute processing power of the CPU. To a certain extent, a higher clock speed increases the processing capacity, but the overall performance improvement also depends on the other components in the system. A CPU does not operate on its own. This will explained in the next paragraphs.

Clock speed same CPU family

If two CPUs within the same processor family are compared, please note that the delta in clock speed is not equal to the delta in processing power increase. The CPU is not the only component in a computer system. The components around the CPU, such as memory, disk and network controllers, are the same. When the process is being carried out, the CPU and other components are used in a mix. The clock speed increase is only seen back on the portion of the processing time the CPU is active for a process. This means that percentage wise, the overall increase in processing power is lower than the increase in clock speed.

Typically, a CPU intensive application with low IO rate will benefit the most from a clock speed increase.

A general rule is that the relative increase in clock speed is seen back for 50 percent in the overall performance.

Clock speed comparison between different CPUs

You cannot compare the clock speed between CPU architectures. One CPU of a certain brand operating on a speed of 2.0 GHz can be faster then another CPU of a different brand operating on higher clock frequency. Certain CPUs can have a more complex instruction set compared others. With a complex instruction set, fewer instructions must be carried out to run the same functionality. Some CPUs have multiple pipelines to carry out instructions in parallel, while others have one pipeline. This means that CPUs can be compared based on their application benchmarks.

System scaling

Servers are equipped with multiple CPUs to increase the overall system capacity. The best situation is that every CPU added to the configuration contributes 100 percent to the overall capacity. For example, going from one processor to two processors, the capacity should almost double; going from two to four, the capacity should double again. In practice, this is not the case. The more CPUs added to a configuration, the less a CPU contributes percentage wise to the overall capacity of a server. This phenomenon is called the scaling of a machine. How the scaling curve looks like depends on many factors (see section ‘Server scaling in relation to sizing’, in Chapter 11). The hardware vendors have information about how their systems scale, but note that it is not only the system which determines the complete scaling, but also the combination, the complete solution, including the operating system, database and application software. Scaling behavior depends on the amount of shared resources. A shared resource causes queuing and flattens the scaling.

The complete solution can, per definition, never scale better than the scaling published by the hardware vendor. Figure 11-8 shows the increase of capacity flattens when CPUs are added.

Figure 11-8 Scaling curve

The following formula can be used to approach and calculate the scalability of a system:

$$p_n = p_1 \times (1 + s + s^2 + ... + s^{n-1})$$

Whereby the following applies:

- p_n = performance with n CPUs
- p_1 = performance with 1 CPU
- s = scalability factor = $(p_2 - p_1) / p_1$

Server scaling in relation to sizing

System scaling is an important sizing subject. For example, if a customer wants to double their processing capacity, the number of CPUs must be multiplied with a factor higher than two due to the scaling effect.

Various areas which can have an influence on the system scaling are mentioned in this section. However, the list is not complete because there are more factors. For more background information about this subject, see specific literature.

Computer architecture

At the start of this chapter, various computer architectures are explained as references for the necessary background information.

Sizing impact on the various applications

Infor LN

Infor LN is installed on various computer architectures at the customer site. The layered, single threaded process oriented architecture (see architecture chapter Infor LN sizing guide) means that Infor LN is memory intensive and has a big memory footprint. Therefore, Infor LN benefits from a larger cache size more than other applications.

Infor LN makes heavy use of application shared memory and, in general, uses memory quite intensely. On SMP machines, all CPUs are connected to the same memory. Accessing shared memory works efficiently on SMP machines; this way, Infor LN works best on SMP machines.

The system scaling depends on the systembus capacity; a high bus bandwidth improves the system scaling.

In general, the Infor LN scaling benchmarks follow the scaling of the server published by the hardware vendors, which indicates optimal application scaling.

Infor LN works on NUMA systems. As explained earlier, NUMA is developed for high-end scaling. The problem is that applications must be NUMA aware. Many applications, such as Infor LN are not. When the application instance is started, the shared memory keeps running on the node where the application instance is initialized. Users will start their application process at random spread over all nodes. If a user is started on node2 and must access shared memory on node1, all communication goes through the interconnect bus because this flattens the scaling. On NUMA systems it is important to work as much as possible locally on a node, although this is not always possible. The newer NUMA systems offer features to influence this type of behavior and influence where processes are started.

When installing on NUMA, take the following architecture behavior into account:

Examples:

- If multiple BSE environments are used, ensure the users using these environments start their processes on the same node as the application instance is running.
- If there is one main BSE environment with many users and the capacity of one node is not sufficient, install the database and Infor LN software each on a dedicated node.
- The operating system tools deliver tools to optimize the environment, such as tools to bind the processes to a certain node. Ensure you make the best use of these tools.

Operating system

Infor LN is supported on many operating systems. The Infor LN architecture is a single threaded process oriented architecture. Each user has a bshell, database driver, and a database backend running, if applicable. In total, there will be many processes on the system. Unix can deal with large amounts of processes better than Microsoft Windows because windows is designed to work with threads. However, the latest Windows versions closely approach UNIX on this subject.

Many benchmarks are carried out on the various operating systems with predecessor versions of Infor LN. There are no major scaling bottlenecks known.

Benchmarks have proven that Microsoft Windows scales very well until four CPUs, but above this, the scaling is very flat. In many cases, this has to do with the hardware design because the used chipsets are not designed for more than four CPUs.

Larger servers are based on a NUMA type of architecture, and Windows is not designed for these types of architectures; the same is true for Infor LN. The combination of Windows/Infor LN does not benefit sufficiently from the NUMA scalability. The solution scales with a scaling factor not more than 1.3, and this solution does not have a competitive price/performance ratio. Infor advices above the four CPUs/eight cores a 3-tier architecture, rather than going above the four CPUs/eight cores on 2-tier on Windows.

IO and network resources

The IO throughput of a server can have a big influence on the system scaling. The IO capacity in the system must be in balance with the CPU capacity in the system. A system with large CPU capacity can serve a lot of requests which are read or written to disk. If the IO subsystem cannot catch up with the CPU, the system throughput stalls and gives a bad scaling curve.

Sufficient spindles are necessary, and the data needs to be well balanced across these spindles to reach the required level IO throughput.

Caching can optimize IO throughput (see section ‘Caching’, in Chapter 11. IO controllers with cache are advised.

Database/Data setup

It is important for optimal scaling that the database is well tuned. The database performance is crucial for the total configuration. Other layers, such as the application, cannot continue if the database blocks because the total solution will not scale.

Database locking is an important scaling bottleneck. With the set-up of the data, try to prevent locking hotspots as much as possible.

Ensure the database runs with sufficient memory and the IO is well balanced.

Single threaded versus multithreaded software

Programs will generally be carried out by the operating system as threads or as processes, depending on how a program is developed. A program must be designed to be able to run multi-threaded; if not, the program will run as a process. If an application supports multi-threading, a process can split itself into one or more threads when the program is being carried out.

The difference between processes and a thread is that processes carry state information, have separate address spaces, and can only interact through the inter-process communication mechanisms of the operating system. Threads have much more sharing and data can be shared. It takes fewer overheads for the CPU to switch between threads than processes. Process oriented programs are frequently called single threaded applications.

From a sizing point of view, it is important to know that single-threaded applications cannot claim more than one CPU per process during the time it is being carried out. A multi-threaded process can claim multiple CPUs simultaneously.

Generally, multi-treaded applications deal more efficiently with the internal memory.

Two-tier versus three-tier

The choice between two-tier and three-tier is an important choice to make during the sizing process, and both solutions have advantages and disadvantages. It is difficult to say which solution fits the best because it depends on the situation of the customer and the IT-policy within the company. The weight factors of the advantages and disadvantages are not the same for each customer. Therefore, in every situation, all factors must be reviewed before a choice is made. Infor does not have a real preference, as both solutions are valid and supported.

This chapter gives an overview of the most important areas when deciding between a two-tier and three-tier solution.

Definition of two and three-tier within Infor LN

Within Infor LN terminology, a two-tier configuration is a configuration whereby the application and database runs on the same server. In a tree-tier configuration, the application and database run on a separate server. The user interface is not taken into account in this definition, but it can introduce an additional tier in some configurations. For more background information, see the Infor LN sizing guide.

Selection considerations

Hardware costs

In the two-tier situation, all involved software runs on one system. In the three-tier situation, the functionality is spread across multiple servers. Therefore, the two-tier systems can be significantly larger than the individual size of the systems in a three-tier configuration. Hardware costs are not linear. Larger servers are relatively more expensive compared to smaller boxes because, in larger servers, all kinds of features and communication busses must be built to let the CPUs and memory work with each other and let the solution scale. The pricing is made visible in Figure 11-9 although the price can differ per vendor and per hardware line.

This non-linear pricing structure can mean it is cheaper to have a hardware solution consisting of multiple smaller boxes than a single, big server. This is only one aspect of the total costs; from a system management point of view, it is easier and cheaper to maintain a few boxes rather than many.

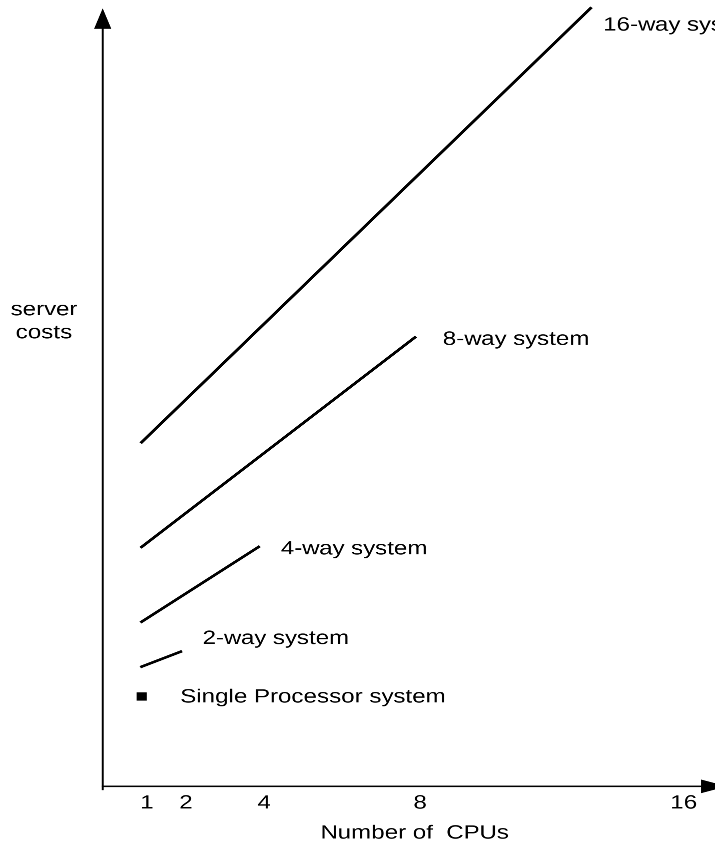

Figure 11-9 Server costs of SMP systems

Database license costs

Some databases base their license costs on the number of CPUs in the server the database runs on, so the more CPUs in the server, the higher the license costs. In a two-tier environment, a certain amount of the CPU capacity is reserved for the application. This capacity is not claimed by the database, but will be used to calculate the license costs. Moving to a three-tier environment, where the three-tier database server is smaller than a two-tier server, allows you to save on database license costs.

For the same reason, some customers divide their system into hardware partitions and run client server within one system.

Heterogeneous environments

With three-tier, you can build a heterogeneous configuration. The database server and the application servers run a different operating system such as a UNIX database server with Windows/Intel application servers.

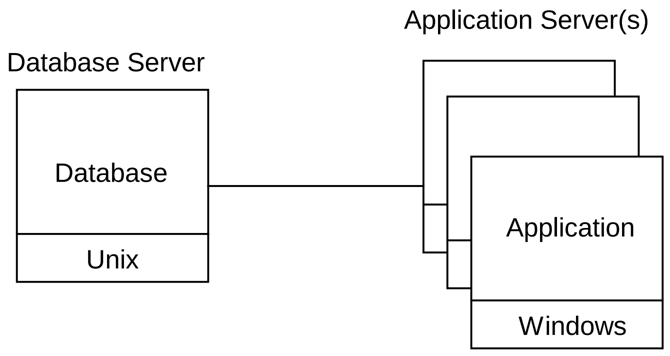

Figure 11-10 Heterogeneous environment

The database is a critical component in a configuration. Customers want to ensure the database is always up and running. Therefore, they want to run the database on a reliable high-end hardware. Running the application on the same high-end hardware can become an expensive solution; to save costs, offloading the application load to low-end cheaper servers is an option. The database and the application form the solution. If one component breaks, the complete solution does not work. Therefore, it is important that the application has the same degree of availability as the database. The application availability will be increased by having a low-end application server as a spare on standby.

High-availability cluster

Some customers use a three-tier configuration for high availability reasons. By definition, a three-tier configuration comprises two or more nodes. The nodes in the three-tier configuration are configured in such a way that they can easily take over functionality of the other nodes. The application server can take over the functionality of other application servers or the database server, or the other way around (see Figure 11-11 and Figure 11-12). High availability is a subject in itself, and will be covered in more detail in later versions of this document. The following figures show some possible scenarios. The systems in the high-availability cluster should have sufficient capacity to take over functionality of each other.

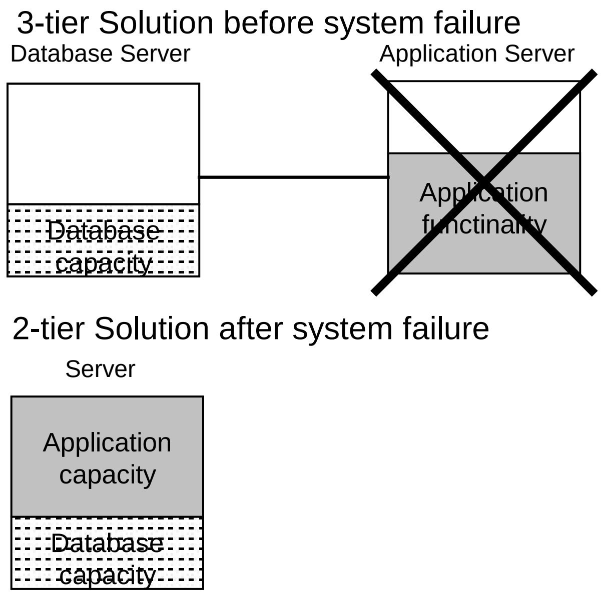
3-tier Solution before system failure
Database Server
Application Server
Database capacity
Application functionality
2-tier Solution after system failure
Server
Application capacity
Database capacity

Figure 11-11 Database server takes over application functionality

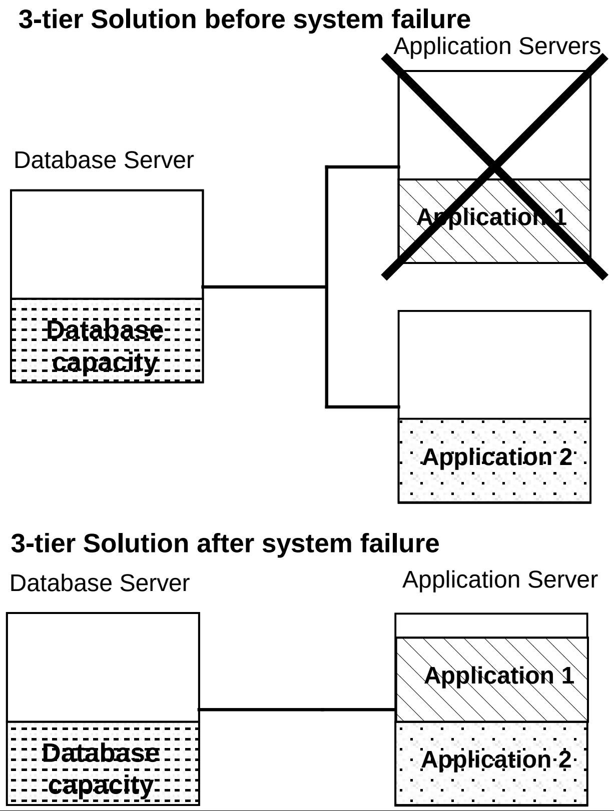

Figure 11-12 2nd application server takes over functionality of application server 1

Scalability

In general, the size of a database server determines the total number of users a certain configuration can support. All users must access the same dataset. When the database server runs out of capacity, this becomes impossible, and therefore the configuration reaches its limits. With two-tier, the database and application layer run on the same server. Both the application and database tier claim capacity from the system. With three-tier, the application functionality is offloaded to the application server; this gives room to serve more users before the database reaches its limit. This means that a three-tier configuration can scale up to the higher number of users compared to a two-tier configuration. Figure 11-13 shows the offloading to the application server.

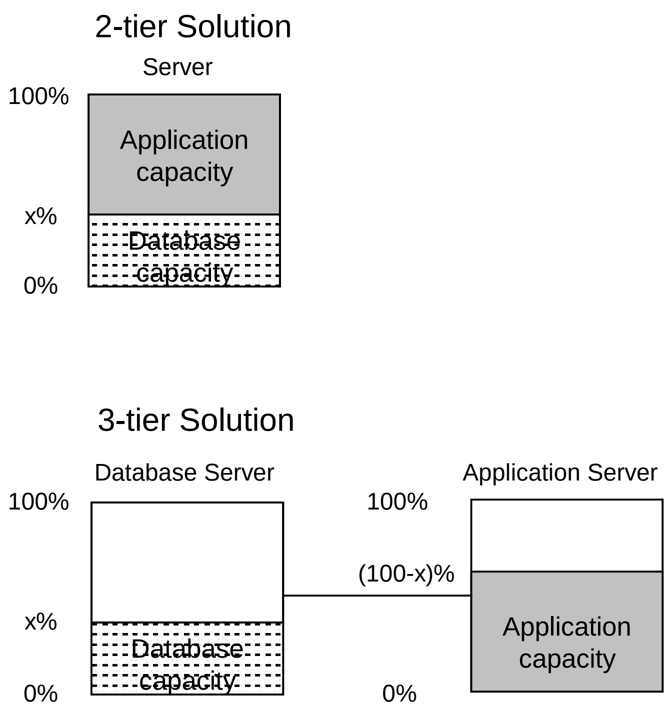

Figure 11-13 Capacity offloading

The number of application servers you can add to a three-tier configuration is unlimited for Infor LN. The highest number of application servers used in an Infor LN benchmark is 20. Technically, it was possible to add more, but they were not available.

The size and capacity of servers has grown significantly over the years. The speed of CPUs has increased, and also the number of CPUs a server can be equipped with. This means that the three-tier advantage, which is scalability, has become less relevant. In the UNIX and iSeries area this is almost no longer an issue, but in the Windows area, it is still a valid discussion. There are not many high-end scalable servers available in the market. Most servers have reasonable scalability up to four CPUs. For the larger customers, this limit will be reached soon. Above the four CPUs, three-tier is the most common solution direction.

Flexibility

One aspect to consider with three-tier solution is flexibility. It is rare that an implementation covers the complete company at once, because implementations generally go by site or group of sites. The timeframe can be multiple years. With a three-tier solution in this case, it is easier to let the hardware configuration grow overtime and buy capacity when necessary; this is also true for a two-tier solution because CPUs and memory can be bought when necessary. But you must start with a system that is able to be extended to the final capacity requirements. In the three-tier cases, you can always add additional application servers and, when the database server is too small, the database server can be replaced with a new one. The ‘old’ database server can be re-used as an application server.

Tuning

In the three-tier situation, a server has a dedicated task of database server or application server. Some claim that the server and software can be tuned better when running separately on a dedicated server. For example, the database can be better tuned when there is no interference with the application. Database server processes can be balanced better across the multiple CPUs. Caches are trashed less frequently when there is no constant switch between database and application.

Hardware re-use

Some customers reuse old hardware in a three-tier configuration, such as after an application upgrade when the ‘old’-server has insufficient capacity to serve the total load. The server is reused as the application server or database server in the three-tier configuration. In this situation, less new hardware must be bought. However, the server should still be in good condition, otherwise it is a risk. You should look carefully at the cost because sometimes it is cheaper to buy new hardware because the price of service contracts increase whilst hardware gets older.

Infrastructure architecture vision

Larger customers built their infrastructure according to a certain vision. This is because they often have many applications, including Infor applications and non-Infor applications, up and running, and each have their own characteristics.

When building an infrastructure for this category of customers, there must be a focus on uniformity and consistency to decrease the complexity. Frequently, these customers work with partitioned machines where resources can be allocated based on the needs.

In this type of infrastructures, customers clearly define a layered infrastructure with a separate layer for the database and the application, and other layers. In this type of environment, the choice for three-tier deployment is more of a requirement than a consideration.

Performance characteristics of two-tier and three-tier

The response times in a two-tier environment are better compared to a three-tier environment because there is a network component involved. Infor advises a high-speed LAN connection between the three-tier application server and the database server. A network roundtrip only takes a couple of milliseconds. An OLTP user does not notice the network delay impact because the number of network roundtrips is low. However, batches have many network roundtrips, and thousands of network roundtrips are not uncommon. The sum of these will mean the batch performance is significantly affected by the network delay. Depending on the type of batch, the batch duration can increase by a factor 1.3 – 2; whether this is a problem depends on the requirements. When the batches are finished within the time limit constraints, this affect on the network does not matter. Batches run during the night and the duration is not an issue.

If the batch duration in a three-tier environment is critical, the following solution is possible, Figure 11-14 Batch instance. This solution is frequently chosen for Infor LN.

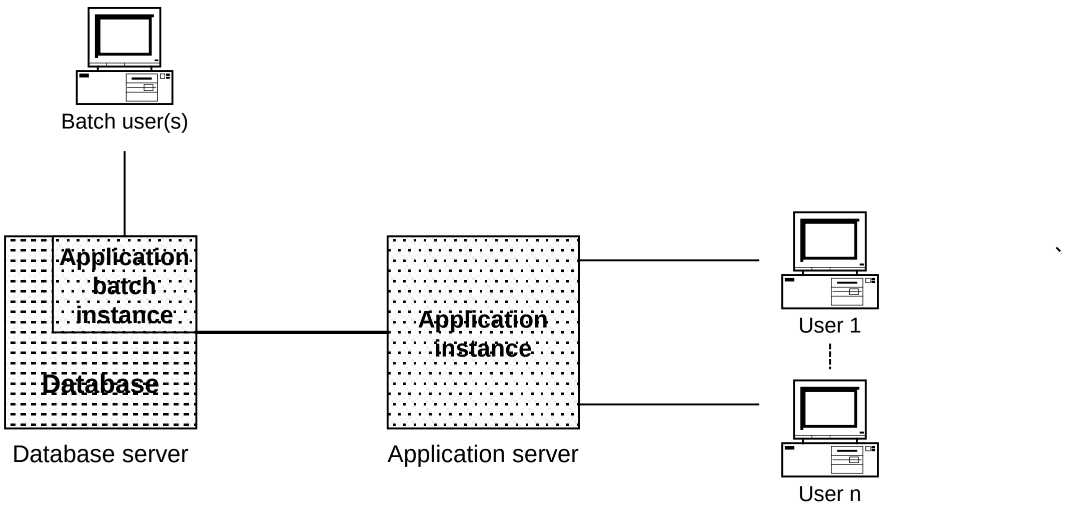

Figure 11-14 Batch instance

Install an application instance for the batch users on the database server. The critical batches can be started on the database server. This creates a two-tier situation for the batches. Installing and maintaining an additional environment creates additional workload, but this is not as bad as it first looks. The application offers functionality to distribute software and application patches. See section ‘Important configuration issues for Infor LN’, Chapter 11, the Master Application Server Concept.

Advantages and disadvantages

Section ‘Selection considerations’, in Chapter 11, shows the many aspects to consider in a three-tier solution; as always, every choice has advantages and disadvantages. This section gives an overview of the advantages and disadvantages:

- In a three-tier environment the response times are worse than in a two-tier environment because of the involved network. The OLTP users do not notice this much because the number of network roundtrips is quite low. Batches have many network roundtrips, and the batch performance is significantly affected by the network.

- From an operational point of view, a two-tier solution is easier to manage; to maintain two servers is more complex than maintaining one.
- A pure three-tier environment without high availability faculties has a lower uptime than a two-tier environment. When the data server is down in a three-tier environment, the application server cannot function. The opposite is also true, when the application server is down the database server can not function. There is a higher chance that one of the two systems will fail than the single two-tier server.
- Infor does not advise systems with less than two CPUs (see section 'Number of CPUs', in Chapter 11). Therefore, three-tier in the very low end is more expensive than two-tier.

Important configuration issues for Infor LN

In a three-tier solution, users connect to a specific application server within the configuration; the application should be reachable on the specific application server. Therefore, the application should be installed on each of the application servers, even though this is not an ideal situation from an application management point of view. All environments should be identical and run the same software version and update/patch level. Some customers install the application on one application server and distribute using a file system copy or remote copy to the other application servers. This is a valid method, but the application can handle this issue by itself. The MAS (Master Application Server) concept is developed to support this.

Master Application Server

When a server is installed as a Master application server, the Infor LN 4-gl application and the porting set will be installed. On all other application servers within the configuration, only the porting set will be installed. There is only one MAS server in a three-tier configuration.

During runtime, the porting set interprets and carries out the 4-gl application. On the MASTER, the 4-gl application can be accessed locally on the other application servers and will be accessed remotely on the MASTER through BSE_REM. BSE_REM as an environment setting to let the application know where the application objects and users information can be found.

.

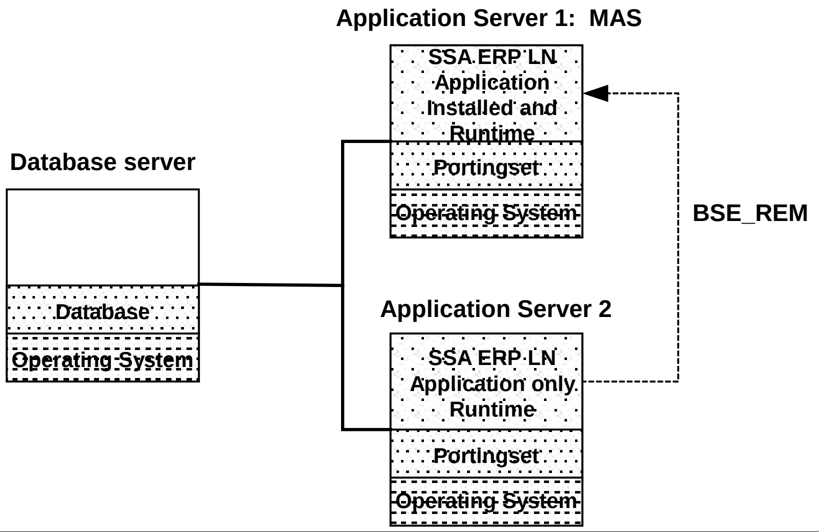

Figure 11-15 Master Application Server

From a maintenance point of view, the 4-gl application is the part which must be updated the most. With the MAS concept, only the MASTER must be updated. The other application server gets the updates automatically when the 4-gl application is accessed. The porting set must be updated on every application server. Porting set updates are rare and rather quick.

In section ‘Performance characteristics of two-tier and three-tier’, in Chapter 11, it explains that in certain situations, a dedicated batch environment will be installed on the database server. In the MAS concept, the batch environment will be handled as a normal application server environment. The batch environment accesses the 4-gl object on the MASTER application server. Only the porting set will be installed on the database server.

Recommendations

The network communication time between the three-tier application and the three-tier database server has a significant impact on the response times. Regarding the network, Infor has the following recommendations:

- Always use an LAN connection between the three-tier application and database server.

  Use the fastest communication link possible; the examples in Chapter 10 show the impact of a slow link. Since there are many packages going back and forth between the application and database server, the network impact is significant.
- Use specific tuning to combine network packages and decrease the number of network roundtrips between the database and application server such as array fetching.
- Always use at least two CPUs on the application and database server even when the sizing indicates a lower value. See section ‘Number of CPUs’, in Chapter 11.
- Never saturate CPUs to the maximum because the response times are not linearly the closer you get to the saturation point. The response times in a three-tier configuration are the sum of the database server and application server responses. When the database and application

servers are close to saturation point, the response times become very high. See section ‘Response times in a multi-tier environment’, in Chapter 11.

Hardware vendor selection

Infor cannot advise on a hardware vendor because many factors in the decision process depend on the local situation of the various regions. The local situation is not always representative for the global attitude of the vendor. This section lists the most important consideration points. Note that the cheapest solution is not always the best solution because service is also a key factor.

Price/Reliability of hardware

Prices can differ significantly between vendors, but you must ensure that the offers are compared fairly. In many quotations, the vendor is asked to deliver a solution for a certain problem and how to solve the problem is kept open. The price and quality of the solution can differ because one vendor can offer low cost hardware, and the other can offer a reliable solution with high quality components.

The availability requirements have a significant impact on the price of a solution. In a business critical situation, it can be worth investing in a high available solution because downtime can exceed the hardware investments several times. However, you must be realistic and asses the availability requirements. It is possible to build robust infrastructures which can deal with all kinds of disasters, but if downtime is accepted, the investment is too much. In certain situations, low cost hardware with spare components can be a perfect solution.

Service

Despite the smart design of the solution and quality of the hardware, there can always be situations when urgent service is required. It is important to know how fast a vendor can react to solve the problem. Secondly, what is the price of the delivered services? It is not the first time that vendors who deliver a very cheap solution do not match the criteria on service.

Relation /Trust

The hardware infrastructure is a critical component in the business infrastructure. For the continuity of the business operation, it is important to have a good relationship with your vendors. Business operations act in a dynamic world and it is necessary to re-evaluate and adjust the supporting infrastructure sometimes. Having a reliable partner who gives advice you can trust and does not give the highest priority to their own gains is very valuable. It is beneficial for everyone involved to build up a relationship with your vendor and work as partners.

Knowledge/Experience

The hardware infrastructure is business critical and must be reliable and robust. It is important that a vendor has sufficient knowledge to advice the customer. Note that the infrastructure must serve mission critical applications. A vendor which acts as a ‘box mover’ is not always the best one.

Support

Check whether the vendor has sufficient bandwidth to deliver enough support.

Knowledge

Available knowledge in a company can be a motivation in selecting a vendor, and you do not have to re-invest in building this knowledge.

Operating system selection

Many applications are supported on multiple operating systems, which means a choice must be made. An operating system selection is quite a fundamental decision.

For some people it is an emotional choice because they believe in the operating system they have experience with and do not like to change. It is advised that you make the choice on rational points.

Hardware dependency

There are operating systems which can run on multiple hardware platforms. There are ports available for the various CPUs and operating systems on the market which only run on certain types of hardware. Both have their advantages and disadvantages. Having an operating system which runs only on certain hardware is, in general, heavily optimized for the specific platform it runs on; a generic OS must make concessions on this point. The advantage of a generic operating system is that there is no lock for certain hardware platforms.

Scalability

The scalability of the operating system is an important selection criterion. The scaling of a solution is determined significantly by the operating system. For more information about this subject, see section ‘Server scaling in relation to sizing’, in Chapter 11.

Open source

Open source operating systems are a new valid alternative. A recognized operating system is Linux, Which has become a mature operating system and can be an attractive option. Infor supports Linux for some applications, but check the support matrix for details. The open source business approach is completely different from the normal business approach. Some companies prefer it, while others are hesitant. Much is written on this subject. Therefore, this subject will not be discussed in detail.

Other

Many selection criteria mentioned in the sections ‘Hardware vendor selection’ and ‘Operating system selection’ are also valid for the operating system selection.

Database vendor selection

Many of the selection criteria mentioned in the previous sections ‘Hardware vendor selection’ and ‘Operating system selection’ are also true for the database selection.

Performance

Previously, the maximum performance and throughput of the database were a major selection criteria. They are still important, but this discussion is moved to the high-end and the majority of customers are not active in this area.

Price/Performance

The price/performance ratio is another important selection criterion and can differ significantly between databases. Note that this ratio can be application dependend. Hardware costs to run databases have declined, which makes the price/performance ratio less significant.

License costs

Database license costs are an important selection criterion. License costs in a two-tier combination can be an issue for CPU-based license models. All CPUs in the server are counted, but they are not reserved for database capacity.

Manageability/Ease of use

Because all databases have acceptable performance and include all elementary functionality, the manageability and operational costs become one of the most significant factors to compare. Now, the operational costs are the biggest factor in the overall costs.

Other

Other selection criteria are as follows:

- Service
- Relation/Trust
- Knowledge/Experience
- Support
- Knowledge

For an explanation, see sections ‘Hardware vendor selection’ and ‘Operating system selection’.

Response time behavior

The application response times depend on the application and the underlying hardware infrastructure and software stack. The response times are built up by many factors such as CPU speed, number of CPUs, 2-tier, 3-tier, OS scheduler, batch, OLTP, load of specific transaction and overall load. It is important to understand the response behavior on certain configurations. This background information can be used when selecting an infrastructure type because it gives a better understanding when the system reaches its limits.

Response time definition

Response times are defined as follows:

Response time is the time between the submission of a task, or request, and the receipt of a response.

The response time is the sum of the delays of all the involved layers and networks. See Figure 11-16, below.

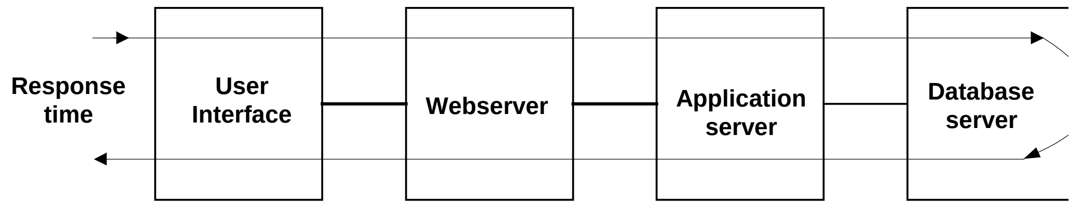

Figure 11-1 Response time

Response times OLTP

The information in this section is applicable for Infor LN and other single threaded applications.

Typical OLTP response times are relatively short; typically, the duration is a couple of seconds but not more than 60 seconds. Between the OLTP transactions there is a user think time or wait time (see Figure 11-17).

The CPU load of an OLTP transaction is continuous. The CPU is hardly busy to serve an OLTP user. Overall, the CPU has sufficient capacity left to serve multiple users in parallel. Depending on the application and CPU time needed per request, the CPU can handle hundreds of users.

If the system serves multiple users, the users are not claiming resources from the CPU consecutively and the users work completely randomly. Users start to obstruct each other and fight for resources, which influences the response times. More users lead to higher CPU utilization and response times start to degrade non-linearly.

For typical OLTP response time behavior, see Figure 11-17. The end user posts an action (Action1) and must wait until the system comes back to the end user with a reaction; this wait time is the Response time. While the end-user waits for a response, the system is busy and the CPU spends cycles for the user. Figure 11-17 shows the CPU is not constantly busy to serve the user because the response time also includes waiting for IO and network delays, and so on. If other users work at the same time on the system, these users get CPU cycles. The CPU must serve all users in parallel.

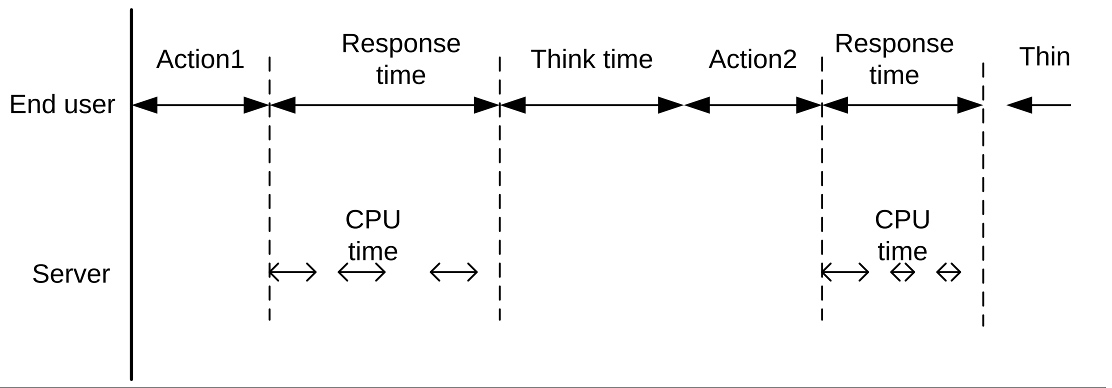

Figure 11-17 OLTP interaction times

Figure 11-18 shows how the response times develop when the CPU becomes more saturated. Usually, the response times stay flat until the CPU is 85 percent saturated; beyond this point, the responses are unpredictable and the curve is not linear. This is one of the reasons why a 20 percent margin for peak load is reserved in a sizing (see section 'Peak load margin', in Chapter 6). A sizing should not be based on 100 percent CPU load.

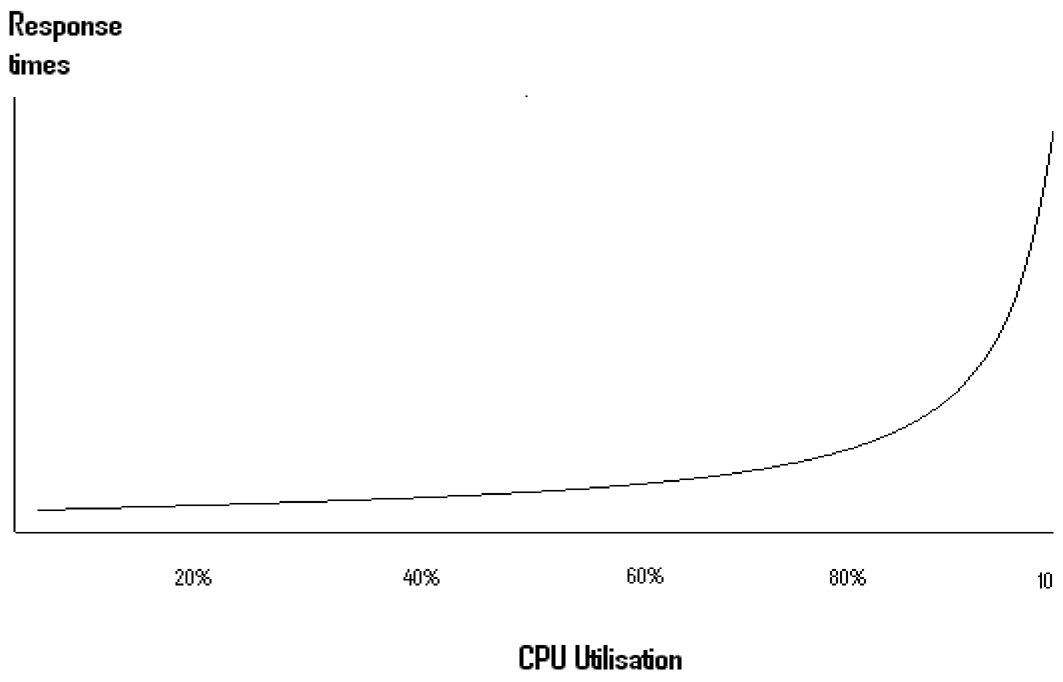

Figure 11-18 Typical OLTP response times

The number of CPUs has an affect on the response times (see Figure 11-19). A one CPU system is compared with a two CPU system; when the system is inactive, the response times for the one and two CPUs are almost the same because the requests are not distributed across the multiple CPUs. The speed of the individual CPU determines the duration of the response times. The response time curve for the two CPUs stays flat longer than one CPU curve. The requests are handled by two CPUs and the queuing starts later. Although not shown in the figure, the same pattern continues above the two CPUs. The more CPUs, the longer the curve stays flat.

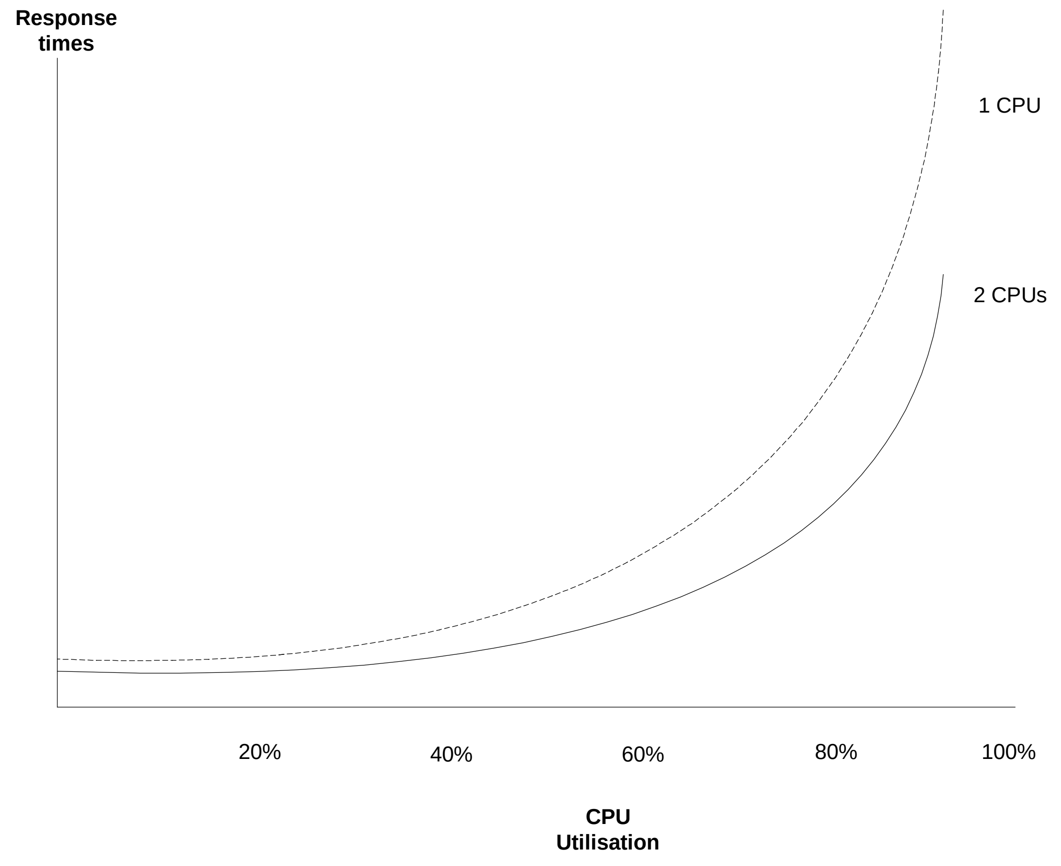

Figure 11-19 Influence of the number of CPUs on the response times

Response times Batch

The response time of a batch job is the batch duration. A batch characteristic is that there is no user interaction. The user or a job manager starts a batch job, and there is no system interaction until the batch is finished. A batch claims as much CPU power as possible. The batch duration is determined mainly by the availability of system resources. If there are no other processes on the system asking for CPU resources, the batch job can claim almost one complete CPU. The batch only gives up the CPU when some IO must be done. Figure 11-20 shows the CPU load versus the response time.

In Infor LN, batches are single threaded; this implies that the batch cannot claim more than one CPU. See also section ‘Single threaded versus multithreaded software’, in Chapter 11.

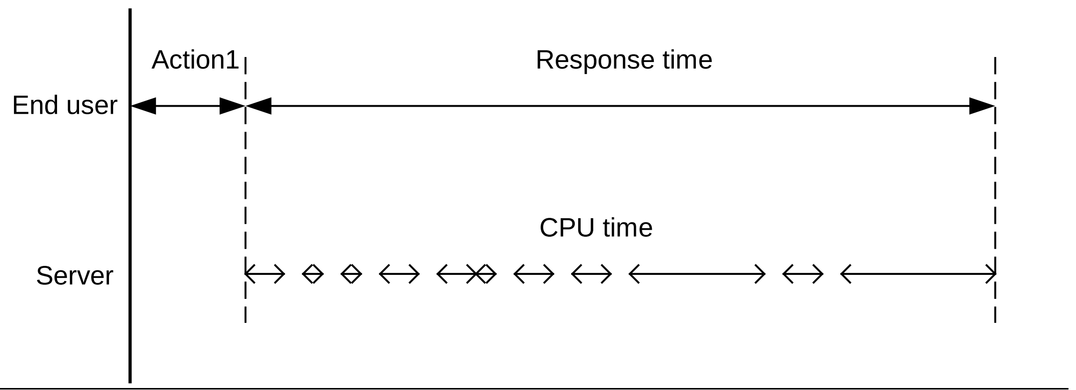

Figure 11-20 Batch response times and CPU saturation

Figure 11-20 shows one single action, Action 1. After the action, the system is busy for a long time. During the response time of the batch duration, the CPU is quite busy, much more than CPU used during OLTP transactions compared with Figure 11-17. The CPU is not used all the time. The pauses in the graph can be waiting for IO, or the CPU is busy to serve other processes on the system.

Batches on multi CPU machines

The number of CPUs in the system has an affect on the batch behavior and duration.

Figure 11-21 shows the CPU load on the system with one CPU. The batch job claims close to 100 percent of the available CPU capacity, which is the same situation as explained in Figure 11-20. When the same batch job is started on a two way system (see Figure 11-22) the batch job again claims one CPU, regardless of whether there are two CPUs in the system. The overall CPU capacity is saturated for only 50 percent, because only one of the two CPUs is used. The batch job duration in the situations (Figure 11-21 and Figure 11-22) will be approximately the same.

The remark “A batch claims one CPU”, does not mean the same CPU is always used. The Operating system scheduler can move the batch process each processor time-slice to another CPU. The expression means the *capacity* of a single CPU.

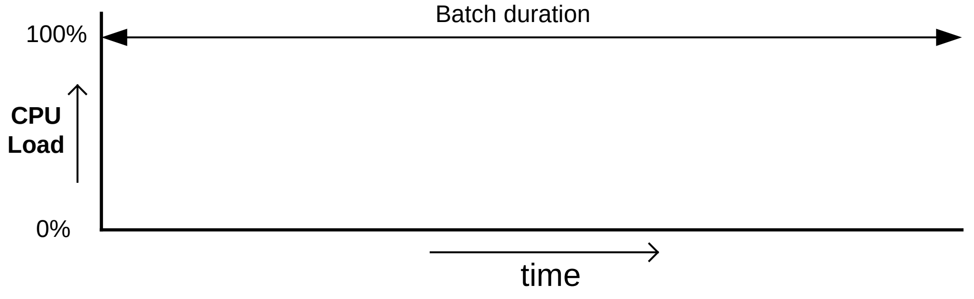

Figure 11-21 One batch and one CPU

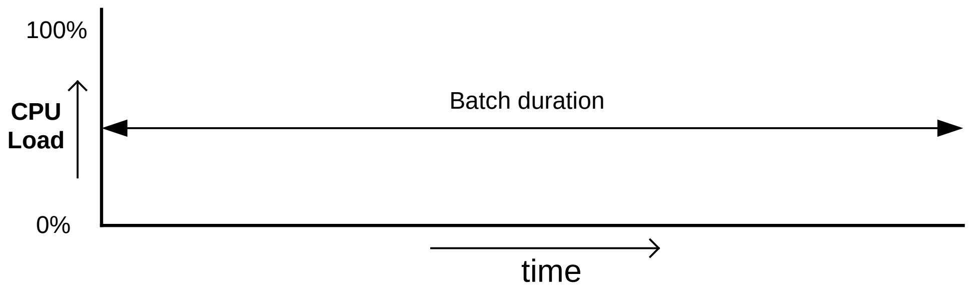

Figure 11-22 One batch and two CPUs

If two batch jobs are started on a one CPU system, both batches start to claim as much CPU resources as possible. Each of the batch jobs want to claim a complete CPU, but only one CPU is available. The batch jobs start to compete for CPU resources. In total, the two batch jobs will claim close to 100 percent of the overall available CPU capacity. 100 percent on a one way system means that the one CPU in the system is completely occupied. The operating system scheduler allocates CPU resource to each of the batch jobs, and the two batch jobs must share the CPU. The duration of the two batch jobs will be much longer than the duration of the batch job when running individually. You should expect a factor two, because there are two batches which must share the CPU. In practice, the factor will be more than twice as high, and a factor four is not unrealistic. The fighting for CPU resources introduces a lot of overheads. The operating system is constantly busy with rescheduling the batch processes. In the OLTP curve, you can see the system response times close to system saturation are not linear; overloading the system with too many batches leads to the same situation.

Two batch jobs on two CPU systems will claim almost two CPUs. In total, the two batches will claim close to 100 percent of the overall available CPU capacity. 100 percent on a two way CPU system means that the two CPUs in the system are completely occupied. The duration of the two batches will be approximately the same as the runtimes of the batches individually. The resources the batch jobs want are available and the batch jobs are not competing for CPU resources.

Based on the batch behavior described above, the generic batch sizing rule is defined as follows:

Allocate for each concurrent batch job one CPU.

This rule is used in most of the sizing guides. Because some products have variations on this rule, see the specific sizing guides.

The rule assumes that no other bottlenecks or capacity constraints are on the system. If, for example, the IO capacity on the system is limited and batches have to compete for IO resources, a batch can never saturate a complete CPU. The same is true when there is insufficient memory in the system.

Heavy versus light transactions

The operating system scheduler divides the available CPU resources amongst the processes which request CPU resources. To do this, the various operating systems use different scheduling policies. Usually, the most demanding resource processes will get the fewest resources. If there are sufficient CPU resources available overall, the influence of this policy is not noticeable. However, if the system is close to system saturation, the influence of this policy is very noticeable. Figure 11-23 shows the response times of the various transactions in relation to the overall CPU load on the system. The heavier a transaction, the worse the response times become when the CPU is close to saturation. Light transaction responses double when the system is loaded, and heavy transactions triple or increase by even more.

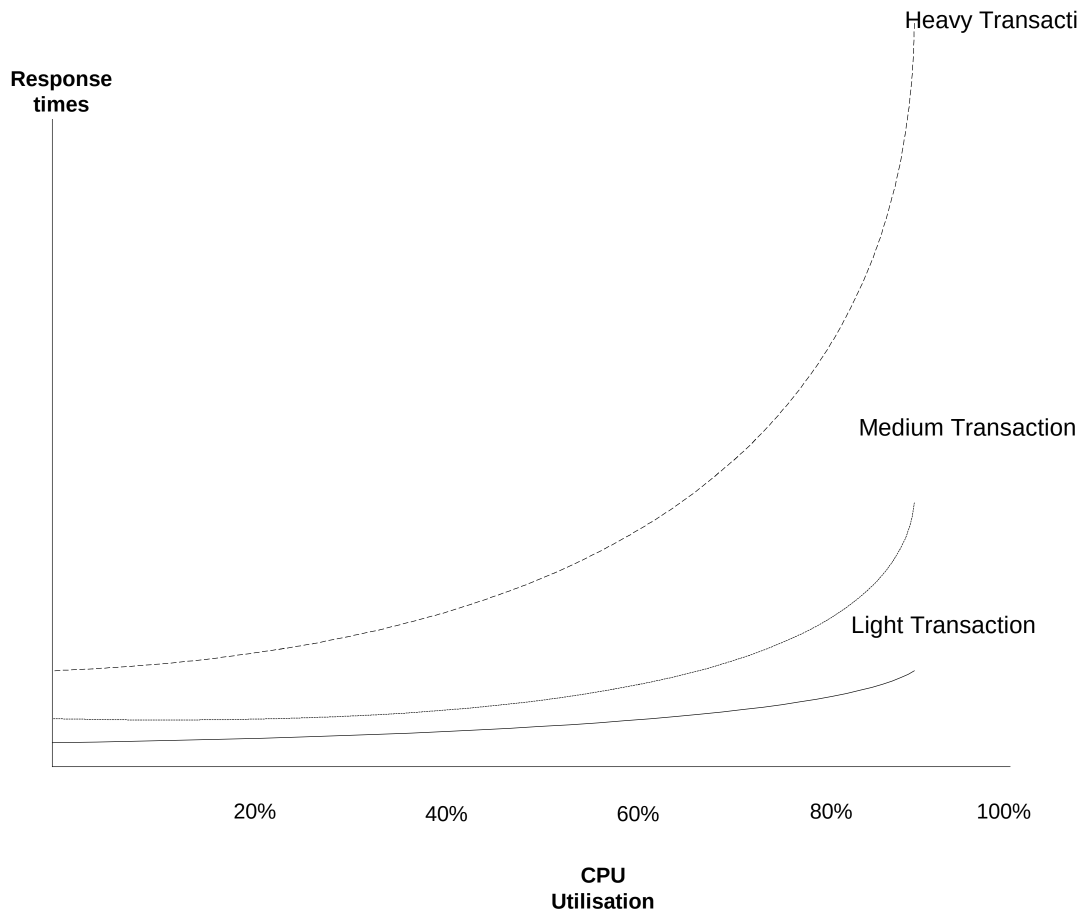

Figure 11-23   Response time in relation to transaction load

If a batch job is started on a system which is almost occupied with OLTP load, the system will become completely overloaded. The OLTP transactions are relatively light than the load of the batch job. The response times of the OLTP users will increase with approximately a factor two, but the batch duration will last longer than a factor four or more.

The balance the operating system scheduler allocates to resources between OLTP and Batch jobs differs per operating system. Windows handles this differently than UNIX, but even between the various UNIX flavors there are significant differences.

IBM iSeries handles batches differently. Generally, for batch and OLTP, different CPU capacity is reserved. See section ‘IBM iSeries characteristics’, in Chapter 11.

Number of CPUs

The number of CPUs affects the response times. Going to multiple CPUS does not make the response times faster with a single-threaded application such as Infor LN, because the application does not benefit from additional CPUs. The speed of an individual CPU determines the duration of response times.

Multiple CPUs make it possible to serve multiple requests in parallel. Multiple CPUs can empty the queue of processing requests faster because the multiple CPUs work on the same queue.

Processing requests can be light or very intensive, such as batches. Having light and intensive requests mixed through each other can cause the various requests to influence each other's response times. The heavy processes try as much as possible to get CPU resources, which indicate the light processes always deal with a busy CPU. The OS scheduler allocates the available resources by giving fewer resources to the heavy processes, which does not prevent the response times from varying significantly. The lower the number of CPUs in a server, the sooner the aforementioned event will happen.

To deal with peak variations, this base rule is applicable for Infor LN:

> *Always provide Infor LN servers with at least two CPUs. This applies to two-tier and three-tier servers.*

If the number of users is very low, it is not advised to follow this rule.

If a CPU consists of multiple cores or hyper threads, you can consider having a minimum of two cores or hyper threads.

Response times in a multi-tier environment

The difference in response times between a two-tier and a three-tier environment is explained in section 'Performance characteristics of two-tier and three-tier', in Chapter 11. For multiple tiers the behavior is the same.

This chapter explains that the closer to system saturation the response times are, they become non-linear. In a multi-tier environment, you must ensure that when the servers of the various tiers are close to system saturation, the sum and end-user response times will be extremely high.

Figure 11-24 shows the response times in the non-linear area are counted together. The combined response time can be very high.

For multi-tier it is important to keep a 20 percent margin for peak load.

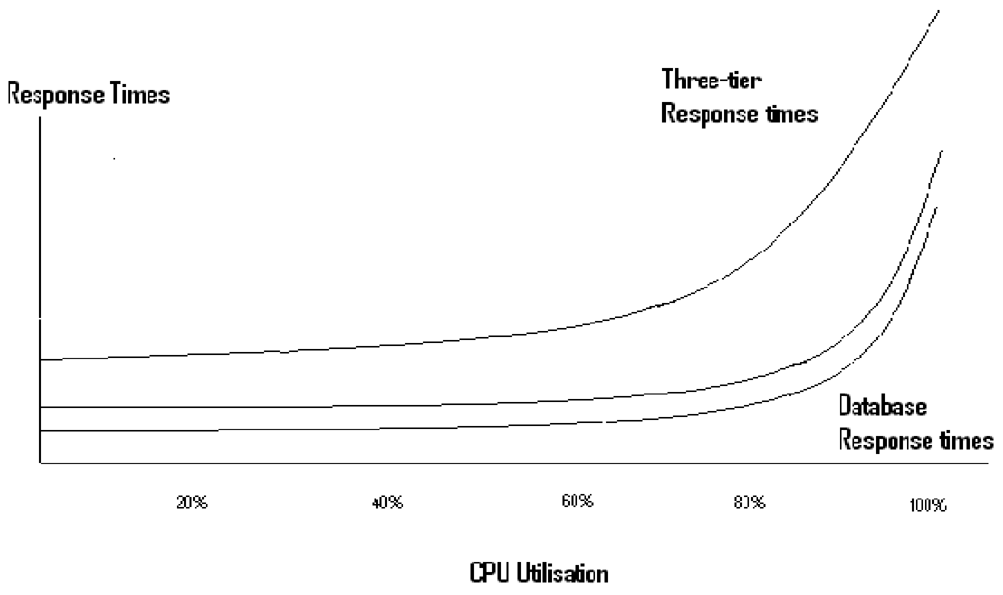

Figure 11-24 Three-tier response times

Infor LN: Single bshell versus parallel bshell performance

The bshell is the virtual machine within Infor LN and is a single-threaded process from an operating system point of view.

When a process is being carried out, the functionality of a single-threaded process will be processed by one CPU. A bshell running a batch job tries to claim as much CPU power as is available, but this can never be more than one CPU. The duration of a batch depends on the speed of the individual CPU. A faster CPU will help improve the batch performance. There are situations where a lot of data must be processes and one batch bshell cannot handle this within the required time. For these situations, the concept of a parallel bshell is developed. A batch job will be split and distributed across a configurable number of bshells. The distribution across the various bshells is done by a master bshell and it is carried out by the slave bshell.

The parallel bshell concept is not organized in the technology layer, but on the application layer. It must be programmed for each batch and implemented for the most common and critical batches.

Duration

The duration is best explained by Ahmdahl’s Law. Amdahl stated that a job can split into two parts: A part which can be paralyzed, and a part which can not be paralyzed. The total time it takes to carry out the job is the sum of the two:

Total time to carry out job = parallel part + serial part

When the parallel task is split in N task and spread across N CPUs, the duration can be calculated with following formula:

Total time to carry out job = (parallel part) / N + serial part.

The formula assumes the scaling across the N CPUs is 100 percent; in practice, this is not the case. The parallel tasks share resources which flattens the scaling (see section ‘Scalability’, in Chapter 11).

Parallel bshell means that multiple batches, although they are shorter batches, run in parallel. This can have significant impact on the system. To benefit from the parallelization, the scaling must be optimal. There can be various bottlenecks which prevent the system from scaling (see section ‘System scaling’, in Chapter 11). Ensure there are sufficient CPUs in the system to handle the multiple batches.

The following is a general rule:

- In practice, the throughput improves 50 percent when the number of bshells is doubled.

The distribution between the parallel part and the serial part in the batch is different. Therefore, how much faster a parallel batch runs compared to a serial batch can vary per batch.

Chapter 12 Sizing Tools

12

An overview of the various tools available to help with sizing are given in this chapter.

Sizing Guides general

There are many sizing guides for Infor products available. A sizing guide is a document which contains detailed sizing information for a specific product. The main focus of the sizing guide is to provide knowledge for the specific product on the various supported databases, operating systems, and implementations. Sizing guides mainly focus on resource consumption.

The target audience for the sizing guide is people with general sizing knowledge. The sizing guides doesn't provide detailed background information on how the sizing process looks like and all the sizing related subjects like high availability, backup etc. This type of subjects will be covered in this sizing whitepaper. It is advised for people who want to learn more about, first to read the sizing whitepaper.

Usually, sizing guides contain the following information:

- Introduction.
- Functional product description.
- Technical architecture description.
- Explanation of sizing methodology.
- A description of how the sizing figures are determined, a description of the test environment, and a test method and summary of the test results.
- Server sizing information for the various database, operating systems, and implementation scenarios such as the following:
    - CPU.
    - Internal memory.
    - Network.
    - External memory.
- Client sizing information.
- Sizing example.
- Sizing questionnaire.

Sizing Whitepaper

This document provides information around many sizing related subjects which are not covered in the sizing guides and also a lot of background information. However, this sizing whitepaper will not provide detailed sizing information; the focus is on the sizing approach and sizing process. This sizing whitepaper contains information on a limited number of products, but updates will contain more information.

Examples of subjects covered in this sizing whitepaper are as follows:

- Sizing explained: What is sizing? Importance of sizing, and when to carry out a sizing.
- Various sizing methodologies.
- Sizing process.
- Sizing recommendations, definitions, attention points and various subjects such as growth, margin, peak load and so on.
- Sizing background information such as the following:
    - Various computer architectures in relation to sizing.
    - Two-tier versus three-tier.
    - Response time behavior.
- Sizing checklist.

Sizing Portal

A sizing website has been developed which gives an overview of all available sizing documents and tools. The website is called Sizing Portal, and can be accessed by the intranet through the following link: http://pbc.infor.com/

For Infor partners, access Sizing Portal by clicking on the following link:
http://channelportal.infor.com/

Sizing account

You can send feedback and questions about the sizing guides and tools to the sizing e-mail account sizing@infor.com. Feedback, whether positive or negative, helps to improve documents, tools, and set priorities.

2nd line Sizing support is delivered. If you need help with your sizing, contact the sizing account. It is not a service whereby they carry out the complete sizing, but if you have problems they are happy to help where required. The ultimate goal of the sizing tools, documentation, and sizing account is to let people become self-supporting.

Infor LN

Sizing guide

The Infor LN sizing guide is based on the setup described in section ’Sizing Guides general’, in Chapter 12. The Infor LN sizing guide contains information for Infor Baan IVc, Infor Baan 5.0, and Infor LN.

Sizing Assistant

The Sizing Assistant is a wizard-based web application which supports Infor Baan IV, Infor Baan 5.0x, Infor LN, and WMS 9.x sizing.

The Sizing Assistant guides the end-user through a sizing. All the necessary questions for a sizing are asked, which is helpful for those not very comfortable with carrying out a sizing; the risk that they overlook something decreases because all questions must be answered. The Sizing Assistant makes it easier to carry out a sizing, but it is still necessary to understand the sizing fundamentals. You must realize that sizing of a customer hardware infrastructure is a complex activity, and choices need to be well considered because it is not just a selection in a tool.

After all questions are answered, the Sizing Assistant calculates the server requirements. After the calculations, all servers of selected hardware and database vendor combinations capable of handling the specified workload are listed and a server can be selected from this list.

A sizing result summary is created. The specified workload, vendor selection, the calculations, and the selected server(s) are listed. This report can be printed or stored on the server.

The Sizing Assistant does not replace the sizing guide, and it must be used in addition to the sizing guide. The intention of the Sizing Assistant is to standardize and speed up the sizing calculations. The Sizing Assistant and sizing guide based on the same data, but the sizing guide contains more specific data and important remarks which cannot be part of an automated tool.

Those who have access to the Infor intranet or partner web can access the Sizing Assistant. You must register for the Sizing Assistant, and login account information will be received by mail. The sizing results are stored on the server. To keep the sizing results separate, an account is necessary. Infor implementation partners can work on the customer.

E-Sizing Guide

The E-Sizing Guide is a HTML version of the Infor LN sizing guide (see ’Sizing Guides general’, in Chapter 12) and contains exactly the same information. The sizing data is better accessible because you can easily browse through the document using the hyperlinks.

Sizing Questionnaire

The Infor LN sizing questionnaire is a document which must be completed by the customer because it collects all the necessary information to carry out a sizing. Therefore, it must be filled out carefully. Please note that input which does not reflect the reality can lead to under or over-sizing. It is advised that you explain the questionnaire to the customer and why the information is important. If possible, the ideal situation is to fill in the questionnaire together with the customer.

Infor LX

Sizing guide

The Infor LX sizing guide is based on the setup described in ’Sizing Guides general’, in Chapter 12. The Infor LX sizing guide contains information for Infor LX 8.x, but does not include system information. A sizing carried out based on the Infor LX sizing guide only provides capacity requirements. Based on this information, you must contact IBM to select the necessary server model.

IBM Workload Estimator

The IBM Workload Estimator is a wizard-based web application which guides the end-user through an Infor LX sizing. All the necessary questions for a sizing are asked, which is helpful for those not comfortable with carrying out a sizing. The risk that something will be overlooked decreases because all questions must be answered. The IBM Workload Estimator makes it easier to carry out a sizing, but it is still necessary to understand the sizing fundamentals. Sizing of a customer hardware infrastructure is a complex activity, and choices must be well considered because it is not just a selection in a tool.

After all questions are answered, the IBM Workload Estimator calculates the server requirements and a recommended server are presented. The IBM Workload Estimator connects to an internal IBM database with the latest IBM iSeries hardware information to select a server system.

Questionnaire

The Infor LX sizing questionnaire must be filled in by the customer because it collects all the necessary information to carry out a sizing. The questionnaire must be filled in carefully. Note that input which does not reflect the reality can lead to under or over-sizing. Therefore, it is advised that you explain the questionnaire to the customer and why the information is important. If possible, fill in the questionnaire together with the customer.

Chapter 13 Sizing checklist

13

This chapter contains two checklists that can be used during a sizing; they help to ensure that every area which has a relation to sizing is covered.

Sizing process checklist

This checklist helps to check whether the most important topics are covered during the sizing process. For background information about the sizing process, see Chapter 4.

	Subject:	**Description**	**Reference in Sizing White paper**
☐	Sizing project group	Define sizing project group. Select right people and ensure all knowledge areas are involved.	Section ‘Who does sizing’, in Chapter 2.
☐	Sizing reason	Describe the reasons to initiate a sizing because this can have an influence on the sizing approach.	Section ‘When is a sizing carried out’, in Chapter 2 and section ‘Process, reasons to carry out a sizing’, in Chapter 4.
☐	Sizing risk	Determine how business critical the sizing is, and the impact.	Section ‘Importance of sizing’ and section ‘Main steps during a sizing’
☐	Sizing methodology	Select sizing methodology.	Section ‘Sizing methodology to use’, in Chapter 2.
☐	Sizing input	Collect information to describe the workload.	
☐	Questionnaire	Fill in sizing questionnaire.	Section ‘Sizing Questionnaire’ and ‘Questionnaire’, in Chapter 12.
☐	Carrying out a sizing	Carry out sizing with specialist of the software vendor, implementation partner, or	

		hardware vendor.	
☐	Define infrastructure	Make important infrastructure decisions such as the following: - Security. - High availability. - Single versus multi-tier solution. - Backup strategy. - Consolidation.	
☐	Evaluate sizing	Check whether the implemented infrastructure has sufficient capacity to serve the workload. If not, carry out a gap analysis.	
☐	Next sizing	Sizing is not a one-time event. Implement a process to ensure sizing is carried out on a regular basis.	Section ‘Sizing definition’ and ‘When is a sizing carried out’, in Chapter 2. Chapter 4 and section ‘Capacity planning’, in Chapter 3

Sizing content checklist for new sizings Infor LN

This checklist helps to check whether the most important sizing topics are covered during a sizing.

	Subject	**Description**	**Reference in Sizing White paper**
Peak load			
☐	Peak period during day	Sizing input must be based on peak period during the day to prevent under-sizing.	Section ‘Sizing peak periods’, in Chapter 3.
☐	Seasonal peaks	Some businesses have seasonal influences. It is advised that you base the sizing on the season peak.	Section ‘Sizing peak periods’, in Chapter 3.
Workload definition			
☐	Determine application	The sizing should be based on the	

	version	correct application version because there can be a significant load difference between application versions.	
☐	Named users	Number of users which can potentially use the system or application.	Section Various user types', in Chapter 6.
☐	Connected users	Number of users which are connected to the application.	Section Various user types', in Chapter 6.
☐	Concurrent users	Number of users actually using the application.	Section Various user types', in Chapter 6.
☐	Load per concurrent users during peak hour	Determine what kind of functionality each user carries out. The various functional areas can have significantly different load characteristics.	Sections: 'Sizing according to the sizing guide', in Chapter 5 and 'Sizing model', in Chapter 6.
☐	Daily Batch load	Which, and how many batches, are carried out during the day	
☐	Night batch load	Which, and how many batches, are carried out during the night.	
☐	Transaction volumes	Sizing is not based on transaction volumes. Based on this information, the sizing expert can fine tune the sizing based on experience.	
☐	Load of Non-Infor software	The infrastructure needs sufficient resources to serve all software.	Section 'Non-Infor Software', in Chapter 6.
☐	Load of other Infor software	The infrastructure needs sufficient resources to serve all software.	Section 'Various Infor software products'.
☐	Define UI	The various UIs have different load characteristics.	
☐	Integrations	Integrations have an impact on the load of the backend server.	Section 'Integration', in Chapter 6.
☐	Customizations	Customizations have an impact on the load of the backend server.	Section 'Customizations', in Chapter 6.
☐	Multi-byte/Unicode	Multi-byte/Unicode usage has a significant impact on performance.	Section 'Multi-byte/Unicode', in Chapter 6.

Important factors			
☐	Growth	Reserve capacity for the expected business growth.	Section ‘Growth’, in Chapter 6.
☐	Margin	In situations where the sizing input is very rough, not well considered, or when there are many uncertainties, it is advised that you consider a margin.	Section ‘Margin’, in Chapter 6.
☐	Peak load	Load is not evenly distributed and can be spiky; you must apply a factor for this.	Section ‘Peak load margin’, in Chapter 6.
☐	Years of history	Disk storage must be reserved.	

Vendor selection			
☐	Select hardware vendor		Section ‘Hardware vendor selection’, in Chapter 11.
☐	Select operating system		Section ‘Operating system selection’, in Chapter 11.
☐	Select database vendor		Section ‘Database vendor selection’, in Chapter 11.
☐	Select vendor web server software		

Infrastructure definition			
☐	Central solution versus decentral solution	This can have an impact on the size and number of the servers and the network requirements.	Section ‘Central versus decentralized solution’, in Chapter 6.
☐	Consolidate various Infor software products	Aggregate requirements for all Infor applications.	Section ‘Various Infor software products’, in Chapter 6.
☐	Two-tier versus three-tier	This has an impact on the size and number of servers. This can have a relation to the high-availability solution.	Section ‘Two-tier versus three-tier’, in Chapter 11.

☐	Number of application servers	This must relation to server size, high-availability solution, and operational costs.	
☐	Heterogeneous environment	Application servers and database servers do not have to run on the same operating system and hardware.	Section ‘Selection considerations’, in Chapter 11.
☐	Investigate high availability requirements	High availability requirements can be crucial in business critical environments. Keywords are as follows: Raid levels, redundancy, and so on.	
☐	Investigate LAN/Wan requirements	Determine network bandwidth and latency requirements.	
☐	Local client versus/Terminal server	Terminal servers can be used to save on network and operational cost.	
☐	Define number of web servers/proxy servers and location	Proxy servers can be considered at the remote locations.	
☐	Load balancing on the web servers	Multiple web servers can be considered for high availability and performance reasons.	
☐	Backup window	The backup window puts requirement on the backup solution. Is there sufficient time to backup offline? Is online backup required?	Section ‘Backup window’, in Chapter 6.
☐	Batch window	Is there sufficient time to run batches in serial or is parallelization required? This has an impact on the number of CPUs.	Section ‘Batch window’, in Chapter 6.

Appendix A Definitions, acronyms, and abbreviations

A

Definitions, acronyms, and abbreviations used in this document are listed below in alphabetical order.

Term	Definition
Availability	A metric to measure the percentage of time a system is ready to deliver services.
Backup window	The time reserved to finish a backup. Usually, the system is dedicated to the backup task during this period.
Bandwidth	The amount of data that can be transferred over a communication link per unit of time. Usually, the measurement unit is bits per second.
Batch/Batch Job	Process intensive activities which are grouped together and carried out as a unit. Between the activities there is no user interaction. Batches can have a significant impact on the sizing.
Batch window	The time reserved to finish batch processing.
Benchmarking	A process of running a standard workload against a system or test configuration to compare its performance with others.
Bottleneck	A resource that first saturates when the workload increases. Often, this resource has high resource consumption and blocks other resources from being saturated.
BRU	Benchmark Reference User. Fictitious user with loadfactor 1.
Cache	A small piece of fast memory holding recently accessed data. It is designed to have fast access when the same data is subsequently accessed. Usually, a cache is used to speed up processor-memory access. In modern architecture, caches are used to speed up disk access and data access by the network.
Capacity Planning	A process of predicting future system load. If future resource demand increases, the system is tuned to free up resources or capacity is added to system to guarantee an available system with an acceptable response.
CC-NUMA	Cache coherent NUMA. Cache coherence manages consistency between cache and memory.
CIFS	Common Internet File System.
Client-server	Client-server is a network computer architecture where the client

	functionality is separated from the server. The client sends a request to the server and the server carries out the request and sends the response back. Client-server architecture can consist of multiple layers which deliver services for each other such as database servers, application servers, web servers, and fileservers. One reason to introduce client server is to reach a high degree of scalability.
Cluster	Group of systems that can handle distributed workload. A cluster provides redundancy and failover.
Concurrency	The handling of multiple requests simultaneously.
Contention	Competition for resources.
CPU time	The amount of time a CPU spends to process a certain task excluding the waiting time.
Disk array	A storage system containing two of more disks designed to improve performance and reliability.
Failover	A method to provide an alternative environment, machine or group of machines for tasks when the original machine(s) fails.
HBA	Host Bus Adapter. An adapter which connects storage devices or networks to a computer.
Latency	Time to complete a request.
LAN	Local Area Network. Computer network which covers a local area. Typically, LANs work on a higher data rate compared to a WAN.
NAS	Network Attached Storage. NAS is a storage subsystem with one or more hard disks. A server can connect to a NAS using a network, and is frequently TCP/IP-based.
NFS	Network File System. File system, shared by a network.
NUMA	Non-Uniform Memory Access. A multi-processor architecture with two or more processors. Each processor has its own local memory and the processors can access each others memory. A local memory access is faster than a remote access.
OLTP	Online Transaction Processing is a type processing which reflects transaction-oriented data entry or data retrieval.
PBC	Performance and Benchmark Center
Performance	Performance is a quality aspect of software with three characteristics for which it is possible to have the following predefined or required attributes: Time Behavior: The reaction of the software to the environment in which it is running regarding the following:

	– The interactive response time. and/or – The throughput of “batch” transactions. Resource Behavior: – The efficiency of resource consumption (CPU, memory, network, disk, and I/O). Scalability: – The deviation of performance and resource consumption as a function of increased user/transaction/data volumes.
Response time	The time between the submission of a task or request and the receipt of a response.
Resources consumption	In the context of this document, this refers to the amount of network, disk, CPU, or memory capacity used.
SAN	A Storage Attached Network is designed to attach devices such as disks and tapes to a server.
SATA	Serial Advanced Technology Attachment is a bus which connects disks to system.
SCSI	A Small Computer System Interface connects peripherals to a computer and is mostly used for storage devices.
SMP	A computer architecture which consist of two ore more identical CPUs. The CPUs are connected and share a single main memory.
Scalability	The ability that added resources to a system can increase processing capacity and throughput.
Sizing	A process to translate capacity requirements into infrastructure requirements.
Think time	The time a user waits between end-user actions while he works actively with the system.
Throughput	The number of requests or data units processed per unit of time. In a communication context, throughput is defined as data transferred per unit of time.
WAN	Wide Area Network. Computer network which covers a wide geographical area. Typically, WANs work on a lower data rate than an LAN.

Appendix B Use case 1: Consolidation

B

Introduction

Company Power Engine Inc. is a lon-time satisfied Infor customer. Headquartered in Amsterdam, the Netherlands, Power Engine has many subsidiaries across Europe and Asia. Each subsidiary has its own server with Infor software running. Power Engine decided to consolidate the servers to be more cost-effective to one single server located at the headquarters.

In this case is explained how such a sizing question can be worked out. The emphasis is on the approach and less on the technical details.

Sizing approach

The sizing task looks straight forward. But when you start thinking about, several questions pop up. To make sure that all aspects are covered the checklist 'Sizing process checklist' and 'Sizing content checklist for new sizings Infor LN' is used as guideline.

- Project group

  The first step is to define a project group with all disciplines involved. Looking to the project description it looks overkill to involve all disciplines, because the project description indicates only a technical project. But there are many questions to answer like will the software version change? Is this a good moment to roll-out new procedures or customizations?

- Sizing reason

  After the first meeting of the project group it became clear that this consolidation project probably is a good moment to combine future requirements which the functional, business and management guys have and impact the sizing. The functional guys want to roll out a new module of the application. This leads to more concurrent users.   Without being very specific management indicated to calculate with business growth of 50 percent the coming three years. Management could not say whether this is organic growth or may be new acquisitions.

- Risk

  The technical guys explained management the risks from consolidation. Having one central server on one location increases risk. If the consolidate server goes down, the complete company will have problems. There are various solutions to solve this problem. The management made a trade off between costs and risks. There will be one single location, with a three tier server solution. Both the application server and database server will operate in a cluster and can take over the functionality of each other in case of calamity. Both servers have sufficient capacity to operate as single server running application and database functionality.

- The main reason is clear: consolidation. But after discussion with the project group new things came to the project.
- Sizing methodology

  All sizing methodologies are possible in this situation. Managements want to be really sure that the solution works. In first instance they liked the safety factor a customer specific benchmark offers, regardless the high costs. But after some discussion the technical projects member indicated that risk for under-sizing actually isn't that high. They proposed the following approach:

  Measure the load on the current systems for a reasonable period of time. Consolidate this data for all servers and use this as input for the new big server solution. The method provides the exact load characteristic. This characteristic is probably more accurate than ever can be reached by customer specific benchmarks. The load of the new functionality will be estimated via the sizing guide. The new functionality will be used by a small group of users. The growth factor is easy to apply. All three are input for the sizing calculation.

  You could have chosen to carry out the complete sizing using the sizing guide. However, in this case a sizing based on measurements is more accurate. The exact customer load curve has to be determined via a theoretical approach. This is a loss of accuracy.

- Sizing input/Questionnaire

  The selected sizing method doesn't oblige a complete filled in sizing questionnaire, unless there is a sizing questionnaire for upgrades available. Some Infor products offer these. It is necessary to complete the questionnaire only for the new functionality Power Engine plans to use.

  On the other hand the questionnaire lists all topics and makes sure that you think about the various subjects.

- Sizing execution

  The first step is to measure the load on the various servers. The four basis components: CPU, memory, disk and network have to be measured. Try to express the system load in BRUs. This can be done by using the sizing guide. The sizing guide shows how many BRUs the specific configuration can handle. The measured system load can be used as a factor to calculate the used BRU load. The sizing guide only shows the latest hardware models; an older sizing guide probably has to be referenced.

  The second step is to investigate how many concurrent users will use the new functionality. Based on the load factor published in the sizing guide and the number of concurrent users the BRU load for the new module can be determined.

  Add the previous two calculated BRU values together and apply the growth factor. This provides the total number of required BRUs.

- Define infrastructure

  Beside the load requirements a hardware vendor, database vendor has to be selected. One has to decide on an operating system and decide how the high availability solution will be build. The project group has already decided on a three-tier configuration, which runs in a cluster. The project group decides for a SAN solution as IO subsystem.

- Evaluate sizing

  Evaluate whether the new infrastructure has sufficient power to serve the load. The configuration will probably work without any problem, because there is a big margin for future growth applied. But make sure that this margin still exists and is not occupied from day one onwards.

Appendix C Use case 2: Sizing based on single-user measurements and extrapolation

C

Introduction

Sizing guides are not available for all products. Sometimes, products are heavily customized and the sizing guide cannot be used. In such situations, it is difficult to carry out a sizing. However, it is possible to carry out a customer specific benchmark, but this is sometimes too big of an investment. In such situations, sizing based on single-user measurements and extrapolation can be an alternative.

Company Self Made Inc. uses Infor ERP software and has many specific software requirements which are not typical for the average customer. Self Made Inc wanted Infor to incorporate these requirements into the standard ERP version. After discussions, it became clear that these requirements were too specific and could only be implemented as customizations. Self Made Inc needed the software solution to operate efficiently. The consequence for sizing is that the standard sizing guide cannot be used.

Sizing approach

You must go through all the steps described in user case 1.

In this user case, only the approach of the sizing method will be explained in more detail, otherwise there is too much overlap between the two cases.

- Investigate the various user profiles.
- Investigate how the system is used during the day.
- Built a test database.
- Key users carry out application functionality which reflects normal business operations; measure the resource consumption.
- Measure response times.
- Calculate the total load.
- Determine peak hour.
- Calculate system.

Carrying out a sizing

Investigate the various user profiles

The used ERP application is a broad application. To determine the application load, not all functionality is important; only the functionality which contributes significantly to the overall load must be investigated. There can be heavy processes or functionality which is may be light, but frequently used. Heavy functionality which is used rarely or run outside peak hours will not contribute to the overall load.

Investigations of the various load profiles at Self Made Inc. delivered the information in the table below; the occasional users are left out.

Functionality investigation

Functionality	**Number of named user**	**When used**	**Maximum concurrent users during peak hour**
Sales order entry	12	Continuous during the day	10
Purchase order entry	15	Continuous during the morning	5
Shop floor order entry	30	Continuous from 05:00h – 24:00h	10
Warehousing	15	Continuous during the day	10
Finance mutation	3	Continuous during the morning	2
Project calculations	5	Continuous during the morning	3
Integration with CRM	1	Once every hour	1
Integration with planning system	1	Once early at 04:00h	1
EDI orders	1	Once every 15 minutes	1
Maintain Sales Schedules	1	Ones every three months	1
Browsing	20	Continuous during the day	15
Business analysis	1	Once a month	1
Closing period	1	Every month end	1

Table 13-1 User load

Investigate how the system is used during the day

Table 13-1 User load shows the peak of the load is during the morning. Statistics collected with the current software also confirms the peak hour is between 10:00h and 11:00h. The assumption is

made that for the new system this will be the same. Note that the highest user amount will not necessarily always cause the highest peak load.

‘Business analysis’ and ‘Closing period’ are rarely carried out. From this perspective you could negotiate the impact, but this functionality seems to be very critical and must be finished within a short timeframe. The outcome of these processes is discussed during a general management meeting, planned every month end; to ensure this, capacity is reserved for these processes.

Test database

To make the tests representative for the customer situation, a realistic test database must be made. For multi-user tests, concurrency/locking is very important and users should not constantly access the same data. This gives an unrealistic representation unless this is a customer realistic situation. In this situation, the concurrency aspects do not play a role because the tests are carried out by single users.

The size of the database should be realistic; larger databases consume more resources. If full table and range scans occur, a large database gives a more realistic picture; this happens during single and multi-user tests.

Key users carry out application functionality and measurements

Key users are asked to work with the application while measurements are carried out. The goal is to determine the load per unit of time. For realistic measurements, it is important that all of the following applies:

- Users should work at their normal speed.
- Users should use the functionality exactly as they would during normal business operations.
- For frequently used application functionality, data should be cached.
- For clean measurements, nobody else should use the system during the tests.
- If users make mistakes, the procedure must be repeated

CPU

How to measure CPU differs per operating system. Some operating systems deliver tools which show graphs; for these types of measurements, it is better to measure the CPU seconds because this is the time a CPU is busy for the process. Note that on fast CPUs, this time is lower than on slower CPUs. Ensure the key-user makes sufficient repetitions through the business to measure significant CPU values. Also note that very low values can have a high deviation error; for example, 0.5 and 1.49 CPU seconds will be rounded to one second.

See the measurement results in the following table:

Measurement results

Role	Steps	CPU [sec]	Lead time [sec]	CPU sec/ minute

Sales order entry	Sales Order header	1.5	60	1.5
	12 Sales Orderlines	3.7	73	4
	Print Sales Order	2.8	120	1.4
	Check stocklevel	0.9	23	2.3
	Create invoice	1.2	204	0.4
Purchase order entry	Purchase Order header	1.4	60	1.4
	25 Purchase Orderlines	6.8	182	2.2
	Check received goods	1.9	230	0.5
	Payments	5.3	123	2.6
Shop floor order entry	Enter shop floor Order	2	1200	0.1
	Complete ten operations	0.7 x 10	1200	16.8
	Book hours	1.9	3600	0
	Complete order	1.3	1200	0.1
Warehousing	Receiving	3.4	189	1.1
	Picking	7.8	100	4.7
	Inter warehouse transport	4.6	4500	0.1
Finance	Check automatic bookings	4	30	8
	Invoicing	2.8	34	4.9
Project calculations	Estimate hours	3	143	1.3
	Planning	56	163	20.6
	Calculate Project costs	78	251	18.6
Integration with CRM	Exchange customer data	2	3600	0
	Transfer quotations	23	3600	0.4
	Transfer sales orders	159	3600	2.7
Integration with planning system	Transfer static data	2	120	1
	Transfer planned orders	117	900	7.8
EDI orders	Transfer sales orders	327	900	21.8
	Transfer sales order confirmations	49	900	3.3
Maintain Sales	Maintain Sales Schedules	14	89	9.4

Schedules				
Browsing	Various programs, data lookup, searching, and browsing	3	37	4.9
Business analysis	Calculate statistics	5300	5557	57.2
Closing period	Finance month closing	4296	4542	56.8

Calculations				
Role	**Steps**	**Number of concurrent users**	**CPU sec/ minute**	**Load per hour**
Sales order entry	Sales Order header	10	1.5	900
	12 Sales Orderlines	10	4	1800
	Print Sales Order	10	1.4	840
	Check stocklevel	10	2.3	1380
	Create invoice	10	0.4	240
Purchase order entry	Purchase Order header	5	1.4	420
	25 Purchase Orderlines	5	2.2	660
	Check received goods	5	0.5	150
	Payments	5	2.6	780
Shop floor order entry	Enter shopfloor Order	10	0.1	60
	Complete ten operation	10	16.8	240
	Book hours	10	0	0
	Complete order	10	0.1	60
Warehousing	Receiving	10	1.1	660
	Picking	10	4.7	2820
	Inter warehouse transport	10	0.1	60
Finance	Check automatic bookings	2	8	960
	Invoicing	2	4.9	588
Project calculations	Estimate hours	3	1.3	234
	Planning	3	20.6	3708
	Calculate Project costs	3	18.6	3348
Integration with CRM	Exchange customer data	1	0	0

	Transfer quotations	1	0.4	24
	Transfer sales orders	1	2.7	162
Integration with planning system	Transfer static data	1	1	60
	Transfer planned orders	1	7.8	468
EDI orders	Transfer sales orders	1	21.8	1308
	Transfer sales orders confirmations	1	3.3	198
Maintain Sales Schedules	Maintain Sales Schedules	1	9.4	564
Browsing	Various programs, data lookup, searching, browsing	15	4.9	4410
Business analysis	Calculate statistics	1	57.2	3432
Closing period	Finance month closing	1	56.8	3408
Total				**33942**

In total, 33942 CPU seconds are required in the peak hour; this translates to 33942/3600 = 9.4 CPUs, which is a high number. From the table above it is derived that 'Business analysis' and 'Closing period' consume 6830/3600 =1.9 CPUs. It is quite expensive to reserve almost 2 CPUs for monthly processes. The decision has been made to run this functionality outside the peak, which means the results are ready a couple of hours later; this is still acceptable. Therefore, the total number of CPUs is 9.4 – 1.9 = 7.5.

The above calculation assumes an even distribution of users across the time; in practice, this will never be the case. Therefore, it is advised to include a margin of 20 percent.

The system used during the tests is quite old. According to the hardware vendor, the CPU speed of the latest system is 2.3 times higher.

This means that (7.5 * 1.2)/2.3 = 3.9 CPUs.

Memory measurements

The memory measurements are listed in the following table. Memory measurement is a difficult subject, so you must ensure the right variables are tested. When the memory per process is measured in single-user mode, ensure the shared sections, such as code sharing, are not measured because these are only loaded once for all users. One option is to start two users with the same functionality and measure the second. Virtual memory also must be carried out from the calculation.

The results can only be used as an estimate. For example, it is very difficult to measure the memory consumption of a database. The more users, the more effective database caching will be. Only experienced values can be used for database on an operating system.

The memory measured is 1866 Mb and the database is estimated on 2 Gb. For operating, a minimum of 512 MB is required according to the specifications of the hardware vendor. It is also advised to reserve 20 percent for peak load. The total memory consumption is: (1866 + 512 + 2048)*1.2 = 5311 MB

Memory consumption			
Functionality	**Maximum concurrent users during peak hour**	**Memory consumption per user [MB]**	**Total memory [MB]**
Sales order entry	10	35	350
Purchase order entry	5	28	140
Shop floor order entry	10	25	250
Warehousing	10	23	230
Finance mutation	2	17	34
Project calculations	3	39	117
Integration with CRM	1	75	75
Integration with planning system	1	74	74
EDI orders	1	103	103
Maintain Sales Schedules	1	16	16
Browsing	15	15	225
Business analysis	1	207	207
Closing period	1	45	45
Total			**1866**