Tivoli. software

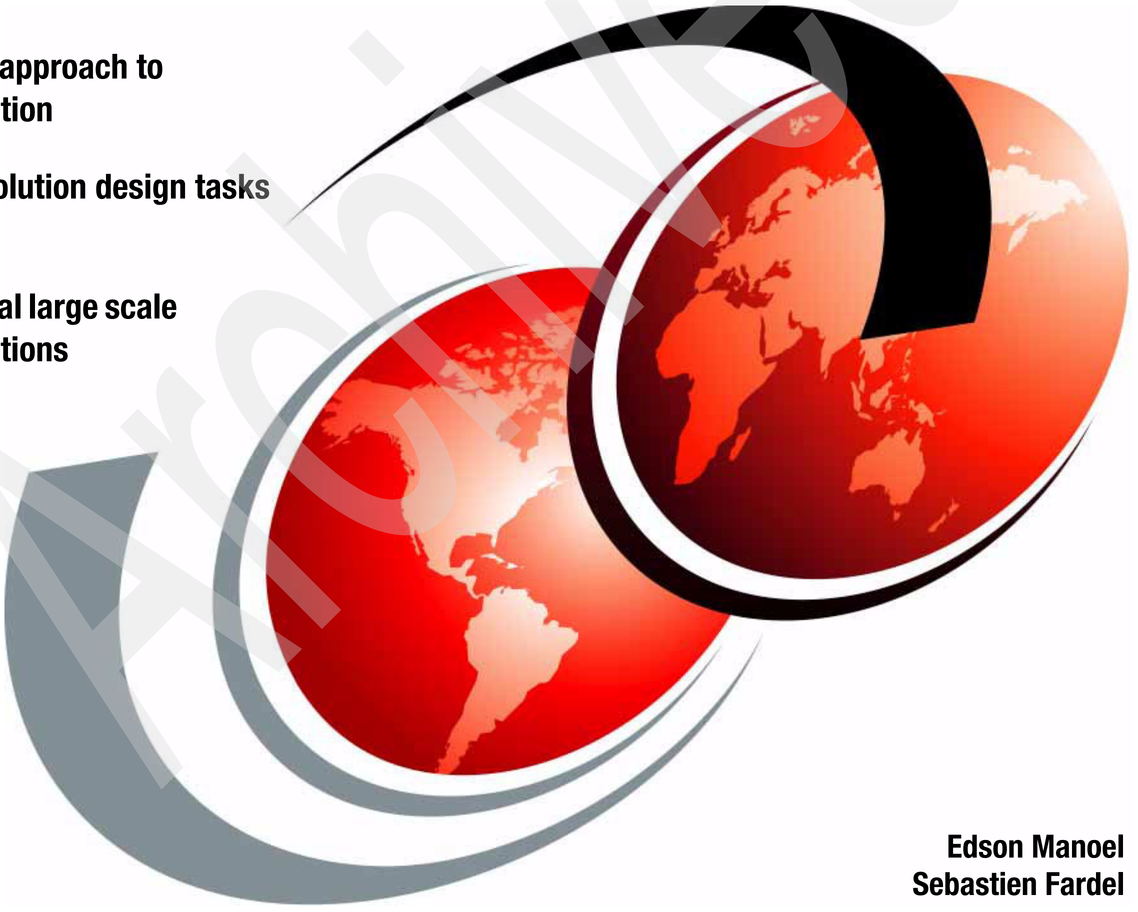
IBM

Implementation Best Practices for IBM Tivoli License Manager

Systematic approach to implementation

Thorough solution design tasks information

Based on real large scale implementations



Edson Manoel
Sebastien Fardel

Redbooks

ibm.com/redbooks

International Technical Support Organization

Implementation Best Practices for IBM Tivoli License Manager

May 2006

SG24-7222-00

Note: Before using this information and the product it supports, read the information in "Notices" on page xi.

Note: This book is based on a pre-GA version of a product and may not apply when the product becomes generally available. We recommend that you consult the product documentation or follow-on versions of this IBM Redbook for more current information.

Attention: This redbook refers to the IBM Tivoli License Manager Version 2.2 product. The IBM Tivoli License Manager product has been recently rebranded as IBM Tivoli License Compliance Manager Version 2.2.

The IBM Tivoli License Manager product name is still in use in this redbook as well as on all of the administrative user interfaces and in the documentation of the product. However, the name IBM Tivoli License Compliance Manager and its acronym ITLCM is also used throughout the redbook in lieu of the old product name and acronym (IBM Tivoli License Manager (ITLM)).

Refer to the product's Release Notes as it provides the most current information about the IBM Tivoli License Compliance Manager Version 2.2 release of the product. The latest version of the Release Notes can be found on the Tivoli Information Center Web site:

`http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?toc=/com.ibm.itlm.doc/toc.xml`

First Edition (May 2006)

This edition applies to IBM Tivoli License Compliance Manager Version 2.2, formerly known as IBM Tivoli License Manager Version 2.2.

Contents

Figures

Tables

Notices

Trademarks

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

AIX 5L™	IBM®	RUP®
AIX®	OS/400®	Tivoli®
Component Business Model™	pSeries®	Virtualization Engine™
DB2 Universal Database™	Rational Unified Process®	WebSphere®
DB2®	Rational®	xSeries®
eserver®	Redbooks™	z/OS®
eserver®	Redbooks (logo) ™	zSeries®
i5/OS®	RS/6000®	

The following terms are trademarks of other companies:

Enterprise JavaBeans, EJB, Java, Java Naming and Directory Interface, JavaBeans, JavaServer, JavaServer Pages, JDBC, JSP, JVM, J2EE, Solaris, Sun, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

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

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

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

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

Preface

The primary objective of this IBM® Redbook is to provide best practices for the implementation of a License Management solution based on IBM Tivoli License Manager V2.2. It can be used as a reference book throughout all phases of a License Management project, and it can guide you on how to effectively deploy IBM Tivoli License Manager V2.2 in a way that quickly generates real business value for clients.

This redbook is a valuable addition to the existing product documentation and should be read in conjunction with the official product documentation, which complements the concepts explained in this redbook. It is divided into two main parts:

In Part 1, "Business aspects and project engagement considerations" on page 1, we discuss the reasons why you should deploy a License Management solution. We also position License Management in the overall picture of the IT Service Management and describe the IBM solution in this area. We also provide best practices on how to approach and engage such a project. The high level tasks of such a project are also introduced. We emphasize the fact that you must not only realize what the key elements are that need to be managed to make your project a success, but also how you should organize your IT operations to get the highest benefit from your solution.

In Part 2, "Solution design considerations" on page 57, we provide detailed information for the architecture and design of a IBM Tivoli License Manager V2.2 solution implementation. Both IT Architects and IT Specialist benefit from the information provided in this part, as it covers a great deal, from an introduction to IBM Tivoli License Manager V2.2, its components, and architecture, and solution design fundamental tasks, to detailed physical and logical design and configuration of all components of an IBM Tivoli License Manager V2.2 solution.

Note: This redbook refers to the IBM Tivoli License Manager Version 2.2 product, which has been recently rebranded as IBM Tivoli® License Compliance Manager Version 2.2.

The IBM Tivoli License Manager product name is still in use in this redbook as well as on all of the administrative user interfaces and in the documentation of the product.

Refer to the product's Release Notes, as they provide the most current information about the IBM Tivoli License Compliance Manager Version 2.2 release of the product. The latest version of the Release Notes can be found on the Tivoli Information Center Web site:

`http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?toc=/com.ibm.itlm.doc/toc.xml`

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Austin Center.

Edson Manoel is a Certified IT Specialist at IBM working in the ITSO, Austin Center, in the systems management area. Prior to joining the ITSO, Edson worked in the IBM Software Group, Tivoli Systems, and in IBM Brazil Global Services Organization. He was involved in numerous projects in designing and implementing systems management solutions for IBM clients and Business Partners. Edson holds a Bachelor of Science degree in Applied Mathematics from Universidade de Sao Paulo, Brazil.

Sebastien Fardel is an Associate IT Architect at IBM Corporation - Global Services - Switzerland, acting as a Systems Management Architect and as an Subject Matter Expert in the Tivoli Automation area. He has been in the IT industry since 1996 and has experience in IT infrastructure management, programming, and the Systems Management area.

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

Wade Wallace
International Technical Support Organization, Austin Center

Fabio Paone
Silvia Giacone
Tommaso Mazzarotto
IBM Software Group - IBM Tivoli License Manager Development, Rome Italy

Matthew Boult
IBM Software Group - Product Introduction, Hursley United Kingdom

Paolo Cavassa
Salvatore Branca
IBM Software Group - Solution Scalability, Rome Italy

Francesco Censi
IBM Software Group - Software Advanced Technology - SWAT, Rome Italy

Carlo Romano
Giancarlo Carbone
IBM Software Group - Product Management, Rome Italy

Lorenza Denaro
IBM Software Group - Information Development, Rome Italy

Paolo Franz
IBM Global Services - Manager, Bussigny Switzerland

Lise Crettaz
IBM Software Group - Sales, Geneve Switzerland

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or clients.

Your efforts will help increase product acceptance and client satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

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

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

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

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

  `ibm.com/redbooks`

- Send your comments in an email to:

  `redbook@us.ibm.com`

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. JN9B Building 905
  11501 Burnet Road
  Austin, Texas 78758-3493

Part 1

Business aspects and project engagement considerations

In this part, we position Software Asset Management from the ITIL and IBM ITSM perspective and give an overview of the business challenges of software assets management and their License Management issues.

We also provide information about project and engagement considerations for the implementation of License Management processes.

1

Business perspective for License Management

This chapter discusses the business rationals and justifications for implementing License Management in an enterprise. It also describes the potential business impacts of not managing the software assets and how to approach the problematic by implementing Software Asset Management, which is part of the overall IT Service Management.

The following topics are discussed in this chapter:

1.1 Business challenges of License Management

While business organizations are focused to improve and optimize their Information Technology (IT), or Information and Communications Technologies (ICT), infrastructure to better align it to their business and decrease associated costs, software is recognized as one of its most critical elements.

Software, whether internally developed or externally procured, is one of the most important and costly part of the IT budget, and therefore requires special attention from business executives. This attention should not be limited to cost control, but should also be extended to compliance monitoring as well. As most software is subject to contractual and legal statements, and regulatory governance requirements, such as Sarbanes-Oxley in US and Turnbull in UK, business executives must also ensure that their enterprise is not exposed to legal actions, financial penalties, or reputation loss.

The three main factors driving the need for a comprehensive License Management solution are:

- Compliance with regulations and governance
- Management of risks on the business operations
- Management and optimization of software costs

In order to help you better understand these factors and their impact on your enterprise, you could ask yourself the following questions:

- Does my enterprise have enough licenses, meaning that my enterprise could not be compliant with the regulations and agreed terms and conditions of license contracts?
- Does my enterprise have too many licenses, meaning that my enterprise is paying for unused software?
- Do the end users of my enterprise run the software *only* on authorized systems?
- Do the end users of my enterprise *only* use the software they are authorized to, meaning they are not using any pirated or illegal software?
- Do the end users of my enterprise use the software licenses they are entitled to, no more (in order to avoid exposure) no less (to optimize the cost)?

By answering the listed questions above, you should be able to evaluate your requirement for a comprehensive License Management solution.

The following sections discuss the key factors mentioned above in more detail.

1.1.1 Compliance with regulations and governance

Regulations and standards are proliferating, making compliance requirements more difficult to manage. This brings a new level of complexity to the IT organizations, which are already hard-pressed to cost-effectively manage the IT supporting the core business processes. Capturing, storing, analyzing, and leveraging data required for compliance could increase resource and people overload, and the costs if the process is not automated.

Compliance could be defined as the process of adhering to a set of guidelines or rules established by internal or external organizations, the second often being a driver for the first. For example, internal organizations may define governance and policies to ensure that they will be compliant with any possible external audits, which may occur with very little notice, sometimes around 30 days before the audit. Organizations like the Business Software Alliance (BSA) use detective work to evaluate the volume of the required license. If the license records they own does not correspond to the evaluation made, they initiate a compliance audit procedure.

So, more and more, enterprises need to meet strong regulations to prove they are managing their operations with respect to the laws and for the shareholders' protection. The following list provides some of the known regulations that enterprises must be compliant with:

- Sarbanes-Oxley (SOX)
- Health Care Information Portability and Accountability Act (HIPAA)
- California Act
- Basel II
- CLERP-9

The objective of this redbook is not to present all the regulations that require compliance. However, it is important to understand to which penalties your enterprise could be exposed to and what usually are the requirements driving a License Management project. So, we selected one of these regulations and try to provide you with a brief overview of what could it mean to not be compliant with regulations.

Sarbanes-Oxley regulation overview

SOX stands for the Sarbanes-Oxley Act of 2002. This act has been part of United States federal law since July 30, 2002 and has been promulgated in reaction to some corporate collapses. It is now applicable to all public companies registered at the United States stock exchange.

The goal of this act is to protect investors by improving the accuracy and reliability of corporate disclosures made pursuant to the securities laws, and for other purposes.

The new corporate responsibility for financial reports is listed in section 302. The Chief Executive Officer (CEO) and the Chief Financial Officer (CFO) must certify the accuracy of annual and quarterly reports under penalty of law. They may incur civil or criminal penalties for violations of this act.

SOX section 404 requires, among other things, that corporations establish and maintain an adequate internal control structure and procedures for financial reporting and assessing the effectiveness of internal control over financial reporting. An external auditor must report on the existence and condition of internal controls linked to financial reporting. Further reports must be provided on all significant deficiencies in the design or operation of internal controls, as well as any fraud that involves management or other employees with a significant role in the company's internal controls, and any changes in internal controls.

The Committee of Sponsoring Organizations (COSO) of the Treadway Commission is one of the frameworks used by management for internal controls. The auditors also look at the Control Objectives for Information and related Technology (COBIT) and Information Technology Infrastructure Library (ITIL) frameworks, which is focused on IT.

IT does affect the reliability and security of systems in which companies keep their financial reports. Organizations must understand how the financial reporting process works and must identify in which areas IT is playing a critical part. From a SOX point of view, only the systems linked to financial aspect or related business processes will be taken in consideration.

Compliance with regulations approach

Obviously, the responsibility of the CEO and CFO must be delegated downward through the organization to those dealing directly with internal controls, meaning a hefty responsibility is laid on the shoulders of the IT managers.

A starting point for being compliant is to identify and document processes, procedures, and internal controls. That exercise will lead to evaluation of the effectiveness of these processes, identification of their deficiencies, and implementation of improvements.

Even if this kind of exercise may require a lot of work, it could bring added value to the entire organization, as the review and reporting leads to tighter controls and greater efficiency.

So, the regulations provide the right opportunity, even if it a forced one, to work or rework on the compliance process, which could result in a better automation of some global operations. In addition, the usage of an IT Processes framework, like ITIL or COBIT, which provides good Best Practices, may be a good starting point.

Besides optimizing the compliance process, implementing adequate dynamic technology could help make the process actionable, automate the operational tasks, and decrease, in real time, the risk of non-compliance by decreasing the business costs.

1.1.2 Risk management on business operations

As described in the previous section, the main risk regarding License Management at the business level for the enterprise is related to compliance with regulations and governance. Nevertheless, a non-existent or non-efficient Enterprise License Management process could have other effects on the business of the enterprise, mainly at the business operations levels. The following list summarizes some of these risks:

- Besides the damaged reputation caused by legal problems becoming public, the required penalties could have an unexpected financial impact. In addition, the manual efforts required to prove the compliance drastically increase the workload and the associated costs of the IT organization.
- Such financial impact may also arise from an acquisition, a merger, or the breakup of a company into separate entities. Those activities require strong due diligence in the licensing area. Not having a solution in place escalates the possible issues that may expose the organization to significant unexpected financial risk.
- Even if a limited number of software vendors engineer license enforcement protocols in their product, some business critical tools may become unusable due to an enforcement policy if the maximum of authorized licenses is reached. This situation generally results in a loss of productivity for the organization, as employees are no longer able to use their tools.
- Generally, the software support provided by vendors is strongly linked to the software licensing aspects. This means not being compliant with the terms and conditions of the licenses may expose your enterprise to a situation where you cannot obtain technical support for the software. This could cause critical problem if your business processes are linked to those tools.

Having a good process in place to manage your software assets may help you to mitigate the risks mentioned above on your business operations.

1.1.3 Software cost management and optimization

Software accounts for around a quarter of the IT budget, and, based on latest trends and reports, software budgets are increasing faster than other budget areas.

If we compared the escalation of hardware and software costs in the IT organization, we would discover that hardware costs are decreasing while the power provided by this new hardware is increasing. This fact could have a positive impact on software costs, as they are often based on hardware capacity. The number of hardware assets per employee is more or less low. In comparison, and as mentioned in the paragraph above, software costs are increasing, and the number of instances of software installed for or used by employees is also increasing.

However, the management of Software costs is often cited as one of the lowest priority IT projects. This could be explained because solutions often deployed to manage software costs did not provide the expected result, because the solution is not integrated with all the necessary processes or because it is hard to identify software usage and what software is unnecessary.

The following list provide some key points of what an optimization of your software costs may bring to your enterprise:

- Being correctly aligned with the terms and conditions of the software contracts as well as being able to prove license compliance are the most compelling reasons for negotiating with software vendors. This could result in great software costs savings. Conversely, a lack of clarity about the correctness of licensing offers an advantage to the software vendors during the negotiation.
- In the same area, being able to centralize the software assets information provides a better view of the real needs, in term of tools, to support business requirements and may also result in software licensing information consolidation. It is common for large enterprise to have selected strategical tools to support their business, but due to decentralization of the software assets management, some other tools, supporting the same business requirements, may be deployed. Leveraging the strategical tool could result in replacing the other tools at little or no additional costs. Even better, this will not only help to save costs of the nonstandard tools, but also the related operational costs required to install and support these tools.

  This centralization and consolidation of software licensing information would be even better if the procurement process is also centralized. Your enterprise, by having a global view, may benefit by volume purchasing and better negotiations could be engaged with software vendors in regards to enterprise license agreements.

Finally, this consolidation approach helps to improve the strategic infrastructure planning. It will be easier to prepare the capacity plan for software that is required to support business needs. This could also offer a chance to reduce costs through competitive replacement. In addition, the infrastructure changes will then be simplified as well.

- Another area where software costs could be optimized is the identification and reduction of no use or low use software. Often, a standard configuration is defined for working platforms. This definition, based on common findings, is over-specified compared to what end users actively use. Better analysis of the business requirements and categorization of software can significantly reduce software and, as a consequence, hardware requirements and costs. The software licensing approach for desktops or mobile computers is mainly based on the principle that one license corresponds to one installation, whether the software is used or not. Our approach would be to define at least three categories of software as follows:

  - Category One for base platform software

    Tools that are part of this category are automatically installed on all platforms.

  - Category Two for business divisions or functional requirements.

    Tools that are part of this category are only installed on machines that are part of the business lines in which these tools are required.

  - Category Three for on demand requests.

    With pull technology, it is easier now to offer on demand capability to end users. This will not only help to decrease the number of installations, but will help to:

    - Analyze the business requirements for having such software.
    - Install this specific software according to predefined release and configuration management processes.
    - Help to charge this software back to the business unit, department, or individual who needs it.
    - Avoid installation of non-authorized software.

- In organizations where costs management is in place and is important, chargeback can be a key topic as well. Without a performing License Management solution deployed in the enterprise, charge back could be overstated. In addition, because chargebacks are rarely based on true software usage, they become inaccurate. This may lead overcharged clients to often search for alternative solutions outside the IT organization.

Currently, the business organizations expect that IT organizations become on demand structures and align their strategy and operations to their business needs. Unfortunately, these business organizations often have IT organizations that are not very flexible in their approach to aligning with business needs. In the area of License Management, this is mainly due to the fact that contracts are static, based on historical capacity and not based on real usage, often resulting in inappropriate expense projections, and creating skepticism in business structures. In this context, business organizations are not able to better manage or optimize their costs as they do not really pay for what they use or need. This results in situations where these organizations stop investing in IT supported projects and develop their own infrastructure or outsource these kind of tasks to external providers.

1.2 Software Asset Management and License Management

Software assets are a key component in the IT equation. Maintaining those software products in a large infrastructure, which can include thousands of servers and desktops, is no trivial task. There is no universal, correct solution to manage software assets but implementing governance and processes based on an industry standard like ITIL will help your enterprise to face this challenge.

So, even if this redbook is dedicated to the implementation of a License Management tool, understanding and controlling the broader context in which a License Management project is executed is the key success factor. This statement is based on lessons learned from large scale implementations of License Management projects.

1.2.1 Software Asset Management

Software Asset Management (SAM) can be interpreted differently by people and has multiple definitions. In order to have a common understanding of this term, we have decided to use the definition provided by ITIL. All demonstrations and explanations in this redbook related to Software Asset Management are based on the following ITIL definition:

"Software Asset Management (SAM) is all of the infrastructure and processes necessary for the effective management, control and protection of the software assets within an organization, throughout all stages of their life cycle. SAM does not include Hardware Asset Management."[1]

[1] Source: Reprinted, by permission of the publisher, from *Software Asset Management*, The Staionery Office, 2003

As mentioned previously, all mention of Software Asset Management in this redbook relates to the definition provided by ITIL. In later sections, we provide a brief overview of Software Asset Management from the ITIL perspective as well as how Software Asset Management is positioned in IBM IT Service Management initiative.

1.2.2 License Management

Software Asset Management is how you manage the Software Assets within your enterprise. An important aspect of SAM is License Management. Software licenses are rights to use software, with certain terms and conditions attached, and are one of the main issues addressed by Software Asset Management.

The rights to use software are totally separate from the legal rights to the software itself, which are normally kept by the software manufacturer or other third party. Licenses may be bought, or may be free, but always subject to special terms and conditions. Even open source software normally has a license agreement attached to it, even though payment may not be required.

Licenses are normally required whenever externally sourced software is used. The term, used in this context, is typically defined as software either being installed on a machine, or being executed on a machine, even if installed elsewhere (for example, a server). They may also be defined in enterprise terms, such as number of workstations or employees, in which case a license is required for each qualifying unit or individual regardless of actual usage.

Even with commercial software, there are several situations where paid licenses may not be required, depending on specific contractual conditions. Often, these situations are not understood and, as a result, organizations may purchase licenses they do not need. These situations include workstations used for dedicated training purposes (with limits on numbers), copies used for evaluation purposes (with conditions on how they are used, and for how long) and copies used for distribution purposes. Likewise, there can be runtime versions of some software, which do not require separate paid licenses. (It may be difficult to distinguish between runtime and non-runtime versions of such software.)

Software licensing is complex. Compliance with all of its terms and conditions requires in-depth knowledge. Typically, an organization will need to assign the responsibility for understanding licensing to specific individuals, and then ensure that they have the necessary training (initial and ongoing) to master the area.

1.3 Software Asset Management in ITIL

Software Asset Management is only a part of an IT process model. In order to clearly position the Software Asset Management in the overall IT organization, it is important to have an overview of the concept of IT Service Management. and how SAM is positioned in regard to the different management frameworks.

1.3.1 ITIL's IT Service Management overview

Before presenting the global IT Service Management picture of ITIL, it is important to first provide some basic information about ITIL.

The ITIL is a series of documents that are used to aid the implementation of a framework for IT Service Management. This customizable framework defines how service management is applied within an organization.

The ITIL was originally created by the Central Computing and Telecommunications Agency (CCTA), a United Kingdom Government agency (currently known as the Office of Government Commerce (OGC)). It is now becoming more popular and has been adopted and used across the world as the standard for best practices in the provision of IT service. Although the ITIL covers many areas, its main focus is on IT service management.

Even if ITIL is most popular in the area of Service Support, which is the practice of those disciplines that enable IT services to be provided effectively, and Service Delivery, which covers the management of the IT services themselves, it covers many areas critical to IT Service Management.

Figure 1-1 on page 13 provides a complete overview of ITIL's coverage.

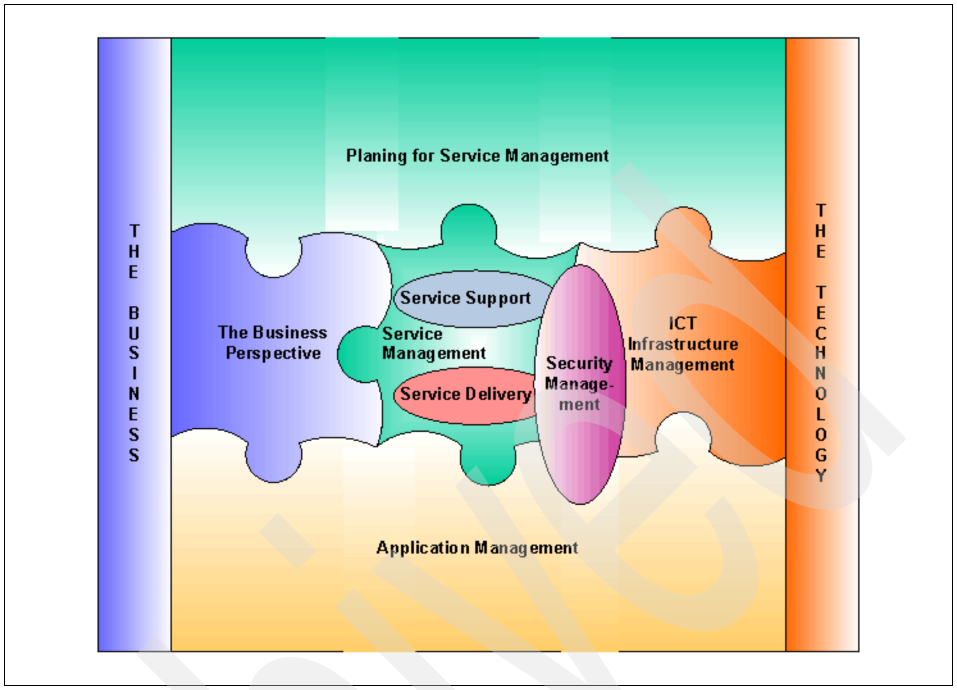


Figure 1-1 The ITIL publication Framework

The individual areas can be briefly described as follows according to ITIL definitions:

- Planning to Implement Service Management

  This area covers the planning of Service Management processes, together with the development of organizational and Information and Communications Technologies (ICT) cultures.

- Service Management

  This area consists of two guides:

  – Service Delivery

    It covers the processes associated with the development and improvement of the quality of ICT services delivered to the business, consisting of Service Level Management (SLM), Financial Management, Capacity Management, IT Service Continuity Management, and Availability Management processes.

  - Service Support

    It describes the function and processes involved in the day-to-day support and maintenance of ICT services, consisting of Incident Management, Problem Management, Configuration Management, Change Management, and Release Management processes, as well as the Service Desk function.

- ICT Infrastructure Management

  This area describes all of the processes associated with the management of the ICT infrastructure, including overall management, Design and Planning, Deployment, Operations, and Technical Support.

- Application Management

  It includes all of the processes and issues associated with the development and management of applications and software life cycles.

- Security Management

  It covers all of the processes and issues associated with the security of ICT services and systems.

- Business Perspective

  This area focuses on the processes of business alignment and communication associated with ICT systems and services.

Refer to the following Web sites for details about what ITIL is and what it can provide:

- Official ITIL Web site:

  http://www.itil.co.uk

- IT Service Management forum Web site:

  http://www.itsmf.com

- Official OGC Web site:

  http://www.ogc.gov.uk

1.3.2 SAM positioning in ITIL's IT Service Management

Usually, and as stated in the ITIL definition of SAM, Software Asset Management does not include Hardware Asset Management. Software Asset Management and Hardware Asset Management together can be referred to as IT Asset Management (ITAM) from an ITIL perspective. IT Asset Management is a subset of IT Service Management.

Generally speaking, and according to the ITIL statements, SAM is more complex and more demanding than Hardware Asset Management. Therefore, the SAM processes need to be greater in scope and more comprehensive in content. As a result, systems that can handle SAM can normally be expected to handle Hardware Asset Management as well. The management of hardware assets is essential as well. Nevertheless, it is important to also notice that hardware costs tend to remain fairly stable and have been declining over the years as opposed to software and contracts, whose costs are on the rise and remain unstable.

According to the ITIL statement, Software Asset Management is a high expense for a company as well as being the most difficult component to manage, but it offers a way to handle management in a cost-effective way.

The following ITIL areas are associated with the operation of SAM precesses:

- Service Management
- Security Management
- Application Management

The ICT Infrastructure Management and Business Perspective areas are also involved but to a lesser extent. The management of contracts and suppliers is also a key area for both SAM and ITIL.

The ITIL organization developed a specific book related to the management of software, *Software Asset Management*, by The Office of Government Commerce Staff.

Figure 1-2 positions the Software Asset Management in the overall picture of the ITIL's IT Service Management.

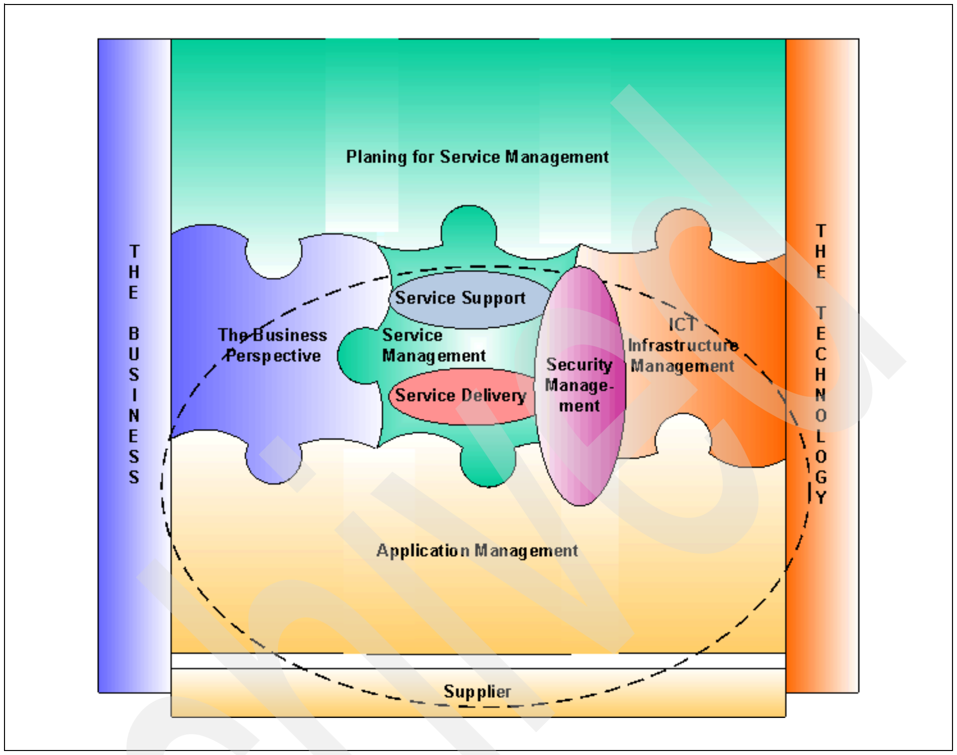


Figure 1-2   Software Asset Management in ITIL's IT Service Management

1.4 Software Asset Management in IBM IT Service Management

Software Asset Management is an important topic within the IBM IT Service Management initiative. This section helps you to clearly position Software Asset Management in the overall IBM IT Service Management framework.

1.4.1 IBM IT Service Management overview

The evolution in IT management has led to resource-oriented silos, such as database experts and tools, network experts and tools, and so on. This leads to inconsistencies and duplication of processes, thus resulting in increased labor costs.

Clients need to build on top of the resource management assets to develop horizontal, repeatable, and process-oriented solutions that can achieve higher levels of efficiencies in their organizations.

This does not necessarily mean reducing the number of head count. It means doing more with the same.

Today, technical silos of expertise are not effective in managing a composite application infrastructure. Application visualization is provided by front-end Web servers, and new business logic is being deployed on a new infrastructure of applications servers and these have been integrated with existing systems and applications. The composite application infrastructure is going to get more complex with the implementation of service-oriented architectures (SOA).

Currently, most application interruptions or performance problems are reported to IT by end users of the business service. The challenge remains the same: to provide the right information to the right people at the right time so they can take action, but how that information is moved across the silos of technical expertise within IT is going to change.

What is needed today is a platform that will integrate the core process by which IT operates. These integrated processes will serve as the new platform for IT Service Management, reducing the labor cost associated with managing IT Services and increasing the effectiveness of IT operations.

The potential benefits are huge. Much is spent on IT *In House Labor* and more than 80% is allocated just to keep IT up and running. Just imagine the amount of savings if IT can gain even 10% productivity in the execution of their internal processes.

To improve efficiency, IT organizations need to look at IT as a whole and build on top of resource management. IT Service Management is the optimal intersection of *people*, *process*, *information*, and *technology*. When these come together, it is easier to break down silos and make IT more efficient and effective.

Figure 1-3 describes the optimal approach of IT Service Management.

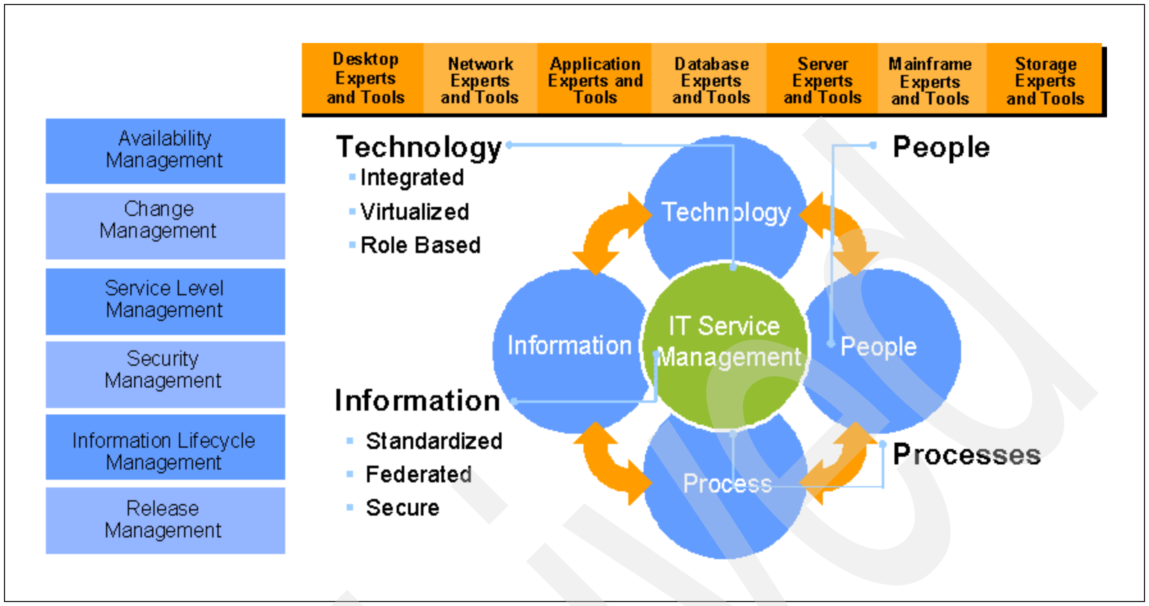


Figure 1-3 Optimal IBM IT Service Management Approach

IBM has taken a comprehensive approach to IT Service Management, which spans what clients want from a Business Model down to the Technologies and products. IBM has assets that could help the client solve their problems at each layer, such as the Component Business Model™ for the Business of IT, Process Models, and also the required tools and methodware.

The IBM IT Service Management approach in term of processes is based on ITIL. However, ITIL addresses processes at a high level and does not provide a prescriptive approach to putting them in operation. IBM, with its approach, makes ITIL actionable by defining a process reference model and also defining a technology reference model.

This helps the client to understand these processes in the context of systems management tools, which can help automate tasks as well as high-level processes.

Figure 1-4 refers to some acronyms that requires some definition:

- RUP®

  IBM Rational® Unified Process®, or RUP, is a configurable software development process platform that delivers proven best practices and a configurable architecture. It enables the enterprise to select and deploy only the process components needed for each stage of an enterprise project.

- ITUP

  IBM Tivoli Unified Process is the first navigational tool to provide the *how-to* for customizing and implementing best practice for mapping, modifying, and improving IT processes. This is the first-ever solution to prescribe specific actions for ITIL, and is a widely used industry guide of best practices for IT processes, through *tool mentors* that help clients improve IT process integration. Clients learn how to alter a system, how to put the right steps in place, and what software and hardware are required. This online tool is available now at the following URL:

  `http://www.ibm.com/software/tivoli/features/it-serv-mgmt/itup/index.html`

Figure 1-4 shows the IBM IT Management Reference Model, which is a real, cross-IBM Initiative.

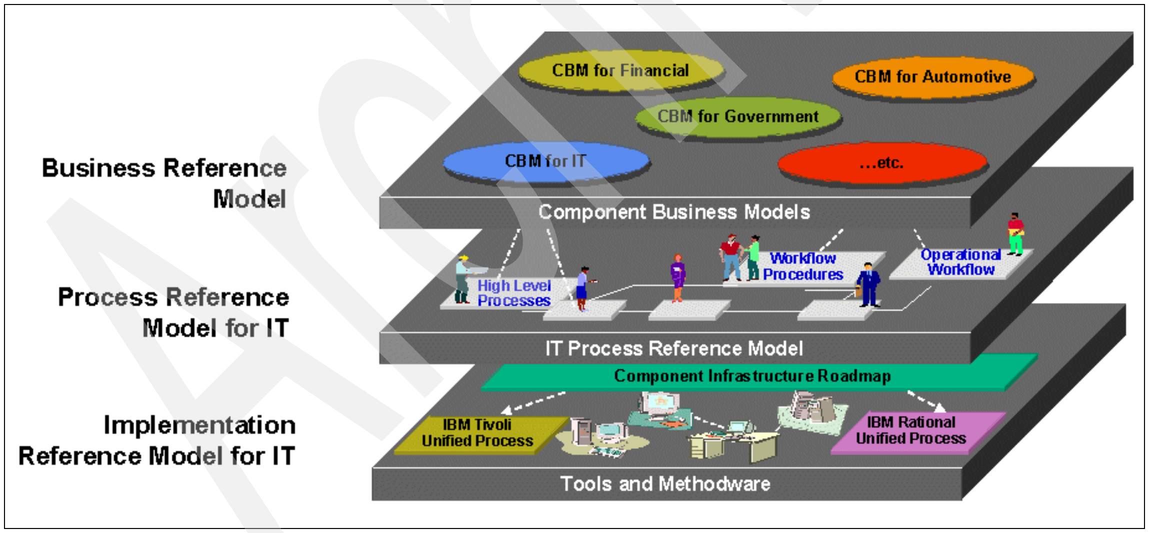

Business Reference Model
Process Reference Model for IT
Implementation Reference Model for IT
CBM for Financial
CBM for Automotive
CBM for Government
CBM for IT
...etc.
Component Business Models
High Level Processes
Workflow Procedures
Operational Workflow
IT Process Reference Model
Component Infrastructure Roadmap
IBM Tivoli Unified Process
IBM Rational Unified Process
Tools and Methodware

Figure 1-4 Comprehensive IBM IT Service Management Approach

The IBM IT Service Management strategy adds a layer of process, data integration, analysis, and automation on top of existing silo-specific management tools and processes. It relies on a federated CMDB, integrated change and configuration process automation tools, and shared workflow and policy management tools.

Figure 1-5 depicts the IBM approach in the IT Service Management.

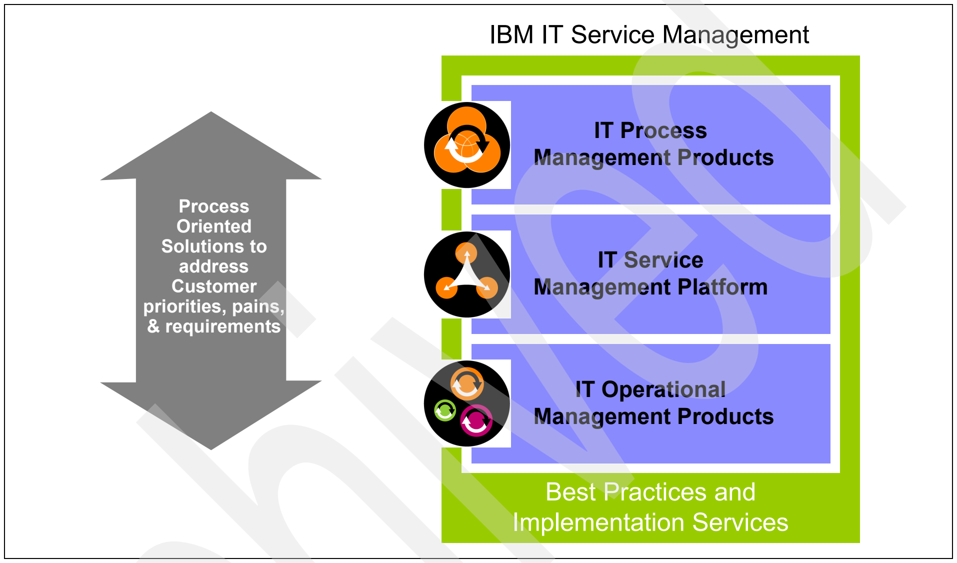

IBM IT Service Management
Process Oriented Solutions to address Customer priorities, pains, & requirements
IT Process Management Products
IT Service Management Platform
IT Operational Management Products
Best Practices and Implementation Services

Figure 1-5 IBM IT Service Management overview

The IBM IT Service Management approach includes:

- ITIL-aligned Process Models and Workflows out-of-the-box allow you to automate and integrate processes rapidly.
- Configuration Management via a Configuration Management Database (CMDB) and workflows that allow you to integrate, standardize, and share information across these tasks and tools.
- Automate specific tasks in a process through integration with today's IBM Tivoli Automation, Security, and Storage products.

IBM has now a very comprehensive and flexible IT Service Management Portfolio. Figure 1-6 on page 21 shows the broad categories of IT operational management products and IT process management products offered by IBM Tivoli.

In between these two sets of products is the IT Service Management platform that integrates them and contains the Change and Configuration Management Database (CCMDB).

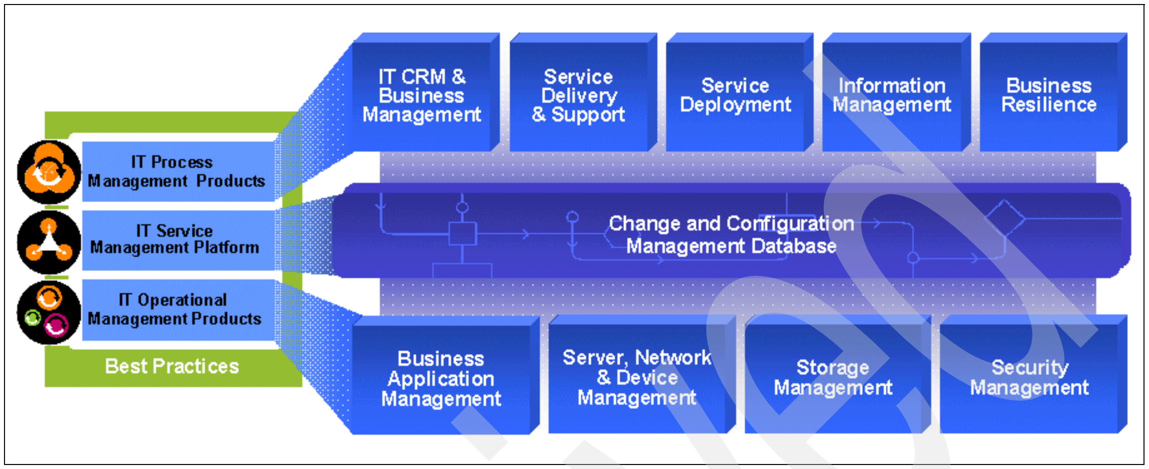

IT CRM & Business Management
Service Delivery & Support
Service Deployment
Information Management
Business Resilience
IT Process Management Products
IT Service Management Platform
IT Operational Management Products
Best Practices
Change and Configuration Management Database
Business Application Management
Server, Network & Device Management
Storage Management
Security Management

Figure 1-6 Flexible IBM IT Service Management Approach

In addition to this approach, IBM has leveraged its experience and best practices into designing the above three solutions and IBM also provides packaging and installation services.

Refer to the following Web sites for details about IBM IT Service Management:

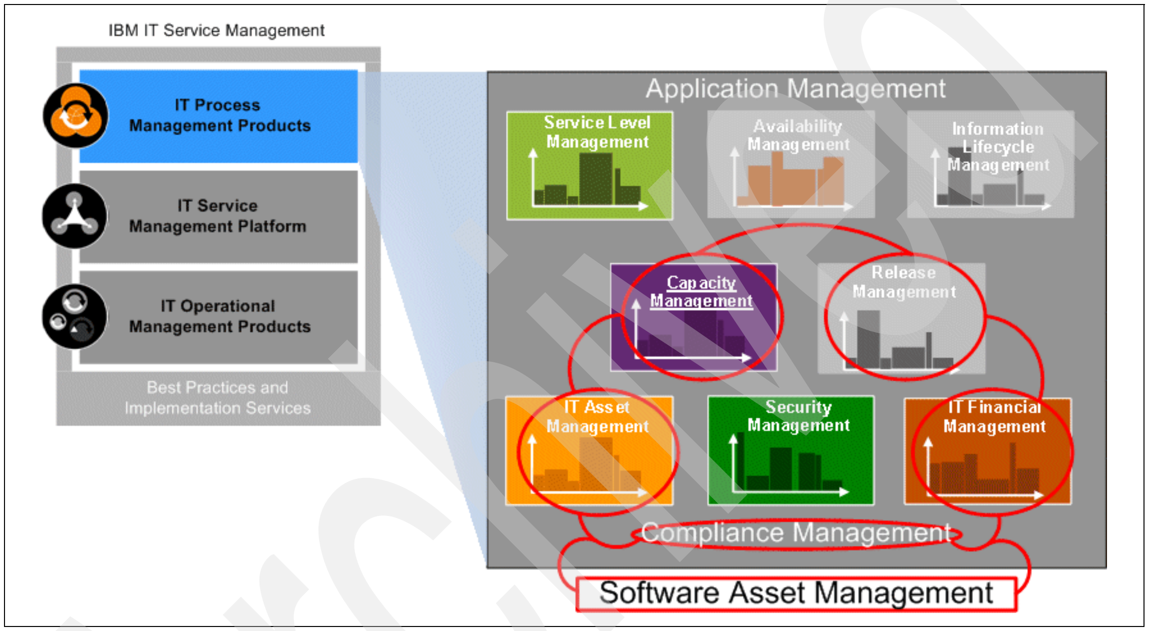
`http://www.ibm.com/software/tivoli/features/it-serv-mgmt/index.html`

1.4.2 SAM positioning in IBM IT Service Management

Software Asset Management's ability to analyze actual software usage and manage related contracts makes it a fundamental part of IBM IT Service Management. As part of IBM IT Service Management, Software Asset Management allows a business to transform from a siloed Software Asset Management approach to one aligned with the overall business operations. Some of the benefits of this transformation to IT Service Management include reducing IT software cost, improved planning capabilities, business and IT integration, optimized capacity planning, and more efficient release and change management. Software Asset Management as part of IT Service Management's processes allow you to manage software licenses and contracts based on business priorities, account for IT assets, and prove contract compliance, thereby demonstrating and maintaining compliance with Sarbanes-Oxley and other regulations.

The IBM SAM solution will also propose an integrated data storage with the IBM Tivoli CCMDB. This CCMDB connection not only offers a single cross-platform view of the software license inventory and usage, but also a view of the entire IT Asset Management status, as it will be possible to get combined reports between SAM information and other information, like operating systems and applications captured from such products as IBM Tivoli Configuration Manager.

Figure 1-7 presents the Software Asset Management positioning in IBM IT Service Management.



IBM IT Service Management
IT Process Management Products
IT Service Management Platform
IT Operational Management Products
Best Practices and Implementation Services
Application Management
Service Level Management
Availability Management
Information Lifecycle Management
Capacity Management
Release Management
IT Asset Management
Security Management
IT Financial Management
Compliance Management
Software Asset Management

Figure 1-7 Software Asset Management in IBM IT Service Management

As depicted in Figure 1-7, Software Asset Management in IBM IT Service Management spans several primary responsibilities.

- IT Financial Management

  IT Financial Management in IT Service Management deals with understanding the costs of IT software. Software Asset Management therefore is a key component of the IT Financial Management process, as it allows clients to fully account for all IT software expenditures across the enterprise. With SAM in the IT Finance Management process, clients can control and manage their IT software budget while at the same time assisting them with software decisions and budgets. A secondary benefit is that a SAM solution enables the allocation of software costs to the users receiving them.

It provides the framework for recording the costs of software, so that the licenses can be allocated and costs recovered for the IT service to the user.

- Release Management

  Release Management in IBM IT Service Management is the process of performing a successful software or hardware change in an IT environment. SAM's involvement in this process is primarily, in the front end, in confirming that the software release is within the contract policy and does not have any interdependencies with other IT assets. The SAM piece of Release Management is critical for the planning and assessing portion of the software release cycle.

- Capacity Management

  Capacity Management is an area concerned with ensuring that IT processing meets the evolving needs of the business. A Software Asset Management solution allows you to understand where you are over-spending and under-spending on software licenses so that you can shift spending to align with the business needs. A SAM solution provides a current analysis of the software usage so that decisions can be made on actual usage rather than static contracts. In this way, a SAM solution ensures that the software license capacity of the company meets the current and future needs of the business.

- IT Asset Management

  Software Asset Management is a component of IT Asset Management. The ITIL definition of IT Asset Management is Software Asset Management and Hardware Asset Management together. For IBM, IT Asset Management is made up of three components: hardware, software, and contract. Out of the three components, SAM is involved with software and contracts.

Figure 1-8 shows how SAM is positioned in the IBM IT Asset Management.

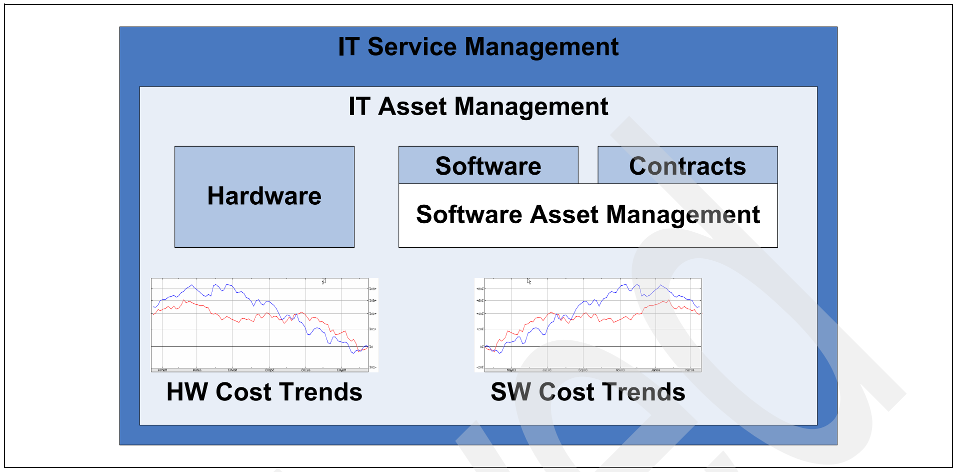


Figure 1-8 Software Asset Management in IBM IT Asset Management

In general, hardware costs tend to remain fairly stable and have been declining over the years, as opposed to software and contracts, whose costs are on the rise and remain unstable. Software asset management is a big expense for a company, as well as difficult to manage.

- Compliance Management

  The IBM Tivoli Software Asset Management solution provides the ability to audit and account for IT assets, software, and expenditures, which forms the foundation to meeting the mandates of numerous regulations. It is IBM's intention to fully integrate its Software Asset Management solution with IBM's other compliance solutions to provide a common infrastructure helping companies with their compliance activities.

The IBM Tivoli SAM solution provides a process to automate, measure, and audit software and IT assets, helping you to manage your regulatory compliance requirements.

1.5 Software Asset Management in other management frameworks

As it was described in sections above, Software Asset Management has been positioned in relation to ITIL's IT Service Management and IBM IT Service Management, which follows, as close as possible, the ITIL standards. However, Software Asset Management could also be positioned in regards to other, well-known management frameworks.

Two other leading open guidelines and standards frameworks in the area of IT Service Management are:

- The British Standards Institution's (BSI) British Standard 15000 (BS 15000)
- The IT Governance Institute's Control Objectives for Information and related Technology (CobiT)

There are also several proprietary management frameworks. As they are all more or less aligned to ITIL, the SAM positioning in ITIL's IT Service Management should fit in these management frameworks as well.

Nevertheless, the message provided in the ITIL Software Asset Management Book regarding integration of SAM in management framework is key:

It is imperative that there is end-to-end continuity and integration between the various constituent SAM processes, whether an organization is based upon ITIL, BS 15000 or CobiT principles and guidelines.

Please refer to Chapter 9, "Mapping SAM to ITIL and other approaches", in *Software Asset Management*, by The Office of Government Commerce Staff.

1.6 Software Asset Management life cycle

Based on its own experience and the one IBM was able to collect during real implementation of SAM, the three main components that are part of a successful SAM system (Figure 1-9), which are the right tools, the right process, and having buy-in, are equally important in sustaining a successful IT Asset Management system.

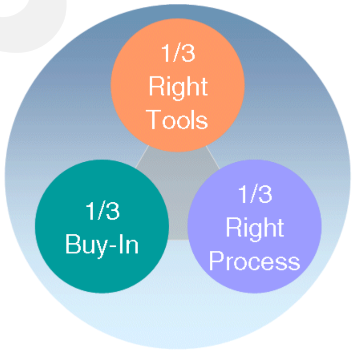


Figure 1-9 Sustaining SAM Success

Software Asset Management has a life cycle that operates around the three components of successful management. SAM is a continuous effort to manage, with efficiency, the software of your enterprise; implementing the SAM life cycle will insure the success of an efficient management of software assets within your enterprise.

Figure 1-10 gives the overall picture of the SAM life cycle.

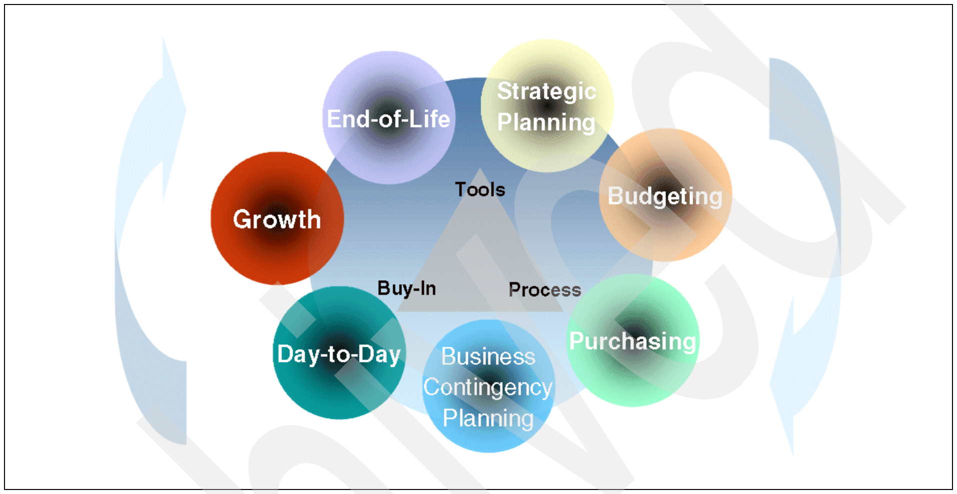


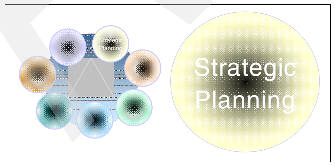
Figure 1-10 Software Asset Management life cycle

Starting with the strategic planning and ending with the proper termination of software, SAM helps in all of the phases by providing the right information, tools, processes, and effectiveness.

By following all seven phases of the SAM life cycle, the tools, buy-in, and process triangle is completed, and you can begin to experience substantial savings and efficiencies.

Each of the seven phase is briefly described in the following sections.

SAM Life Cycle - Strategy Planing

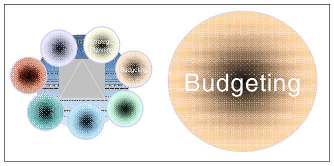


Establishing a strategy for the organization of Asset Management can be a very expensive process for an enterprise. To help define and carry out this strategy, the IT organization should have all the required tools and processes to provide critical information to the executive management. This means the IT organization needs to be able to clearly depict what software exist in the enterprise, what software is being used, and if it is being used by the right people. Based on this, the executive management can determine if the corporation is working cost effectively with regards to its strategic plan.

This Strategy Planning phase of SAM helps to:

- Quantify the utilization of software that supports an organization's adherence to corporate strategies.
- Optimize where the software is being used, and by whom.
- Identify and investigate software expenses.

SAM Life Cycle - Budgeting

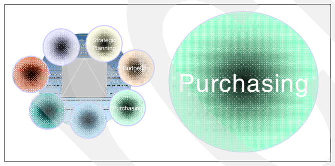


One major concern of an enterprise is to adequately define its software budget for the future so that the enterprise can keep its competitive edge without wasting funds on inefficient and unnecessary Software.

The Budgeting phase of SAM helps to:

- Accurately track what actually is used, and know what software to budget for and what software to eliminate.
- Gain visibility to all contracts for maintenance, and budget for only the ones needed.
- Gain visibility to all service contracts, and maintain only the ones you need.

SAM Life Cycle - Purchasing

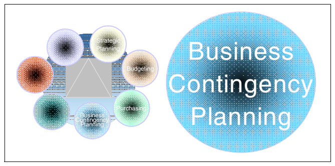


How to manage expenses is a concern that is brought up on a day to day basis. SAM allows you to optimize the global expenses of software purchases. It gives you insight into what your enterprise owns and creates a huge opportunity to reduce expenses by detecting minimally used programs, multiple versions of the same program, and different programs that have the same function. It is a great way to rationalize and eliminate extra and unneeded software.

The Purchasing phase of SAM helps to:

- Know exactly what is used and negotiate more cost-effective Enterprise License Agreements (ELA) and related agreements.
- Identify unused, under used, and multiple versions of implemented software. Using this information, software could be rationalized and related costs eliminated.

SAM Life Cycle - Contingency Planning

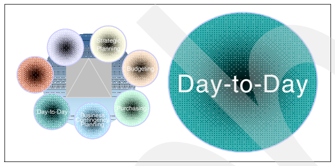


In the unfortunate event of a disaster, it is important that a enterprise has the ability to recover quickly and efficiently. From an IT Asset Management perspective, SAM will allow you to effectively prioritize recovery by providing you with information as to what hardware and software are essential to getting your business up and running again. In addition, SAM also can give you the information needed for contacting key suppliers and viewing the terms of essential contracts.

The Contingency Planning phase of SAM helps to:

- Identify the software and hardware that is critical to support the business.
- Identify and track key supplier relationships.
- Know exactly what contracts are in place, and their specific terms and conditions.
- Prioritize recovery of the different components.

SAM Life Cycle - Day-To-Day



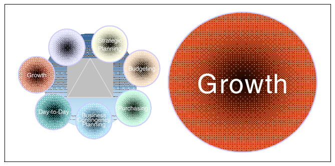
The day-to-day world of software assets is a large and complex environment. The executive management must be able to operate quickly and responsibly. With SAM, the financial organization has the ability to validate the accuracy of the invoices on a day-to-day basis as well as increase the efficiency of IT charge back. In addition, SAM provides helpful documentation with regards to software license compliance by reducing the time and cost involved with software audits.

An effective SAM program demonstrates that controls are in place by tracking hardware, software, and the contracts that maintain them.

The Day-to-Day phase of SAM helps to:

- Immediately validate invoices.
- More effectively use chargeback.
- Establish procedures, records, reports, and other support documentation to indicate compliance when faced with a software audit.
- Demonstrate adherence to stringent accounting standards (Sarbanes-Oxley).

SAM Life Cycle - Growth

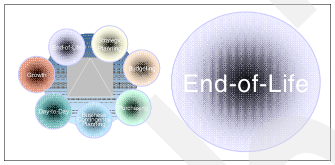


One main concern associated with the growth of an enterprise is the increasing costs of maintaining a growing business. Companies want to increase revenue and infrastructure growth while maintaining stable expenditures. SAM makes the planning of upgrades much more valuable and cost efficient. This is done by showing where assets are used most and how to redistribute them so that the enterprise does not have to buy new ones as well as giving the option to plan upgrades according to this data.

The Growth phase of SAM helps to:

- Exactly know what IT assets exists, and to understand why and when to re-deploy, acquire, or dispose of them.
- Plan software upgrades more effectively and efficiently.

SAM Life Cycle - End-Of-Life



Often, it is difficult to know when it is the right time to dispose of a particular software. SAM provides the information needed to make accurate decisions based on the business requirements. This allows you to remove unwanted software and save money.

The End-of-Life phase of SAM helps to:

- Exactly know which assets are no longer used and determine if redeployment is appropriate.
- Exactly know what software is not used, redundant, or multi-versioned, and use that information for end-of-life decisions.

1.7 Software Asset Management maturity levels

SAM is a complex management system to deploy within an enterprise. Even if you defined and implemented SAM and its life cycle in your enterprise, it is important to be able to evaluate the maturity of your SAM.

Figure 1-11 shows the different level of SAM maturity. The intention of this section is not to provide techniques on how to assess your SAM maturity, but to show what benefit could be brought by a SAM and what return of investment (ROI) could be expected in term of costs saving.

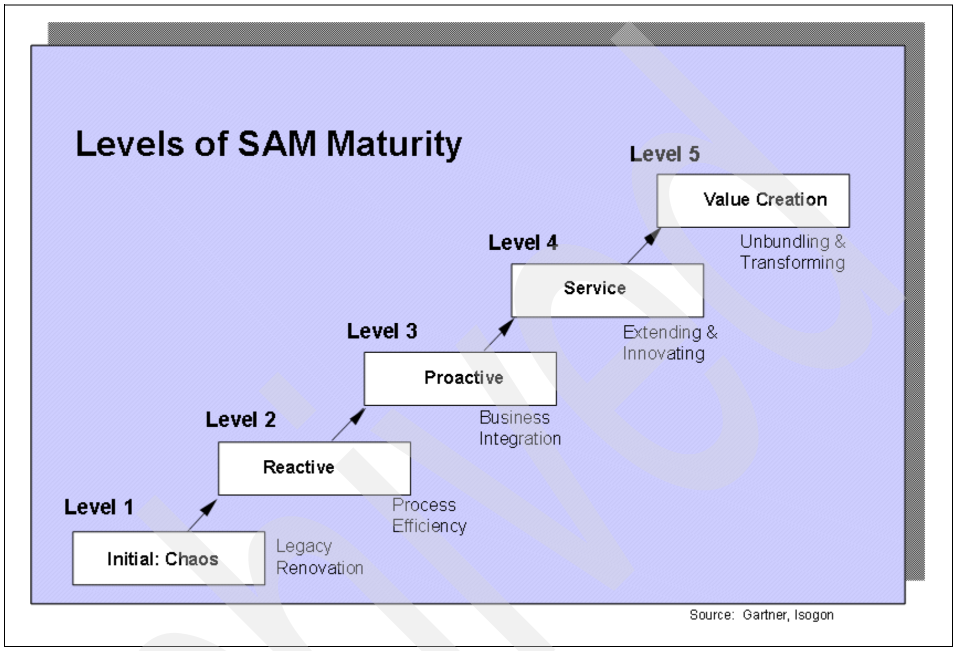


Figure 1-11 Software Asset Management Maturity Levels

In order to correctly assess the possible ROI of a SAM, it is important to assess the different level of each of the SAM Maturity attributes. Figure 1-12 on page 31 provides some of those SAM Maturity attributes.

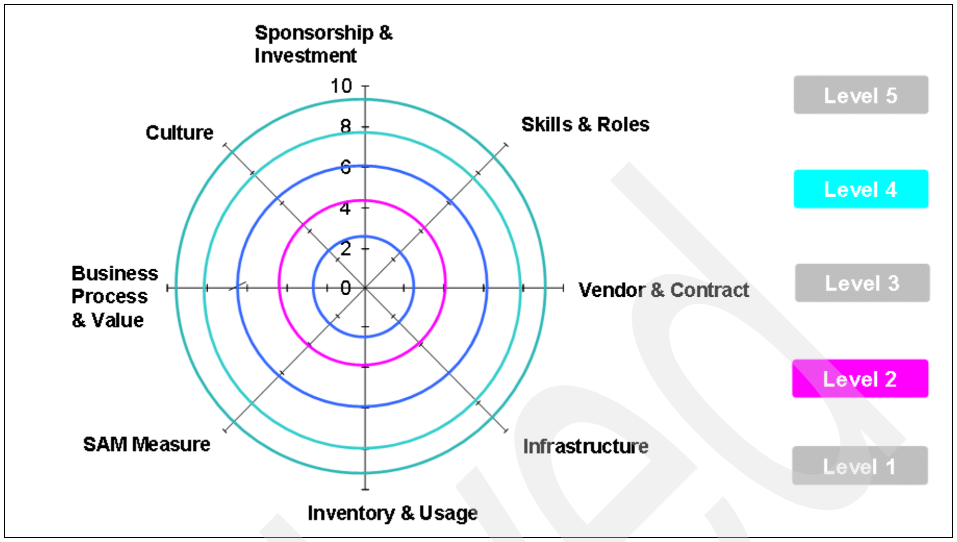


Figure 1-12 Software Asset Management Maturity attributes

1.8 IBM SAM Solution

The IBM Software Asset Management Solution is composed of different product offerings.

- IBM Tivoli License Manager

  This product specifically manages the so called distributed environments. It is able to manage either servers or workstations.

  Note: The IBM Tivoli License Manager V2.2 product has been recently rebranded as IBM Tivoli License Compliance Manager V2.2.

- IBM Tivoli License Compliance Manager for z/OS®

  This product is reserved to managed the mainframe environment.

- IBM Tivoli Contract Compliance

  This product offers the ability to manage all the contracts aspects, including the financial requirements.

The two first products integrate smoothly with contract compliance to provide the information you need to proceed through the entire Software Asset Management life cycle.

Figure 1-13 depicts the IBM End-to-End SAM solution.

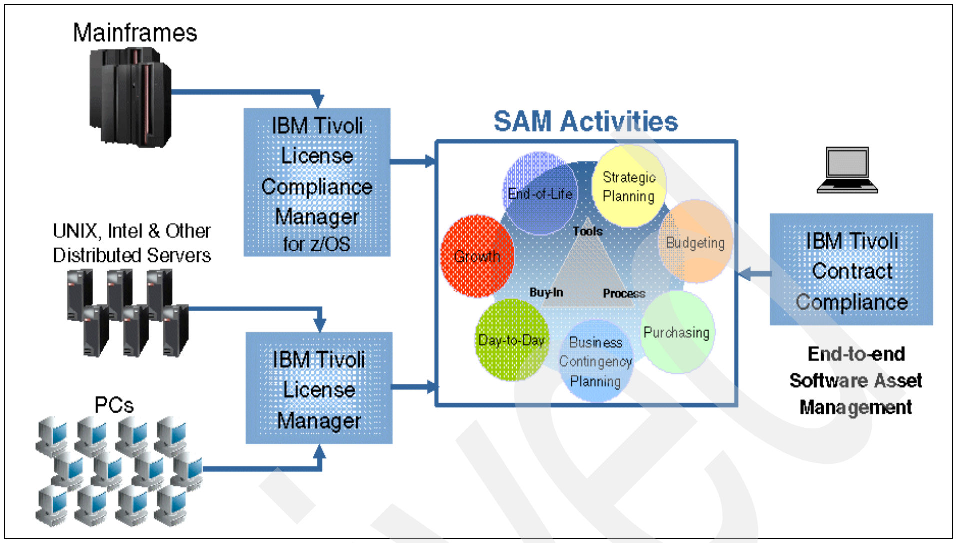

Mainframes
UNIX, Intel & Other Distributed Servers
PCs
IBM Tivoli License Compliance Manager for z/OS
IBM Tivoli License Manager
SAM Activities
End-of-Life
Strategic Planning
Tools
Budgeting
Growth
Buy-In
Process
Purchasing
Day-to-Day
Business Contingency Planning
IBM Tivoli Contract Compliance
End-to-end Software Asset Management

Figure 1-13 IBM SAM Solution overview

Figure 1-14 on page 33 shows where these three product offerings are positioned in the comprehensive portfolio of IT operational management products offered by IBM. The products are grouped in these broad technology categories:

- Business Application Management

  Products that look at the end-to-end performance of applications as they cross various technology silos.

- Server, Network, and Device Management

  Products that look at specific resources from a specific technology perspective.

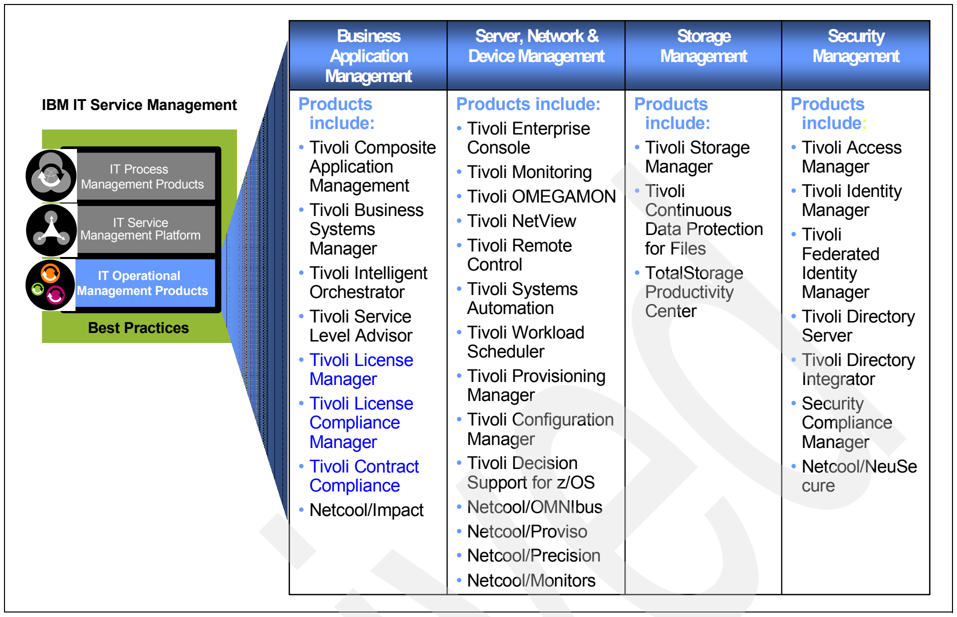
- Storage Management

  Products that help to manage backup and recovery, plus some aspects of the information life cycle.

- Security Management

  Products that manage the security aspects.

The IBM Tivoli Software Asset Management modules are part of the Business Application Management category.


IBM IT Service Management
IT Process Management Products
IT Service Management Platform
IT Operational Management Products
Best Practices

Business Application Management	Server, Network & Device Management	Storage Management	Security Management
Products include:	Products include:	Products include:	Products include:
• Tivoli Composite Application Management • Tivoli Business Systems Manager • Tivoli Intelligent Orchestrator • Tivoli Service Level Advisor • Tivoli License Manager • Tivoli License Compliance Manager • Tivoli Contract Compliance • Netcool/Impact	• Tivoli Enterprise Console • Tivoli Monitoring • Tivoli OMEGAMON • Tivoli NetView • Tivoli Remote Control • Tivoli Systems Automation • Tivoli Workload Scheduler • Tivoli Provisioning Manager • Tivoli Configuration Manager • Tivoli Decision Support for z/OS • Netcool/OMNIbus • Netcool/Proviso • Netcool/Precision • Netcool/Monitors	• Tivoli Storage Manager • Tivoli Continuous Data Protection for Files • TotalStorage Productivity Center	• Tivoli Access Manager • Tivoli Identity Manager • Tivoli Federated Identity Manager • Tivoli Directory Server • Tivoli Directory Integrator • Security Compliance Manager • Netcool/NeuSecure

Figure 1-14 IBM SAM Solution in the IBM IT Service Management Portfolio

1.8.1 IBM SAM Solution benefits

The IBM Tivoli End-to-End Software Asset Management solutions are based on three main principles that help align your software costs with your business needs.

- ► Inventory of software
- ► Identification of software usage
- ► Validation of contracts compliance

Figure 1-15 clearly shows that those three principles are sequential and interconnected.

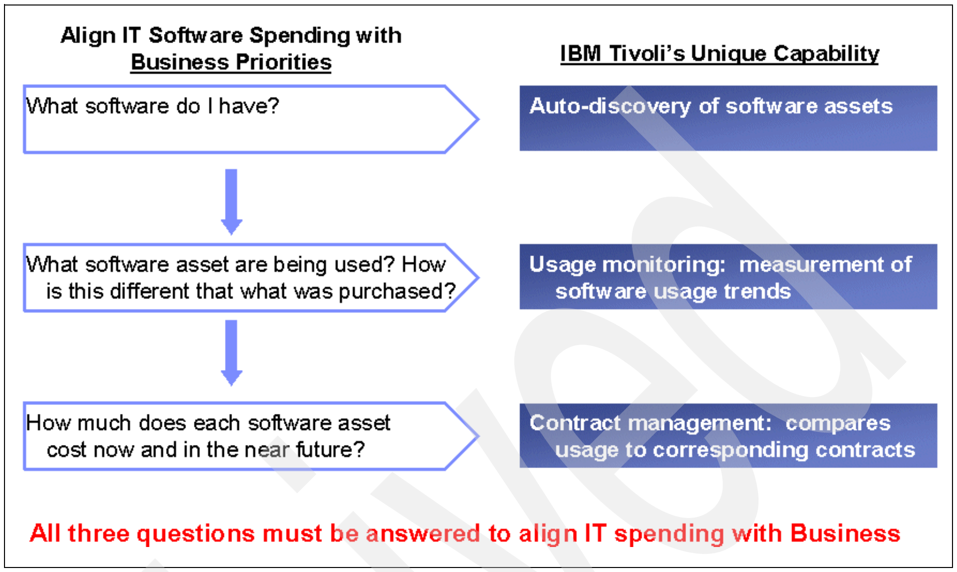

Align IT Software Spending with Business Priorities
IBM Tivoli's Unique Capability
What software do I have?
Auto-discovery of software assets
What software asset are being used? How is this different that what was purchased?
Usage monitoring: measurement of software usage trends
How much does each software asset cost now and in the near future?
Contract management: compares usage to corresponding contracts
All three questions must be answered to align IT spending with Business

Figure 1-15 IBM Software Asset Management solution approach

Based on those three principles, the IBM solution could help you achieve the ideal situation. This ideal situation would be for you to have software that you use and for which you have contracts. This would offer your the ability to validate your software invoices and be in a good position to negotiate new contracts or renegotiate current ones.

However, a Software Asset Management solution could also provide you with information regarding less ideal situations. Those situations could expose your enterprise to high risks of being audited for the installation or usage of unlicensed software. They could also cause your enterprise to pay for unneeded, or even worst, non-existing software.

In these situations, you could take remediation actions based on the information provided by your IBM SAM solution and tend to the ideal situation.

Figure 1-16 on page 35 provides you with an example of a less than ideal situation in Software Asset Management.

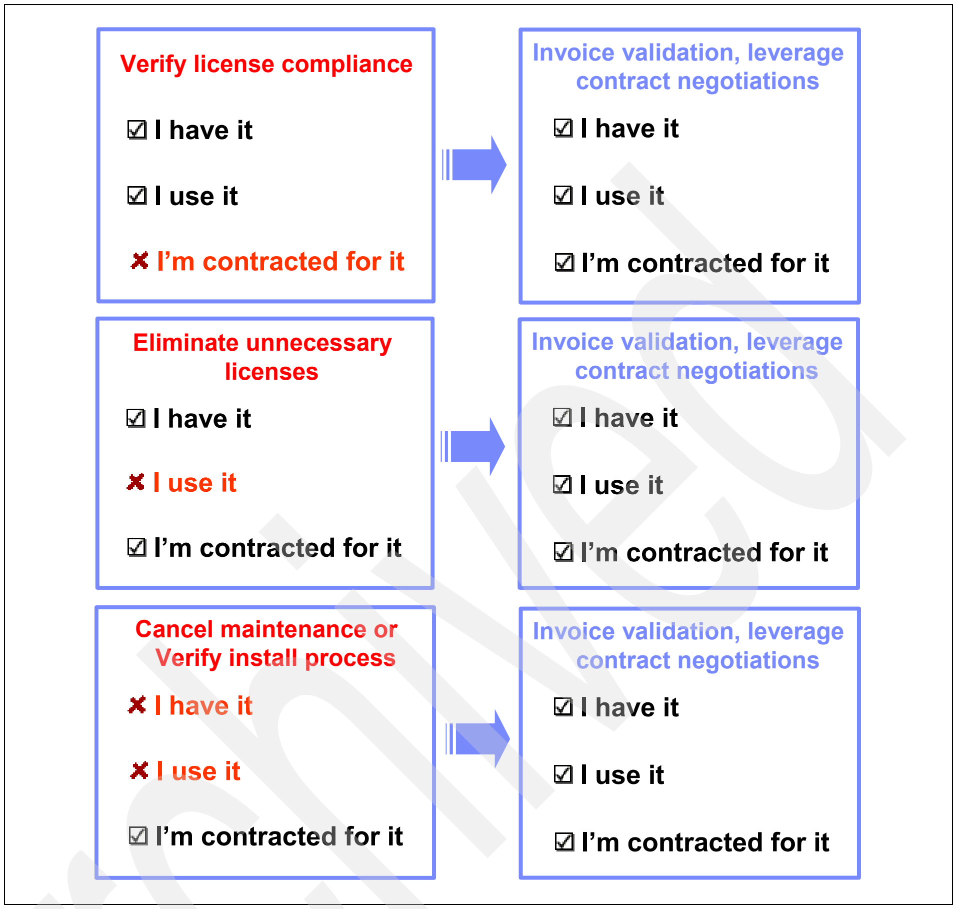


Figure 1-16 Benefits from the IBM Tivoli License Management Solution

An effective SAM solution, with the help of the IBM Tivoli products, could bring the following gains:

- Compliance Risk Reduction
  - Software vendor license compliance audits
  - Sarbanes-Oxley Act Section 404
- Software Cost Management
  - Integration of contract and licensing data
  - Reduction in IT Software budgets and Software Support costs
  - Identification and reduction of no and low use software
  - Cost effective upgrades and optimization of CPU capacity planning

– Efficient server and software consolidations
– Strong contract negotiation leverage
– Invoice validation and effective chargeback
– Disaster recovery and business continuation

1.8.2 IBM Tivoli License Manager overview

Often, software products do not provide technical support for License Management activities, leaving product License Management completely in the hands of the organizations buying those products.

IBM Tivoli License Manager provides software inventory, use metering, and license allocation services on Windows®, UNIX®, and OS/400® platforms. It can be scaled to meet the needs of large and small organizations, and supports the management of multiple organizations.

Note: This redbook refers to the IBM Tivoli License Manager V2.2 product, which has been recently rebranded as IBM Tivoli License Compliance Manager V2.2.

The IBM Tivoli License Manager product name is still in use in this redbook, as well as on all of the administrative user interfaces and in the documentation of the product.

Refer to the product's Release Notes, as they provide the most current information about the IBM Tivoli License Compliance Manager V2.2 release of the product. The latest version of the Release Notes can be found on the Tivoli Information Center Web site:

http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?toc=/com.ibm.itlm.doc/toc.xml

Information about installed software and software use is collected from monitored computers by an agent that can be deployed on a range of Windows and UNIX platforms and is reconciled with license and contract information that you define. Information is stored in a central IBM DB2® database and can be accessed using a Web user interface.

IBM Tivoli License Manager is based on three main pillars:

- Installed software detection and usage metering

  The detection and collection of installed software is performed using the Common Inventory Technology (CIT) module. This software is also used by other IBM Tivoli products for software recognition. The use of CIT enables scan results to be shared between Tivoli products, reducing the impact of multiple Tivoli applications running on the same system.

  The deployed IBM Tivoli License Manager agent is able to detect the starting and stopping of other software installed on the system. This means you will be able to know the level, number of users or processors, and duration of use of a product. It is important to notice that the metering unit used depends on the type of license you define for the products you want to monitor.

  IBM Tivoli License Manager installation detection and usage monitoring base their capability to recognize a software on a catalog of software signatures. IBM provides such a catalog, which includes product information for a wide range of IBM and non-IBM native and Java™ applications. While IBM provides regular updates to this catalog to ensure that the most well-known software in the software marketplace is available, you could extend this catalog to include you own information for detecting, for example, in-house software.

- Licenses and contract information

  In order to give you the opportunity to reconcile software usage and installation with the license agreements of your software contracts, IBM Tivoli License Manager offers the ability to define licenses and contract information. This includes information about the procurement of the license, and includes the license type and the license quantity. The license type identifies a set of rules that determine how the use of the license is counted, for example, by number of instances of the product installed, by number of concurrent user sessions, or by the number of processors in use.

  IBM Tivoli License Manager also offers the ability to manage products that share components with other applications, that is, part of a bundle, or when its licensing model depends on how and where it is used, for example, sometimes the same product has different license agreement if used as a stand-alone product or as a prerequisite software for another application.

  Finally, IBM Tivoli License Manager offers you the ability to assign a license model to a different group of targets depending on whether, for example, the licenses are assigned to a single user or a group of machines.

- Reporting

  IBM Tivoli License Manager includes a set of preconfigured reports for the analysis of collected installed software and software use information and its reconciliation with the licenses that have been defined for products.

This redbook provides best practices on how to deploy a License Management solution in a large scale environment based on the IBM Tivoli License Manager product.

Refer to Chapter 3, “IBM Tivoli License Manager general overview” on page 59 for detailed information about the features, capabilities, and architecture of IBM Tivoli License Manager V2.2.

2

Project and engagement considerations

The project engagement phase is a crucial phase in any project. However, in a License Management project, it is a vital element. As mentioned in the previous chapter, the main aspect of License Management is to reconcile what has been acquired, in term of agreements and cost, with what it is really deployed within the entire enterprise. This means reconcile the procurement information with IT information, which mainly do not use the same language.

Chapter 1, "Business perspective for License Management" on page 3 provided key motivations factors for having a Software Asset Management solution and a License Management solution. This chapter now focuses on the engagement phase of a License Management project and what are the organizational factors that must be taken into account for making your project a success.

The following recommendations, phase descriptions, and terms provided in this section are based on the IBM Global Services Method. Nevertheless, these recommendations could easily adapt to your own project method, as we mainly provide conceptual information instead of a too detailed level of information.

This chapter describes the following topics:

- 2.1, "Overall project phases" on page 41
- 2.2, "License Management project engagement phase" on page 42

- 2.3, “License Management project delivery phase” on page 44
- 2.4, “License Management project delivery key elements” on page 47
- 2.5, “Steady state roles and responsibilities” on page 49
- 2.6, “From the business perspectives to implementation” on page 54

2.1 Overall project phases

In any large, small, internal, or external projects, it is important to clearly define the project, including the objectives and all the requirements, and to deliver it according to a methodological approach, either through a standard method or one your own.

Generally, a project has two main phases: the first one is the proposal step (or project engagement) and the second one is the delivery of the project. They could be described as follows:

- Project engagement

  As mentioned, even if your License Management is an internal project or if you will provide a solution to a client, it is important to develop a clear project proposal. This proposal should contain the following elements:

  – The scope of work of the project, which defines the objectives of the project and the responsibilities of each party. This includes what deliverable and work products will be delivered in the project

  – The definition of the project, meaning the proposal itself. This is important to ensure that every party engaged in this project has the same understanding of the objectives and the context of the project.

  – The method that will be used.

  – The assets or intellectual capital that will be used or could be reused.

  – The acceptance criterion and how they will be measured.

  – The effort estimation and pricing information, meaning a first version of the project plan.

  – Analysis of the main risks of the project.

- Project delivery

  The delivery of the project is the biggest part of the project and consists of:

  – Reworking the project plan, defining the staff with names and a schedule, refining the project workbook, and assessing all the risks.

  – Designing the solution using an adequate methodology.

  – Ensuring that a quality assurance session is done at regular intervals to be sure that you will be on time and deliver what was planned.

  – Finally, the project will be handed over to the operational organization after it is accepted by the sponsors of the project.

  – One important step that is often missed is the closing of the solution by developing lessons learned and by the creation of new assets, or improvement of the method used.

These two main phases are described hereafter specifically for an License Management project based on IBM Tivoli License Manager.

2.2 License Management project engagement phase

The goal of this section is not to provide you with a complete engagement model for an License Management project based on IBM Tivoli License Manager V2.2. It is more, based on the best practices, to inform you what are the most important aspects that need to be taken into account and which risks you should plan to mitigate during the engagement phase in order to not face problems during delivery.

This phase is also known as the Solution Startup phase.

During the engagement phase of an License Management project based on IBM Tivoli License Manager, you should analyze these points:

- Sponsors and leadership

  This is the most critical success factor of your project. While an IBM Tivoli License Manager project will bridge the IT organization and the procurement organization, board level sponsorship and commitment are essential to ensuring successful implementation of the License Management solution.

  Senior management must clearly define and communicate the strategical aspects and the business importance of such a project. In addition, as business impacts may arise from not being compliant, senior management will also have to take quick decision when issues will be brought to them.

  Nonetheless, such project could easily be a success in term of technology delivery, but may fail due to the non-integration of other critical processes within the enterprises. Senior management will have to assure that such a project could really be integrated with other processes.

  In addition, as senior management must address legal implications that pertain to the use of software, hardware, and service and copyright laws, it has to communicate clearly the global enterprise objectives of this project that is for to be a simple IT project.

- Centralization of acquisition and procurement

  Some of the most significant benefits of a License Management solution comes from the centralization of acquisition and procurement. In the specific context of an IBM Tivoli License Manager project, centralization will help your enterprise to bridge and reconcile one single entry point from the IT side with another single entry point of the procurement organization. In addition, having a competency team able to analyze all licenses agreement will drastically simplify the integration and usage of the solution.

- Policies and procedures

  Policies and procedures that are practical and mandatory for everyone touching IT assets (procurement to retirement) must be developed, implemented, and monitored for adherence.

  The policies covering licenses, agreements, copyright, and acceptable use of software must be formulated and disseminated to all personnel by the sponsors of the project.

- Roles and responsibilities

  Roles and responsibilities of these roles must be clearly defined and documented. These roles must support and take ownership that results from these policies. This must include internal and external personnel and each consequence of violations must be well understood by every player, including end users.

- Skills and competence

  Based on the best practices, it has been recognized that it is sometimes difficult for procurement people to translate the terms and conditions and license agreements into technical elements for the solution to do reconciliation between collected information and procurement. The same happens the other way around, meaning that IT people do not always understand the contractual aspects of the licenses.

- Standardization and centralization of software deployment

  Even if a License Management project is not directly impacted by the way the software is installed, optimization of licenses is mainly dependant on this process. By better categorizing the software based on the real business need, monitoring of licenses will be simplified and the distribution of licenses quota will be easy to do. For example, software that could be licensed at the corporate level should be part each platform, software that is required by a specific business unit should only be installed and monitored on the business unit's platforms, and finally, software for personal requirements should be provided on demand.

  In addition, standardizing the installation process ensures that the process includes the corporate licenses policies. Reduction of the number of software for the same task is controlled by this central process.

- Contracting

  If the procurement of the licenses of your enterprise is outsourced to a third-party company, it is then also important to integrate them in the project and to ensure that you will receive the required information and that they will be able to use the collected data to optimize how they manage your licenses costs.

- Cost and benefit analysis.

  It is important to be able to demonstrate the positive impact, both quantified and non-quantified benefits, that your License Management project will have on the total cost of ownership. For that purpose, it is important to collect and use relevant information from the start of the project.

Of course, the engagement phase includes other steps and activities, but as mentioned, the scope of this section was to highlight the critical points that you must take into account in your engagement. Nevertheless, the following sections of this redbook help you to size your engagement and build a first version of your project plan.

The main active roles in this phase are:

- Project Manager

  Responsible for engaging the project, evaluating the cost and the risks, managing the relationship with the project's sponsors, finding adequate resources and get approvals throughout the entire project execution.

- IT Architect

  Responsible for defining what tasks are required to implement the project, evaluate the technical risks, and select the right methodology.

2.3 License Management project delivery phase

The previous sections provided some critical information, which need to be collected, defined, and managed for engaging a license manager under the best conditions. The goal of this section is similar, as it provides some recommendations on how to approach the delivery of the License Management project based on IBM Tivoli License Manager V2.2 under the best conditions.

Once again, the scope of this chapter is not to provide a clear project plan of the delivery phase, but to clearly highlight what the approach, based on the best practices, should be.

Of course, there are different approaches that could be defined to implement a License Management solution. However, we will present one that could easily be leveraged on a large scale. This does not mean it could not be used for small scale implementation, but it will need to be tailored to fit the requirements and complexity of the project.

Figure 2-1 on page 45 presents the main phases required to deliver a License Management project.

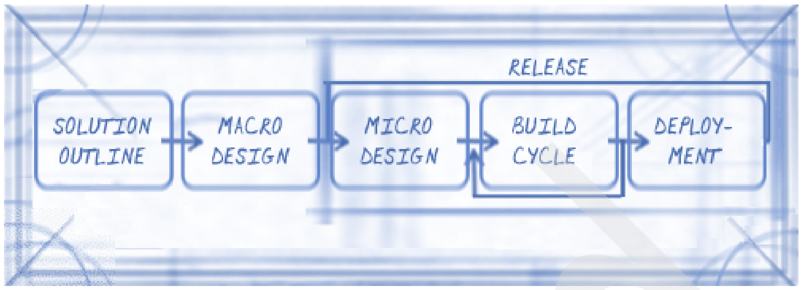


Figure 2-1 ITILM Project Delivery Phase Decomposition Overview

The list hereafter gives an overview of each phase of the delivery. It is important to first clarify the notion of release. Even if IBM Tivoli License Manager is a packaged software, in a large scale project, it may not be possible for all of its capabilities or technical specifications to be implemented in one shot. Delivery of a first release containing, for example, basic product functionality maybe required. Then a second release, which includes, for example, scalability and tuning, may be delivered.

1. Solution Outline

   The purpose of this phase is to finalize all aspects of the project plan and refine the cost. In addition, all the detailed requirements of the project must be collected in this phase. During this phase, the Chief IT Architect and the Project Manager gain a basic understanding of the business problem and the Chief IT Architect sketches out an approach for solving it.

2. Macro Design

   In this phase, the architecture team identifies and structures the components of the solution, including processes, applications, and infrastructure. In addition, the team describes how the pieces of the solution relate to one another. During this phase, the proposed solution is normally built and developed to execute design verifications and acceptance.

3. Release Micro Design

   The architecture team describes, in detail, the operation and cooperation of the elements of an operational subset of the solution, called a release or a solution increment, and guides the project team in planning the development of the release. The team also maintains a view of the entire solution as a series of releases, each operational and building on previous releases.

4. Release Build Cycle

   The project team, perhaps advised and guided by the architecture team, develops and tests the release in a controlled environment and plans for the deployment of the release.

5. Release Deployment

   The team inserts the release into the production environment and ensures that it cooperates with the existing IT environment. Although the architecture team performs an advisory role, it can be a critical one in the sense that it brings to the team a sense of what the production environment can handle, and how it must be adapted to accommodate the release being deployed.

Finally, after the deployment of each release, the solution is handed over to the operational team, which means adequate working instructions are defined by the project team for the support personnel to correctly operate the solution as it has been designed.

The main roles active in this phase are:

- Project Manager

  Responsible for starting the project, monitoring its advancement, managing risks and issues when they arise, managing the resources and the costs, and ensuring that the objectives will be met in the defined budget. Of course, the Project Manager is in charge of the entire communication. In case of a possible shift either in the delay or cost, he is in charge negotiating with the sponsors.

- IT Architect

  Responsible for designing the entire solution and ensuring that the solution that will be provided answers the requirements and is viable. He works closely with the Project Manager.

- IT Specialist

  Responsible for assisting the IT Architect in the technological aspects. The IT Specialist will also be in charge of implementing the solution designed and managing all emerging technical problems.

2.4 License Management project delivery key elements

Besides the elements that need to be managed during the engagement of the project and the activities that need to be done during the design and the implementation of the solutions, there are key elements that should be managed. Even if they are not directly linked with the technical aspects of the project, based on the best practices, not managing them will result in real difficulties in handing over the solution, for the operations to maintain it, and for your client, either internal or external, to use it.

The are key elements are:

- Management of the asset change. There is a well known process called Install, Move, Add, Change (IMAC) that covers that type of asset change. As IBM Tivoli License Manager is measuring software usage on assets, it is important that this is maintain accurately. For example, when a machine is disposed, this change must be reflected in IBM Tivoli License Manager to avoid this machine still being part of the compliance report. The more precise the synchronization between the assets lists and the IBM Tivoli License Manager information is, the more accurate the compliance will be and the easier it will be for the license manager, for example, to reassign disposed licenses to another requestor.
- During the test phases of the project and the pilot, a regular analysis of the reports must be performed to validate and prove that all processes are implemented as required and data is correct to provide accurate information.
- The License Management must be done from the acquisition to the disposal of the software. Not covering all the aspects will not ensure accurate compliance.
- The following enterprise processes may be impacted by a License Management project. It is important to control and communicate with another process owner if the License Management aspects is really integrated and a “bridge” is available.
  - Budget and forecast
  - Acquisition: Controls, clear procedures, and reconciliation
  - Software deployment: Control on exception, business need, centralization, record installations at installation time, and for on demand software.
- One of the key successes of an License Management project based on IBM Tivoli License Manager, which ensures that the solution is viable during its entire life cycle, is the management of the signatures. You must be sure that the signatures will be maintained adequately and that global enterprise policies are defined around the definition of software signatures.

- Accuracy of the information is a key element for management. As the objective of IBM Tivoli License Manager is to reconcile the business and technical information, the following points are important:
  - If procurement information (contract and procured licenses) is not defined as described in the terms and conditions of the contracts, IBM Tivoli License Manager reports will not provide the expected information.
  - Each software entitlement must be cross-checked and reports must be reviewed until a final agreement on the quality of information is reached.
  - The way to monitor the licenses must be extremely precise. If the wrong license type or the wrong unit type is used, the reports will not provide accurate information about the relationship between terms and conditions and monitored metrics.
  - IBM Tivoli License Manager relies on signatures to discover the use or the installation of a software. If these signatures are not defined or not correct, IBM Tivoli License Manager will not be able to get any metrics and do any reconciliation.

As mentioned, these elements are not all directly linked to the IBM Tivoli License Manager solution, so the Project Manager should keep the sponsors of the project alerted in order to get efficient support to solve some issues, mainly political, proactively.

This recommendation is to help you and your enterprise get the highest return on investment as possible out of your License Management solution based on IBM Tivoli License Manager. It really benefits every key player in the enterprise managing or using licenses to have them manage a state of the art and automated solution.

The following statement has been extracted from the Software Asset Management (SAM) ITIL book:

"It is impossible to implement and effective SAM process without the successful design, development, implementation and maintenance of accurate SAM Databases, automatically updated from the live infrastructure."[1]

So, it is important that you engage your project in the best conditions, manage your deployment as best as possible, but also to operate it with accuracy. In this context, we recommend that a clear Quality Assurance process must be defined around the IBM Tivoli License Manager solution to be sure that Business decisions are made with 100% reliable information.

[1] Source: Reprinted, by permission of the publisher, from *Software Asset Management*, The Staionery Office, 2003

Important: Business decisions will be made based on IBM Tivoli License Manager reports. If this information is not accurate, significant impact to the business can occur.

2.5 Steady state roles and responsibilities

In order to operate the solution in the best condition and to really perform License Management in your enterprise, roles and responsibilities should be defined within your organization.

2.5.1 Operational roles and responsibilities

At the end of the macro design phase, the IT Architect and the Project Manager should start defining the best organization that will operate the final solution. As an IBM Tivoli License Manager solution, from a technical perspective, is often crossing more than one workbench and, from a business perspective, is serving business people, a support organization should be defined as a matrix-like organization, oriented in a Service Management vision.

Here we discuss the roles and responsibilities, not the number of head count. The following roles should be defined in a steady state environment:

- Service Manager

  Responsible for interfacing with the clients of your License Management, gathering new requirements, and ensuring that the service is providing the expected quality. In addition, as your technical solution may cross more than one workbench, this role will be the main interface with the managers and resources of each workbench. This will assure the viability and good evolution of the solution deployed.

- IT Architect

  This role is also known as the steady state architect. This role is responsible for analyzing the impact of each new requirement collected by the Service Manager. This role should also estimate the work required to design and implement the changes. In addition, it should also follow the improvements that are made at the technical level, for example, a new version of the IBM Tivoli License Manager Software, and explain to the Service Manager what values these improvements could bring to the client. In this situation, the Service Manager may create new opportunities.

- IT Specialist

  Even if such a solution is based on different components that could be managed by a different team, the License Management solution based on IBM Tivoli License Manager needs to be operated by a technical specialist that cross all the workbenches.

2.5.2 License Management roles and responsibilities

In Chapter 1, “Business perspective for License Management” on page 3, we presented the positioning of the License Management, which is part of the Software Asset Management, in the context of ITIL. There is no specific role defined for SAM. However, if you implement a License Management solution, the following roles must be defined in your enterprise. Some of them may already exist in your organization, in this case, new responsibilities related to License Management must be assigned.

Even if the roles are mainly related to the IBM Tivoli License Manager naming convention, we provide the information to which IBM Tivoli License Manager role they correspond. This should help you define or assign responsibilities to standard ITIL roles. Also, refer to 6.3, “IBM Tivoli License Manager Administrators” on page 196 for details on roles defined for IBM Tivoli License Manager.

Procurement Manager

The Procurement Manager assume at least the following *enterprise wide* responsibilities:

- Analyzes the license ordering request.
- Orders licenses from the vendors.
- Records the licenses contract information in the Contracts Database.
- Makes the license contracts conform to the compliance information provided by the License Manager.
- Manages an ordering list of waiting licenses.
- Charges back to the clients for the licenses ordered specifically for them.

The Procurement Manager must assume at least the following *IBM Tivoli License Manager specific* responsibilities:

- Has financial responsibility for all contracts and negotiations.
- Owns the information necessary for decision making about procurement of software within the IBM Tivoli License Manager Organization.
- Defines the contracts that must be brought into IBM Tivoli License Manager.

- Defines the procured licenses for each license contract, including how the procured licenses must be monitored in IBM Tivoli License Manager.
- If other systems are in place for Contract Management, the Procurement Manager must extract the procurement information and import this information in IBM Tivoli License Manager.
- Analyzes compliance reports and data analysis of the license manager.
- Updates the license contracts and procured licenses after after streamlining the contracts with vendors.

This role is under the responsibility of Procurement Management, as mentioned in 4.3, “Respective roles of Procurement Management and ICT Management should be part of the Procurement organization”, in *Software Asset Management* by Office of Government Commerce Staff

License Manager

The License Manager assumes at least the following *enterprise wide* responsibilities:

- Provides compliance information for the licenses by doing the reconciliation between the procurement information and the inventory of the licenses.
- Informs the Procurement Manager about violations or underusage of licenses.
- Verifies the accuracy of the inventory information about licenses.
- Provides licenses inventory information to Management and Operations staffs.
- Supports and manages any licenses inventory request.
- Optimizes the management of the licenses.
- Provides internal and external reports about licenses inventory information.

The License manager must assume at least the following *IBM Tivoli License Manager specific* responsibilities:

- Configures the entitlements to match software contracts.
- Distributes licenses to users and machines.
- Generates reports to assist the Procurement Manager with compliance analysis, software acquisition, and planning.

This role is under the responsibility of the Configuration Manager, as stated in Section 7.3.10, “License Management and should be part of the Configuration Management organization”, of *Software Asset Management*, by the Office of Government Commerce Staff

Catalog Manager

The Catalog Manager assumes at least the following *enterprise wide* responsibilities:

- Adds and maintains configuration information about software signatures for all platforms.
- Adds and maintains the relationship between the logical product hierarchy and the software signatures.
- Adds and maintains the relationship for bundled software applications.
- Ensures that the defined software signatures are the ones used to detect the use or installation of a software.
- Detects and reports non-authorized software used in the IT environment via the Unknown Software discovery tool.

The Catalog Manager must assume at least the following *IBM Tivoli License Manager specific* responsibilities:

- Adds and maintains the product, version and release hierarchy for each product (including pricing group evaluation).
- Adds and maintains module information (including the detection scope of work).
- Links a module to the product hierarchy.
- Manages the unknown file information.
- Imports regularly the IBM Catalog and provides new signatures to IBM that should be part of the standard IBM Catalog.

The Catalog Manager role is not a defined role within ITIL. This role is a sub-role of the Configuration Manager role. The Catalog Manager is in charge of maintaining the Definitive Software Library (DSL) and the Configuration Management Database (CMDB), which stores all information related to Configuration Items (CI).

The Catalog Manager is in charge of storing all the configurations of the component items (CIs). As software is also a CI, the Catalog Manager manages and stores all information related to software, including the best way to detect software.

In the context of License Management, IBM Tivoli License Manager requires software signatures to detect the use and the installation of software. This information is related to software, which is under the responsibility of the Configuration Manager for management purposes. However, this information must be given or acknowledged by the Release Manager, as the signature is part of the release and is so defined as a record of a release.

Release Manager

The Release Manager manages the release of the software deployed in your enterprise and mainly assumes the following responsibilities:

- Defines release policy and planning.
- Designs, builds, and configures the release.
- Defines the acceptance of releases.
- Defines the planning of a rollout.
- Signs off on the release of an implementation.
- Audits hardware and software prior to and following the implementation of changes.
- Installs new or upgraded hardware.
- Stores controlled software in both centralized and distributed systems.
- Releases, distributes, and installs software.

The Release Manager knows exactly which software is deployed or will be deployed within your enterprise. In addition, the Release Manager is always the subject matter expert on how the license terms and conditions must apply for software. In regards to signature management, as the Release Manager knows the application perfectly, the Release Manager could easily provide this information to the Catalog Manager or the Configuration Manager in ITIL terms.

This role is under the responsibility of the Release Management, as mentioned in Chapter 9, “Release Management and should be part of the Release Management organization”, of *Software Asset Management*, by the Office of Government Commerce Staff.

There is no specific IBM Tivoli License Manager Roles for the Release Manager.

Configuration Manager

The Configuration Manager manages the configuration of the components items and mainly assumes the following responsibilities:

- Plans and defines the purpose, scope, objectives, policies and procedures, and the organizational and technical context for Configuration Management.
- Selects and identifies the configuration structures for all the infrastructure’s Configuration Items (CI), including their ‘owners’, their interrelationships, and configuration documentation. The Configuration Manager allocates identifiers and version numbers for CIs, labelling each item, and entering them in the Configuration Management database (CMDB).
- Ensures that only authorized and identifiable CIs are accepted and recorded, from receipt to disposal.

- Reports all current and historical data concerned with each CI throughout its life cycle.
- Verifies and audits the physical existence of CIs and checks that they are correctly recorded in the configuration management system.

This role is under the responsibility of the Configuration Management, as mentioned Chapter 7, “Configuration Management and should be part of the Configuration Management organization”, of *Software Asset Management*, by the Office of Government Commerce Staff.

There is no specific IBM Tivoli License Manager Roles for the Release Manager.

2.6 From the business perspectives to implementation

In Part 1, “Business aspects and project engagement considerations” on page 1, we discuss the reasons why you should deploy a License Management solution. We also position the License Management in the overall picture of the IT Service Management and describe the IBM solution that could help you in this area. We also provided you some best practices on how to approach and engage such a project. The high level tasks of such a project have also been introduced.

You also discovered not only what the key elements you need to manage and take into consideration for making your project a success are, but also how you should organize your operations to get the highest benefit from your solution.

The next chapters of this redbook will provide deeper information, based on the IBM best practices we were able to define during real, large scale implementations, on how to implement an IBM Tivoli License Manager solution and will mainly focus on the following aspects:

- First of all, you will be introduced to the IBM Tivoli License Manager technical architecture and capabilities. It is important to first understand the high level architecture and discover the different components, and their names. This will help you better understand some of our recommendations and some of the architectural decisions presented. This information is presented in Chapter 3, “IBM Tivoli License Manager general overview” on page 59.
- Then, every IT solution deployed within an enterprise must be aligned with requirements. We will guide you on how to define those requirements and provide you with the key requirements you should at least define in your project. Also, a License Management solution is dependent on a number of business and technical elements that makes planning an essential task in order to ensure a successful deployment of the solution. This information is presented in Chapter 4, “Solution design fundamental tasks” on page 75.

- Once the requirements are collected and the solution design fundamental tasks are completed, the IT Architect is ready to prepare the entire IBM Tivoli License Manager solution. This involves creating the solution in terms of physical and logical designs aspects and are deeply related to IBM Tivoli License Manager product architecture, features, and capabilities. Both Chapter 5, “IBM Tivoli License Manager physical design” on page 89 and Chapter 6, “IBM Tivoli License Manager logical design” on page 185 provides a great deal of information about what physical and logical elements of an IBM Tivoli License Manager solution are built.

Part 2

Solution design considerations

In this part, we provide detailed information for the architecture and design of a IBM Tivoli License Manager V2.2 solution implementation. Both IT Architects and IT Specialists will benefit from the information provided in the following chapters, as they cover detailed aspects of the solution from an introduction to the product, its components and architecture, solution design fundamental tasks, to detailed physical and logical design and configuration of all components of an IBM Tivoli License Manager V2.2 solution. The following topics are discussed:

3

IBM Tivoli License Manager general overview

IBM Tivoli License Manager is the IBM solution to reconcile software license usage in an organization with the licenses owned by that organization. It offers a way to control and apply software vendors' licensing policies, enabling easy management of different products in different ways and reflecting the rules of each product's license agreement.

This chapter gives an overview of the IBM Tivoli License Manager V2.2 product, describes its functionality, components, architecture, and gives some considerations for implementation. These topics are described in the following sections:

- 3.1, "Product overview" on page 61
- 3.2, "Architecture" on page 62
- 3.3, "Physical components" on page 64
- 3.4, "Logical components" on page 66
- 3.6, "ITLM Catalog Manager" on page 72

Note: This redbook refers to the IBM Tivoli License Manager V2.2 product, which has been recently rebranded as IBM Tivoli License Compliance Manager V2.2.

The IBM Tivoli License Manager product name is still in use in this redbook, as well as on all of the administrative user interfaces and in the documentation of the product.

Refer to the product's Release Notes, as they provide the most current information about the IBM Tivoli License Compliance Manager V2.2 release of the product. The latest version of the Release Notes can be found on the Tivoli Information Center Web site:

`http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?toc =/com.ibm.itlm.doc/toc.xml`

3.1 Product overview

License Management is one of the many processes involved in the delivery of IT services. By having real-time control of software assets, an enterprise can understand exactly what resources are needed to support its business.

As described in Chapter 1, “Business perspective for License Management” on page 3, the IT Infrastructure Library (ITIL) defines Software Asset Management (SAM) as all of the infrastructure and processes necessary for the effective management, control, and protection of the software assets within an organization, throughout all stages of their life cycle.

IBM Tivoli License Manager is a Web-based solution that meets the challenge to support the complex software asset management process. It provides software usage metering, procurement management, and license allocation and compliance support services on different host platforms.

IBM Tivoli License Manager can help enterprises meet the software assets management objectives by accomplishing, often silently, a certain number of tasks, described as follows:

- Collecting information about installed products using the Tivoli Common Inventory Technology for scans.
- Identifying the start and the stop of software products on any machines on which the ITLM Agent is running.
- Providing reports that allow the comparison of installed, used, and procured licenses in support of procurement management activities
- Metering software usage for products that have no license definitions or requirements.
- Generating alerts automatically informing administrators when license usage reaches a predefined level.
- Assigning a pool of licenses to users and machines with product-specific, context-driven quantity that defines the number of software instances, either installed or in execution, for a specific product for compliance checks.
- Associating the information contained in contracts to licenses’ definitions.
- Providing security of confidential information that is ensured by the maintenance of security profiles for access to the Administrative user interface, and by the encryption of data during transmission between the various components of the ITLM solution, as well as when accessing the Administrative user interface.
- Electronic entitlement for IBM Software using pre-filled, error free IBM license terms and definitions.

- Capability to manage a complex product made up of several components or bundles, possibly installed on different systems.
- Logical partitioning and virtualization support using layer abstraction based on partitioning and virtualization elements in the virtualization stack (node $\rightarrow$ shared pool $\rightarrow$ LPAR $\rightarrow$ virtual machine).

3.2 Architecture

IBM Tivoli License Manager V2.2 relies on IBM Middleware Software for a platform and uses the following software components:

- IBM DB2 Universal Database
- IBM WebSphere Application Server
- HTTP Server (IBM HTTP Server is bundled with IBM WebSphere Application Server.)
- HTTP Server plug-in (A plug-in for IBM HTTP Server is bundled with IBM WebSphere Application Server.)

Refer to *IBM Tivoli License Manager Planning, Installation, and Configuration*, SC32-1431 for a detailed list of software requirements for an IBM Tivoli License Manager V2.2 implementation.

IBM Tivoli License Manager V2.2 leverages a three-tiered architecture as follows:

- Tier 1

  A client that runs on the end-user server or desktop, named ITLM Agent.
- Tier 2

  A resource manager providing real-time services for the client named ITLM Runtime server
- Tier 3

  An application server that contains the business logic and stores the core data for the application named ITLM Administration server.

Figure 3-1 on page 63 depicts the IBM Tivoli License Manager three-tiered architecture.

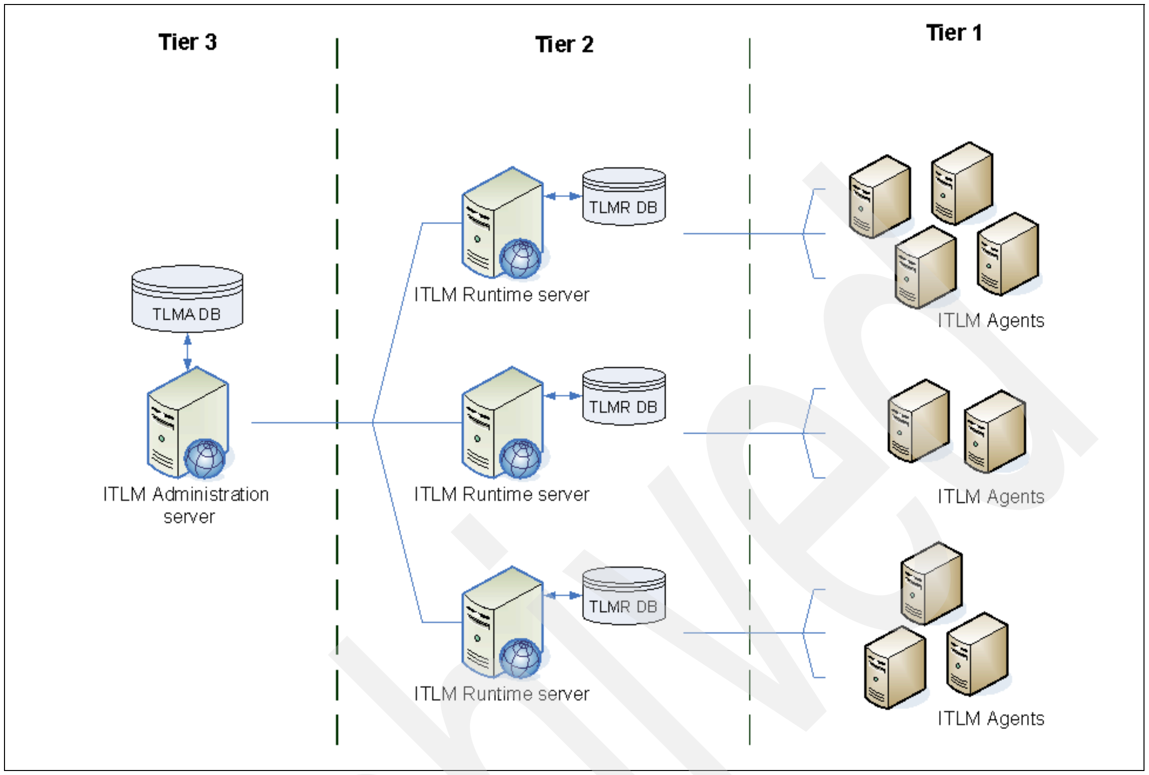


Figure 3-1 IBM Tivoli License Manager three-tier architecture

All communications between IBM Tivoli License Manager components use the HTTP protocol. HTTP requests made by an ITLM Agent to the ITLM Runtime server or by the ITLM Runtime server to the ITLM Administration server are first received by an HTTP server that must be installed on each ITLM Administration server and ITLM Runtime server. The HTTP server must forward the HTTP request to the corresponding ITLM application running on IBM WebSphere Application Server. To allow such a transfer, the IBM WebSphere Application Server application installs a HTTP Server plug-in for WebSphere® plug-in on the HTTP server. This plug-in is able to transfer the HTTP request to the ITLM application running on the IBM WebSphere Application Server.

Both the ITLM Administration server and ITLM Runtime server need to store and access data in an RDBMS. To access this RDBMS, IBM WebSphere Application Server uses the Java DataBase Connectivity (JDBC™) technology.

IBM Tivoli License Manager three-tiered architecture is comprised of physical and logical components. The following sections go into details about these components and how they interact to each other.

3.3 Physical components

This section provides an introduction to IBM Tivoli License Manager physical components. These components, as well as architectural decisions in which they are involved, are covered in great detail in Chapter 5, “IBM Tivoli License Manager physical design” on page 89.

The IBM Tivoli License Manager V2.2 three-tiered architecture is made up of the following physical architectural components:

- ITLM Administration server
- ITLM Runtime server
- ITLM Agent

Figure 3-2 shows the relationships and communication between the elements that will be detailed in the following sections.

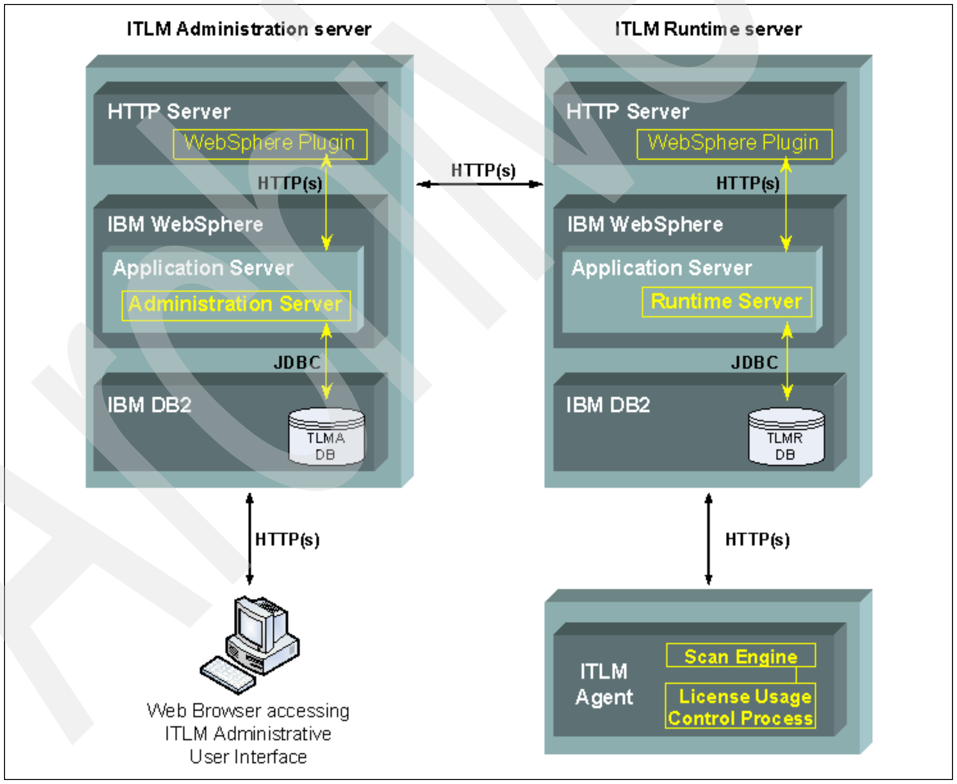


Figure 3-2 Physical components communication

3.3.1 ITLM Administration server

The ITLM Administration server is the core of the solution and works as the central arbitrator within the License Management strategy by providing the following services:

- Stores and maintains the information about products and licenses in a central database, which is organized by Organization.
- Gathers the software usage and installation data collected by the ITLM Agents and processed by the ITLM Runtime servers.
- Provides a administrative user interface that can be used to perform all the administration tasks and create historical reports of license usage and product installation over time.
- Forwards e-mail notifications to the license administrator upon detection of possible violation of software product usage.

There should be only one ITLM Administration server in any given IBM Tivoli License Manager environment.

3.3.2 ITLM Runtime server

An IBM Tivoli License Manager environment must have at least one ITLM Runtime server, but more servers can be set up to scale the solution and cover large sites. They are managed directly by the ITLM Administration server and are mainly in charge of the ITLM Agents' registration process and sending ITLM Agents software inventory information to the ITLM Administration server. In addition to that, each ITLM Runtime server provides the following functions:

- Receives the results of software scans from the ITLM Agents and processes them to build a view of the software installed on each machine.
- Provides a Web interface that is used to deploy ITLM Agents to Nodes.
- Forwards software catalog information containing known software signatures to ITLM Agents.
- Monitors the activity of the ITLM Agents, notifying the system administrator when an ITLM Agent has been stopped or removed from the system.
- Forwards e-mail notifications to the licenses administrator upon detection of events that have occurred on the ITLM Runtime server and its ITLM Agents.

3.3.3 ITLM Agent

A small agent footprint must be deployed on each of the client machines that are to be monitored by IBM Tivoli License Manager. ITLM Agents provide the primary interface of License Management. ITLM Agents run a very small amount of software, less than 500 KB of executable on all platforms.

Each ITLM Agent performs silently, and without any user intervention, the following functions:

- Executes a complete scan of the target machine, providing the ITLM Runtime server with the software and some hardware information collected. Hardware inventory information is limited to those aspects that are necessary for effective software asset management.
- ITLM Agent communicates with only one ITLM Runtime server. If the communication is not possible, the ITLM Agent stores data and sends it as soon as possible.
- Identifies the starting or stopping of software products and sends software usage information to the ITLM Runtime server. In case the ITLM Runtime server is not available at that time or the machine the ITLM Agent is running on is not connected to the network, ITLM Agents can cache software usage information and send it at a time when its ITLM Runtime server is available.
- Periodically checks for upgrades of the ITLM Agent code, so that the user intervention on the target machine is not required.

3.4 Logical components

This section provides an introduction to IBM Tivoli License Manager logical components. These components, as well as architectural decisions in which they are involved, are covered in great detail in Chapter 6, “IBM Tivoli License Manager logical design” on page 185.

Understanding the IBM Tivoli License Manager logical components helps with the definition of the IBM Tivoli License Manager supervising structure, also known as Topology, in support of the following operations:

- The management and organization, from a logical stand point, of elements that are to be monitored by IBM Tivoli License Manager.
- The definition of software entitlements and specification of rules to be used on monitored software usage.

An Organization is the highest level of the IBM Tivoli License Manager logical architecture and, whenever you log on to the ITLM Administration server, you must select an Organization in order to manage only its licenses and supervising structure. Each ITLM logical component must belong to an Organization, with the exception of the ITLM Administration server and some IBM Tivoli License Manager administrators, which can manage multiple IBM Tivoli License Manager Organizations.

The IBM Tivoli License Manager Topology defines the structure in which the Organization's components, such as ITLM Runtime servers, Divisions, and ITLM Agents, are organized. The IBM Tivoli License Manager Topology is composed of two types of logical components:

- Components that form the monitoring structure
- Components that record demographic information

In addition to the components that are part of the Topology, the following are also IBM Tivoli License Manager logical components:

- Components that support procurement management

Figure 3-3 provides an overview of all ITLM logical components and the links between them.

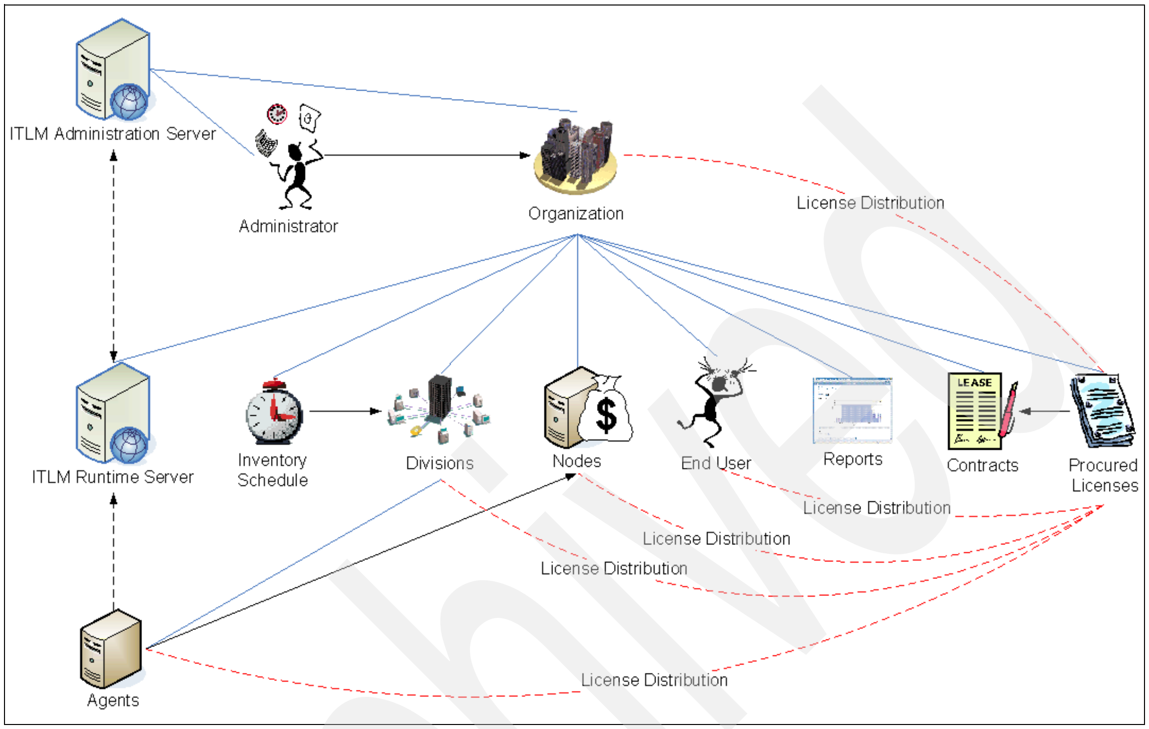


Figure 3-3 Logical components

The following sections go into detail about the above described logical components.

3.4.1 Monitoring structure components

For each Organization, a monitoring structure must be defined. This structure includes the following logical components:

- ITLM Runtime servers
- ITLM Agents
- Divisions

ITLM Runtime servers

ITLM Runtime servers must be first registered with the ITLM Administration server before they can be part of the monitoring infrastructure.

Each ITLM Runtime server will be communicating with its ITLM Administration server at predefined intervals, and they should preferably be located close to the ITLM Agents they will manage.

ITLM Runtime servers requires a RDBMS to store and access License Management related data that it collects from and sends to their ITLM Agents. The recommendation is that the database server hosting the ITLM Runtime server database be installed on the same computer as the ITLM Runtime server.

ITLM Agents

ITLM Agents must be deployed on nodes that are to be monitored and part of the License Management solution. The distribution of ITLM Agents among the ITLM Runtime servers to make the most efficient use of resources must be decided during the design phase of the IBM Tivoli License Manager implementation project.

ITLM Agents are the resources to be monitored for license usage. They must be subscribed physically to one ITLM Runtime server and logically to one Division, with the first subscription not being linked with the second.

ITLM Agents communicate to a single ITLM Runtime server. The ITLM Agent can communicate to a remote ITLM Runtime server in case its Division is logically split into different physical locations. We recommend having the ITLM Agent communication to a local ITLM Runtime server, regardless of the Division it belongs to.

When a new ITLM Agent is registered to an ITLM Runtime server, two recognition elements must be provided. These are a Name and a Computer Label, also referred to as the Node Tag. Both Name and Node Tag information must be unique for each ITLM Agent. The Name identifies the ITLM Agent to the ITLM Runtime server, which in turn links the ITLM Agent to a Node entry. The Node Tag is usually based more on business information than on technical characteristics. For example, in case an Asset Management system is already deployed in the enterprise, each machine must have an Asset Tag, which is normally linked with financial information or a cost center. By using the same information as a Node Tag in the IBM Tivoli License Manager environment, the two solutions could be integrated more easily, in order to, for example, charge the software license usages to a particular cost center.

Divisions

A Division is the second level of the IBM Tivoli License Manager logical architecture and is part of the Organization's Topology. It is used to group ITLM Agents so that they can be logically organized. Divisions can be selected for some administrative tasks, for example, to define different frequencies of software inventory scans, to create specific reports, and as a target of license entitlement distributions.

At least one Division must be defined per Organization. Each ITLM Agent should be subscribed to one and only one Division. Splitting the Organization structure to more than one Division gives more flexibility to manage the flow of inventory data. Even if each ITLM Agent only sends software inventory data that matches with a recognized software signature in the ITLM Master Catalog, having a huge number of ITLM Agents attached to the same Division may create a network bottleneck.

3.4.2 Demographic information components

For each Organization, data regarding demographic information can be stored in the ITLM Administration server database. This information includes the following logical components:

- Nodes
- Application Users

Nodes

A Node is a physical asset representation in an IBM Tivoli License Manager solution. It can be any workstation or server of an organization. For IBM Tivoli License Manager, the details, such as the location or description fields, of a Node should be related to the Asset Management and financial information implemented at the Organization's site.

This information can be imported from an Asset Management tool or from an Inventory solution that might have already been deployed in the organization. This import is performed using the IBM Tivoli License Manager XML import utility.

It is important to notice that each ITLM Agent of the Organization's Topology will have a corresponding Node. If a Node does not already exist for an ITLM Agent, during the ITLM Agent registration process, a new Node definition is automatically created. However, if a Node is manually created using the Administrative user interface, the ITLM Agent is not automatically installed on the Node.

Application Users

Application Users are part of the Organization's Topology. These are the users of software applications and the details of an Application User correspond to the ones defined for the specific Operating System login used by an end user. License distributions can be applied to Application Users and, therefore, licenses count may be performed based on the user ID that starts applications on monitored Nodes. Application Users definitions can also be imported by using the IBM Tivoli License Manager XML import utility.

3.4.3 Procurement management

Managing procurement in IBM Tivoli License Manager is divided into contract management and License Management tasks that have direct links between them, as well as analyzing license compliance reports.

For enterprises that use large databases to track ordering information for their software products, it is important to have a link between those databases and the License Management system that is used to monitor any license agreements.

IBM Tivoli License Manager can be used as a centralized repository for procurement information that is related to the information stored in the software information repository of the enterprise. Business information of licenses can be used to track the use of products in terms of the license agreements between enterprises and the vendors of the software.

For small to medium sized enterprises that do not use an alternative system for the management of contract data for procured software products, IBM Tivoli License Manager provides a contract management facility that allows direct linking between the contract data and the license data in the repository.

Administrators use IBM Tivoli License Manager to create contracts and link them directly to the licenses that they create. The legal terms and conditions data can also be added to the contracts and licenses definition in the repository.

IBM Tivoli License Manager procurement management activities are described in great detail in Chapter 6, “IBM Tivoli License Manager logical design” on page 185.

3.5 Software monitoring

IBM Tivoli License Manager provides a task that enables IBM Tivoli License Manager administrators to enable software usage monitoring for software products for which a signature is known. Defining the monitoring of products provides Organizations with the methodology to monitor software usage, even when the monitored Nodes are disconnected, and extract reports on software usage to analyze software license compliance according to the terms and conditions of a license agreement.

By enabling software usage monitoring, the ITLM Agent is able to collect usage statistics for the software product. IBM Tivoli License Manager by default does not collect usage information.

Software monitoring can be enabled in two distinct ways:

- When a software product is assigned to a license definition, IBM Tivoli License Manager automatically enables the usage monitoring for the software product.
- IBM Tivoli License Manager administrators can manually enable software usage monitoring for software products with no license defined, as long as a valid software signature is defined for the product in the ITLM Master Catalog.

Additional information about software monitoring can be found in Chapter 6, “IBM Tivoli License Manager logical design” on page 185.

3.6 ITLM Catalog Manager

The ITLM Master Catalog is a central repository of product information about all software components and related files for products that can be monitored by IBM Tivoli License Manager.

The ITLM Master Catalog resides in the ITLM Administration server database and a subset of it is periodically downloaded to each ITLM Runtime server. This subset of the ITLM Master Catalog, called Runtime Catalog, only includes those entries from the ITLM Master Catalog that relate to products that have been discovered running on nodes by ITLM Agents that are assigned to the ITLM Runtime server. A subset of the Runtime Catalog is also downloaded to each registered ITLM Agent.

IBM Tivoli License Manager includes an ITLM Catalog Manager tool that enables you to maintain the ITLM Master Catalog of products that need to be monitored. The ITLM Catalog Manager tool has a graphical interface that enables you to perform the following tasks:

- Update the ITLM Master Catalog with updates provided by IBM on a regular basis.
- Extend the ITLM Master Catalog with custom defined products using a hierarchical structure: Product, Version, Release, Component, and Signature.
- Create additional entries to the ITLM Master Catalog from unknown signatures that have been detected by ITLM Agents for which no corresponding entry exists in the ITLM Master Catalog.
- Manage signatures by defining custom signatures and assigning or removing them to components.

Figure 3-4 provides an overview of the ITLM Catalog Manager tool in relation to other IBM Tivoli License Manager components.

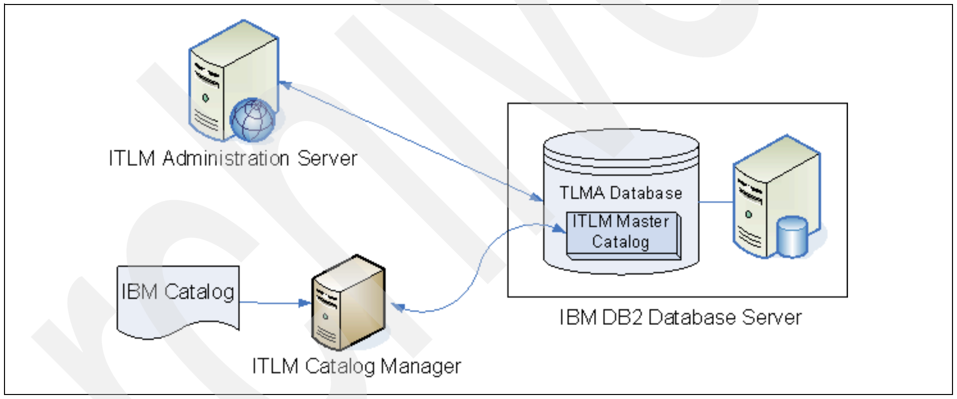


Figure 3-4 ITLM Catalog manager tool

Note: The ITLM Catalog Manager is not available in the *IBM Tivoli License Manager for IBM Software* package.

Catalog management tasks is of vital importance to the overall IBM Tivoli License Manager solution, as procurement management and software usage monitoring activities are dependent on software signatures defined and maintained as entries in the ITLM Master Catalog. In procurement management terms, licenses must be assigned to a software product signature in order to be valid and usable upon distribution. From a usage monitoring perspective, software usage monitoring is automatically enabled for software products with a valid license assigned to them, and manually defined by the IBM Tivoli License Manager administrator for software products that may not have a license defined to them. The IBM Tivoli License Manager administrator first selects a valid software signature for the product and then enables software usage monitoring based on the software signature of the software product.

Figure 3-5 shows the relationship between Catalog management, procurement management, and software monitoring tasks.

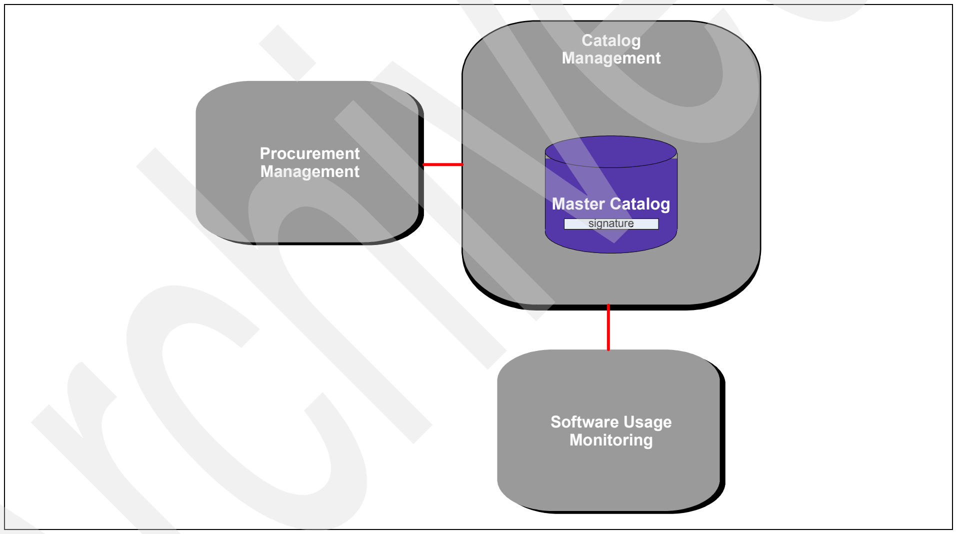


Figure 3-5 Catalog management relationships

Additional information about the ITLM Catalog Manager can be found in Chapter 6, "IBM Tivoli License Manager logical design" on page 185.

4

Solution design fundamental tasks

This chapter provides considerations for ensuring an effective implementation of a License Management solution using IBM Tivoli License Manager.

A License Management solution is dependent on a number of business and technical elements that makes planning an essential task in order to ensure a successful deployment of the solution.

When planning for a deployment of a License Management solution using IBM Tivoli License Manager, some activities are essential to the overall success of solution. These accomplishment of such activities lead to a more appropriate design of the License Management architecture, which includes the physical and logical designs of a IBM Tivoli License Manager implementation.

The following activities are discussed in this chapter:

4.1 Gathering solution requirements

The definition of requirements is an essential part of any successful project. They provide the foundation that enables the project to meet cost, schedule, and technical goals, as they are the basis for the solution design, development, test, and acceptance.

Being a subset of a large Asset Management solution or as a stand-alone project, requirements for the License Management solution must be gathered in order to define the scope of how IBM Tivoli License Manager will be implemented to satisfy the required services.

In general, requirements must obey certain policies, such as:

- Requirements must be baselined and changes to them must be controlled.
- Requirements must be developed, prioritized, and approved by the stakeholders of the project.
- Requirements must follow defined quality attributes.
- Requirements must be traceable.

Quality attributes policies

In regards to requirements following a defined set of quality attributes policies, this set of quality attributes policies can be global or individual attributes, as follows.

Global attributes

The global set of requirements must follow the following policies:

- Unique

  Requirements must not overlap or be redundant with other requirements.
- Complete

  Everything the solution is intended to perform throughout the solution's life cycle must be included. In addition, there must be no *to be defined* (TBD) or *to be reviewed* (TBR) statements.
- Consistent

  One or more subsets of requirements must not conflict.
- Comparable

  The relative necessity of the requirements must be included.

- Modifiable

  Changes to the requirements can be made easily, consistently (free of redundancy), and completely.

- Attainable

  Solutions exist within performance, cost, and schedule constraints.

Individual attributes

Each individual requirement must follow the following policies:

- Unambiguous

  The requirement must be clearly stated with a single interpretation.

- Understandable
- Correct

  The requirement must define the conditions and limitations that are desired, error free, and complete.

- Concise

  Unnecessary information must not be included in the requirement.

- Design independent
- Feasible

  The requirement can be implemented within project constraints.

- Verifiable

  A finite, cost-effective process has been defined to check that the requirement has been accomplished.

Requirements traceability

Requirements traceability demonstrates how each requirement must be tracked through the solution life cycle. Traceability of requirements must be performed until each requirement is delivered or withdrawn. Requirements traceability is bidirectional and includes:

- Tracking each requirement to the previous levels of requirements
- Tracking each requirement to design and build documentation
- Tracking each requirement to test documentation
- Tracking each requirement to acceptance criteria

4.1.1 License Management solution specific requirements

The goal of gathering License Management solution specific requirements is to itemize each relevant piece of information that will drive the high level design and implementation of the solution. These requirements will impact and influence the high level architectural decisions of the implementation of the License Management solution using IBM Tivoli License Manager.

As a best practice, these requirements must be detailed and maintained in a *High Level Solutions Requirement* document. This document must be used during the life cycle of the License Management project and reviewed on a regular basis. The final evaluation of the License Management solution will be graded and measured for quality regarding these requirements.

In the High Level Solutions Requirement document, each identified requirement must follow the attributes defined in 4.1, "Gathering solution requirements" on page 76 and contain as much relevant information as possible, such as, but not limited to the following:

- Requirement ID
- Requirement description with a rational behind the requirement
- Impact
- Priority
- Decision and actions

Requirements for a License Management solution can be classified into two categories as follows:

- Requirements that are global to the solution.
- Requirements that are specific to the products or applications that will be implemented for the solution. In our case, IBM Tivoli License Manager.

Global solution requirements

Table 4-1 on page 79 provides examples of global requirements that will influence high-level architectural decisions of a License Management solution. Obviously, this is not an exhaustive list, and is presented here to give a general idea of global requirements.

Table 4-1 Global solution requirements sample

ID	Description	Impact	Priority	Decisions and actions
GTLM01	Maintains records of software inventory to a level of accuracy in line with agreed service levels.	High	High	Accuracy level must be defined in the Service Level Agreement (SLA) document. A Quality Assurance (QA) process must be defined to control the accuracy of data provided by IBM Tivoli License Manager.
GTLM02	Proactive scan of non authorized software.	Low	High	Define and maintain a non-authorized software list. Define the process for tracking and removal of non authorized software.
GTLM03	Provides reports on License Management and compliance.	High	High	Analyze IBM Tivoli License Manager out of the box reports. Define new reports if necessary.
GTLM04	Identifies violations of software license agreements.	High	High	Make architectural decision of the meaning of software license violations to the enterprise. Use IBM Tivoli License Manager software usage monitoring capabilities to monitor software usage and software installations for license compliancy.
GTLM05	Licenses Contract management	High	High	Identify the existing software procurement process adopted by the enterprise. Determine at which levels in the enterprise licenses are procured (corporation, division, department, user, and so on).
GTLM06	Identifies underusage of software licenses and creates a process for defining a threshold of minimum use of software licenses to trigger licenses renegotiation.	High	Medium	Identify and create detailed reports to display the number of used/installed software showing maximum and minimum values, as well as deviation between the number of used/installed and procured software licenses.

ID	Description	Impact	Priority	Decisions and actions
GTLM07	Obeys Data Privacy laws and considerations. Some countries do not allow storing or disclosing user information either in the local country or outside the country the user is located.	Medium	Medium	Identify Data Privacy laws for each country involved in the solution.

IBM Tivoli License Manager specific solution requirements

Table 4-2 provides examples of requirements that will influence high level architectural decisions of the physical and logical designs of the IBM Tivoli License Manager implementation. Obviously, this is not an exhaustive list, and is presented here to give a general idea of requirements.

Table 4-2 IBM Tivoli License Manager specific requirements sample

ID	Description	Impact	Priority	Decision and actions
STLM01	Introduces the notion of the IBM Tivoli License Manager V2.2 logical architecture hierarchy. As the scalability limit of the product is reached regarding the number of ITLM Agents, the IBM Tivoli License Manager logical hierarchy notion (Organization and Divisions) is very important and will drastically impact the physical and logical approach and constraints.	High	High	Determine at which levels in the enterprise licenses are procured. Determine the number (total and per location) of nodes to be managed. Consider Data Privacy laws for each country involved. Determine how many IBM Tivoli License Manager environments are required to fulfill License Management requirements.

ID	Description	Impact	Priority	Decision and actions
STLM02	IBM Tivoli License Manager servers placement. This is a essential requirement of the solution, as it has a high impact on the architecture of the physical design of the solution.	High	High	Determine the number (total and per location) of ITLM Agents, users, and software application characteristics, as well as Data Privacy laws per country, to decide on the number of ITLM Administration servers and whether an ITLM Runtime server is needed in the location. The number of ITLM Runtime servers will depend, among other elements, on budget restrictions and IBM Tivoli License Manager scalability limits. Obtain a network architecture diagram to decide the impact on the network. The highest traffic is between ITLM Agents and their assigned ITLM Runtime server.
STLM03	Software license metering. The desired level of software license metering and monitoring affects the way ITLM Agents request licenses and can have an impact on network traffic and user response time.	Medium	Medium	Define an approach for license metering. For example, monitor the usage of software that is a member of a predefined list of software defined in the ITLM Master Catalog. Define the IT process to update the software list on a regular basis. Define the IT process to manage the usage of software that is not part of the ITLM Master Catalog.

ID	Description	Impact	Priority	Decision and actions
STLM04	Software Entitlement. Entitlements are based on software licensing contract information and describe the terms and conditions under which the product is licensed.	High	High	An IT process must be defined in order to maintain Software Entitlement information, such as changes made to the software licensing contract, expiration dates, and so forth. Although IBM Tivoli License Manager can provide software usage information without any entitlement definition, it is not possible to reconcile software usage and license usage unless Software Entitlements are defined. For Electronic Entitlements, define the IT process for obtaining and importing the XML file, provided by IBM, which contains the terms and conditions under which the software is licensed.
STLM05	Procurement Management. This relates to software contract management and software license data described in the contract.			Define the IT process for software contract management. Create and identify administrators for the procurement manager role. If the enterprise uses a system to track software ordering information, identify the interface between those systems and IBM Tivoli License Manager.

ID	Description	Impact	Priority	Decision and actions
STLM06	Application users and mobile users. An application user is the user who can start software applications on nodes. The details of application users are maintained to allow licenses to be restricted to specified users.	Medium	Medium	Identify software licenses entitlements by user capacity level. Define IT process for defining, updating, and removing application users in IBM Tivoli License Manager. Determine if application user information can be imported from existing user management systems. Define policies for offline software use, in case mobile users are disconnected.
STLM07	Integration with existing Asset Management tools.	Medium	Medium	Define interfaces to be used for integration. Identify which information should be exchanged between IBM Tivoli License Manager and Asset Management tools. Define the IT process for data exchange.
STLM08	Test and validation environment. A test and validation environment is needed during the High and Detailed design phases. Cost permitting, this environment can also be used during the life cycle of the solution to test and validate each change before implementing them into production.	Medium	High	Allocate hardware and software resources. Determine if the test and validation environment will be kept throughout the life cycle of the solution. Implement/Modify IT Change Management process to include IBM Tivoli License Manager test and validation procedures.

ID	Description	Impact	Priority	Decision and actions
STLM09	IBM Tivoli License Manager Administrators	High	High	Identify the team and individuals to perform the various IBM Tivoli License Manager administrative roles. Each role has a fixed set of assigned tasks. Define IT process for creating and assigning roles to IBM Tivoli License Manager Administrators.

4.2 Defining the scope of the solution

Defining the scope of the License Management solution is perhaps the most important part of the up front definition and planning process. However, the scope of the License Management solution can only be defined once all the requirements are gathered and outlined.

If there is no definition of what the License Management solution is supposed to deliver and what the boundaries of the project are, there is very little chance for success.

For a License Manager solution using IBM Tivoli License Manager, at a minimum, the following items must be part of the scope definition:

- Deliverables

  A list of the deliverables (documents, reports, and so on) that will be supplied by the solution
- Target

  A list of the groups and levels in the enterprise to which License Management will be performed. For example, Human Resource, Accounting, and Finance.
- Functionality

  Describe the major features and functions of IBM Tivoli License Manager that are in and out of scope and to which Targets.

  For example, consolidation of information about products installed and software use, association of contracts and licenses, creation of licenses using information imported from an electronic entitlement file, procurement management, and so on.

There are many other categories of scope that can be used to describe the boundaries of the solution. The more aspects of scope that are identified, the better off the solution will be.

4.3 Making high level architectural decisions

Once requirements for the License Management solution (both global and IBM Tivoli License Manager specific) are gathered as well as the scope of the License Management solution have been defined, high level architectural decisions are made to facilitate the physical and logical designs of the IBM Tivoli License Manager implementation.

These decisions and policies are indispensable, fundamental, and strategic. They must be made, used, and respected throughout the life cycle of the License Management solution.

The following list provide examples of high level architectural decisions and policies. It is not an exhaustive list, but can be enhanced, depending on the list of requirements, and used as a start point for any License Management solution.

- Software license compliance is defined as the process of monitoring and enforcing policies defined by the license contract, ensuring that there is no non-conformance.
- Effective License Management cannot be achieved without auditing the nodes hosting the software. An Software Audit process must be defined to examine software license compliance.
- A software license violation occurs when the number of procured licenses for a software application are overtaken.
- A software license abnormality occurs when under usage of procured licenses for a software application is detected.
- Passive License Management is defined as software licensing monitoring (usage and installation) with no real time license checks nor immediate actions in case of license violations or abnormality.
- Active License Monitoring is defined as software licensing monitoring (usage and installation) with real time license checks and may enforce license policies defined for the software application in case of a license violation.

- Software applications can be categorized, but not limited to, into four groups:

Known Software	Software for which an product entry is defined in the ITLM Master Catalog.
Unknown Software	Software with no product entry defined in the ITLM Master Catalog.
Authorized Software	Software products to which licenses are defined.
Unauthorized Software	Software products with no licenses defined.

- The scan process to count the number of Known Software installed on nodes will occur on a regular basis, for example, twice a month. This scan process must be scheduled at different times for different groups of nodes, based on the IBM Tivoli License Manager topology definition.
- The usage monitoring for Authorized Software and need to be monitored will occur in real time but in an offline mode, meaning that usage statistic information will be first stored on the end user node and then collected on a regular basis. For example, once a week.
- The usage monitoring for Known Software that are Unauthorized that needs to be monitored on demand for tracking and analysis purposes will occur in real time but in an offline mode, meaning that usage statistic information will be first stored on the end-user node and then collected on a regular basis, for example, once a week.
- The tracking of Unauthorized Software will occur in real time but in an offline mode, meaning that usage statistic information will be first stored on the end-user node and then collected on a regular basis, for example, once a day.
- The scan process to count the number of Unknown Software installed on nodes will occur under the control of a Software Audit process.

Note: Data Privacy laws defined in some countries do not allow scanning for unknown information. This is one of the reasons to clearly define the Unknown Software tracking as a Software Audit process.

- Active License Monitoring must be enabled only if required by licenses contracts or business need.

In every implementation of a License Management solution, using a set of guidelines must be followed in order to properly create the physical and logical designs of the solution. The following high level architecture policies and best practices relates to the physical and logical designs:

- A document must be created as the unique source of information for both the physical and logical designs for the License Management solution. This document must be owned by the IT Architect in charge of such designs and referred to by anyone involved in the implementation of the solution.
- The physical and logical design must be built initially in the test and validation environment and rolled out into the production after a validation agreement or a regression testing agreement in case of design changes.
- Any changes to the physical and logical designs must be documented and presented to the team responsible for the validation tests.
- Any changes to the physical or logical design of the solution done after an integration testing environment and requirements validation task have been initiated, should be reproduced and validated in the test and validation environment.
- The solution will be rolled out to the production environment upon agreement and acceptance of the solution by all parties involved in the License Management solution.

5 IBM Tivoli License Manager physical design

This chapter addresses the physical design for the implementation of IBM Tivoli License Manager. The physical design develops the underlying physical infrastructure on which the solution will operate. Sufficient time needs to be allocated to ensure that the correct design has been developed, because when deployed and operational, the physical design may be difficult to change without a disruption of the IBM Tivoli License Manager environment.

The following topics are discussed in this chapter:

Note: This redbook refers to the IBM Tivoli License Manager V2.2 product, which has been recently rebranded as IBM Tivoli License Compliance Manager V2.2.

The IBM Tivoli License Manager product name is still in use in this redbook as well as on all of the administrative user interfaces and in the documentation of the product.

Refer to the product's Release Notes, as they provide the most current information about IBM Tivoli License Compliance Manager V2.2. The latest version of the Release Notes can be found on the Tivoli Information Center Web site:

`http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?toc=/com.ibm.itlm.doc/toc.xml`

5.1 Physical components considerations

In order to have the best infrastructure deployed to support the requirements of the License Management solution using IBM Tivoli License Manager, it is essential to understand what the physical components involved are, their role, functionality, scalability, limitations, dependencies, and so on.

Having this knowledge in perspective, as well as well documented requirements, scope definition, and high level architectural decisions, it is possible to produce the physical design of a IBM Tivoli License Manager implementation.

In this section, we provide general considerations regarding the physical components of the IBM Tivoli License Manager architecture as follows:

- 5.1.1, “Environment sizing and scalability limits” on page 91
- 5.1.2, “ITLM Administration server” on page 94
- 5.1.3, “ITLM Runtime server” on page 95
- 5.1.4, “Database environment” on page 97
- 5.1.5, “ITLM Catalog Manager and other management tools” on page 98

5.1.1 Environment sizing and scalability limits

Some architectural and tuning decisions are dependent on the size of the environment to which IBM Tivoli License Manager will be deployed. This is especially important during the physical design conception phase. This section helps define the size of the environment.

In terms of scalability of the environment on which the License Management solution using IBM Tivoli License Manager operates, the following are the recommended maximum numbers:

- A recommended maximum of 45,000 ITLM Agents per ITLM Administration server
- A recommended maximum of 15,000 ITLM Agents per ITLM Runtime server
- A recommended maximum of 20 ITLM Runtime servers per ITLM Administration server

However, the above numbers are *not* a hard coded limit and could be surpassed by utilizing a powerful hardware infrastructure. These limits are dependant on multiple factors, such as the number of software products monitored, and the number of software start and stop events.

Table 5-1 and Table 5-2 provide the minimum hardware requirements for the physical elements of a IBM Tivoli License Manager solution. Always refer to *IBM Tivoli License Manager v2.2 Planning, Installation, and Configuration*, SC32-1431 for the latest information.

Table 5-1 Minimum CPU requirements: all IBM Tivoli License Manager servers

Operating system platform	**Minimum CPU requirement**
Windows and Linux® on xSeries® servers	Intel® Pentium® IV 1.5 GHz
AIX® and Linux on pSeries®	RS/6000® Model 7044/270, two processor IBM RS/6000 375
Solaris™	Sunblade 2000 model, two Ultrasparc III processors
HP-UX	rp2470, two PA Risc 2.0 650 MHz processors
Linux on zSeries®	One dedicated processor
Other platforms	No special requirements

Table 5-2 Minimum RAM requirements

IBM Tivoli License Manager physical component	**Physical memory requirement**
ITLM Administration server	1 GB RAM
ITLM Runtime server	1 GB RAM
ITLM Administration server and database server on same machine	2 GB RAM
ITLM Runtime server and database server on same machine	2 GB RAM
ITLM Administration server, ITLM Runtime server, and database server on same machine	3 GB RAM

In order to facilitate the sizing and scalability explanations that follow in the remaining chapters, for the purposes of this redbook, we define three values to define the size of an IBM Tivoli License Manager environment: small, medium, and large. They are defined as follows:

- Small IBM Tivoli License Manager environment
  - Less than 4,000 managed ITLM Agents
  - Up to two ITLM Runtime servers
  - ITLM Administration server and Database server hosting the ITLM Database, possibly installed on the same physical machine
- Medium IBM Tivoli License Manager environment
  - Up to 20,000 managed ITLM Agents
  - Up to 5 ITLM Runtime servers, possibly in different geographies
  - ITLM Administration server and Database server hosting the ITLM Database, possibly installed on the separate machines
- Large IBM Tivoli License Manager environment
  - Above 20,000 managed ITLM Agents
  - More than five ITLM Runtime servers spread out in different geographies
  - Number of nodes up to 5,000 per remote site per ITLM Runtime server
  - Possible management of more than one ITLM Organization, in case the consolidated License Management is not a requirement.
  - ITLM Administration server and Database server hosting the ITLM Database installed on the separate machines

Specifically for IBM Tivoli License Manager *V2.1*, for large environment, having an estimate on the number of start and stop events can be helpful to determine the scalability limits of the environment. This is not a relevant issue with IBM Tivoli License Manager V2.2.

Table 5-3 on page 94 provides some scenarios for the recommended number of ITLM Agents based on the number of software start and stop events. These numbers are based on scalability tests performed by the IBM Tivoli License Manager Development and Testing teams, assuming the following factors:

- Number of hours of ITLM Agent operation a day: 10
- Maximum number of entries in the ITLM Runtime server database a day. Specifically, in the USAGE table: 2.0 million
- Products detected per scan per node: 250 (this is significantly more than the average 60 products on a typical workstation)

Table 5-3 Scalability based on the Agent activity

Number of start and stop events per hour per Agent	Recommended maximum number of Agent per IBM Tivoli License Manager environment
4	45,000
3	65,000
2	72,000

5.1.2 ITLM Administration server

Each IBM Tivoli License Manager installation must have one ITLM Administration server. The ITLM Administration server provides the following facilities:

- A central repository of product, license agreement, license usage, inventory, and Organization information
- A Web interface that can be used to perform License Management and administrative tasks and to produce historical reports of license usage and inventory information over time

The ITLM Administration server can serve more than one Organization, provided that the total number of ITLM Runtime server and ITLM Agents that connect to it do not go above the recommended scalability limits, which will decrease performance to unacceptable levels.

The ITLM Administration server is generally placed close to where the ITLM Administrators are located. It should also be connected to a network segment that could easily serve all ITLM Runtime servers. However, one of the most important requirements for the ITLM Administration server placement is the location of the database server hosting the ITLM Administration server database. A high speed connection between the ITLM Administration server and the database server must be available.

The ITLM Administration server supports the following services:

Runtime Management Service	Manages the ITLM physical topology.
Core Service	Controls ITLM core tasks, such as software monitoring usage data aggregation, software licenses definitions, and so on.
Web User Interface Service	Provides administrative tasks, and controls user access and actions.

Architectural decision example: Based on the requirement that licenses must be accounted for at the corporate level, as well as the fact that there are less than 45,000 ITLM Agents in the scope of this project, only one world wide ITLM Administration server will be deployed. Every ITLM Runtime server will be managed by this single ITLM Administration server. This ITLM Administration server will be placed in the enterprise's main data center.

5.1.3 ITLM Runtime server

Each IBM Tivoli License Manager installation must have at least one ITLM Runtime server. Each ITLM Runtime server provides the following facilities:

- An intermediary repository for information about the software installed on the ITLM Agents that the ITLM Runtime server manages.
- The capability to generate and send e-mails to provide notification about events that have occurred on the ITLM Runtime server or its ITLM Agents.
- A Web interface that can be used to deploy the ITLM Agents to nodes.

Assigning ITLM Agents to ITLM Runtime servers is a critical portion of the Physical Design. Performance, scalability, and the amount of data ITLM Agents send to the ITLM Administration server using the ITLM Runtime server are the main points for the ITLM Runtime server number and placement.

Each ITLM Runtime server will be communicating to the ITLM Administration server at predefine intervals, and they should be located relatively close to their ITLM Agents within the physical network. This minimizes network traffic when ITLM Agents send software monitoring and other data. Another reason for putting the ITLM Runtime server close to the ITLM Agents is the flow of software inventory coming from the ITLM Agents to their respective ITLM Runtime server. The inventory information is then consolidated and only the delta information is sent to the ITLM Administration server.

We recommend that the database server hosting the ITLM Runtime server database be installed on the same server as the ITLM Runtime server.

The placement and number of ITLM Runtime servers will depend on the following:

- Physical design elements: network topology, hardware specifications, and so on
- Logical design elements: number of ITLM Organizations, and so on
- Scalability limits

We advise having a maximum of 15,000 ITLM Agents connected to the same ITLM Runtime server. As a best practice, we advise setting an upper limit to the number of ITLM Agents for each ITLM Runtime server that is much lower, for example, 10,000. The main reason is to have very scalable ITLM Runtime server architecture regarding the overall physical design. When determining the upper limit, take into consideration the combination of the maximum number of supported ITLM Runtime server by a single ITLM Administration environment and the number of ITLM Agents in scope.

Of course, if benchmark information demonstrates that the ITLM Runtime servers are underutilized, an ITLM Runtime server consolidation task could be planned, while still respecting the scalability limits of the solution.

- Budget allocation for the project

  If the number of ITLM Agents in a remote location does not justify the investment of an additional ITLM Runtime server dedicated to the location, those ITLM Agents can be managed by a remote ITLM Runtime server, and some optional security configurations can be used as well to ensure data safety.

  On the other hand, having a failover ITLM Runtime server for high profile locations is advisable.

- Data Privacy laws of the locations involved

  Some countries Data Privacy laws can be restrictive of the centralization of software usage information outside of that country. Having a dedicated local ITLM Runtime server managing the ITLM Agents of that country simplifies the process of enabling the Data Privacy and Audit processes. It also makes it easier to allocate a dedicated IBM Tivoli License Manager environment for the country in the case where Data Privacy laws are very restrictive.

  The maximum number of ITLM Runtime server per location will always mainly depend on the number of ITLM Agents in that location.

Architectural decision example: A minimum of one ITLM Runtime server is planned to be installed at each of the enterprise’s branch offices. This supports the Data Privacy laws of the country in case a branch office is located abroad. In the context of this example, the maximum number of ITLM Agents managed per ITLM Runtime server is 7,500. This ensures a highly scalable architecture solution with an estimate growth rate of 20% a year for each branch office. Also, the database server hosting the ITLM Runtime server database will be installed on the same physical machine as the ITLM Runtime server.

5.1.4 Database environment

Both the ITLM Administration server and ITLM Runtime server rely on a database infrastructure to store license and software monitoring data. The ITLM Catalog Manager accesses the ITLM Administration server database directly to perform software catalog maintenance tasks.

IBM Tivoli License Manager V2.2 supports IBM DB2 UDB Enterprise Server Edition server V8.2. This prerequisite application is shipped with IBM Tivoli License Manager V2.2 with restricted-use licenses that allow it to be used only in association with the use of IBM Tivoli License Manager. The licenses also authorize downloading, installing, and using any fixes issued for the product.

Estimate the minimum database size as 500 MB for the ITLM Administration server database and 200 MB for the ITLM Runtime server database, plus a quota for each ITLM Agent that is deployed. For the ITLM Administration server database, estimate the quota as 400 to 500 KB for each ITLM Agent. For each ITLM Runtime server database, estimate the quota as 300 to 400 KB of disk space for each ITLM Agent that reports to the ITLM Runtime server.

For large environments, the recommendation for the installation of the database server for the IBM Tivoli License Manager server components is as follows:

- The ITLM Administration server database and the ITLM Administration server installed on separate machines. In this case, ensure that a high speed connection is available for the communication between ITLM Administration server and the database.
- The ITLM Runtime server database can be installed on the same computer as the ITLM Runtime server component.

In terms of access to databases, IBM Tivoli License Manager V2.2 no longer requires IBM DB2 Client as prerequisite for IBM Tivoli License Manager servers. Also, if the database is installed on a different computer, ensure that a high-speed connection is used between the IBM Tivoli License Manager server components and the database server.

In case the scalability limits of the IBM Tivoli License Manager environment are reached, and depending on the enterprise requirements for consolidating License Management information, the License Management solution may require that additional IBM Tivoli License Manager environments be put in place. Each of these IBM Tivoli License Manager environments are controlled by a single ITLM Administration server. These ITLM Administration servers cannot share the same ITLM Administration server database.

Architectural decision example: Based on the requirement that License Management information must be consolidated at enterprise level, only one world wide ITLM Administration server database will be defined. This unique database will be created on an IBM DB2 UDB server running on a dedicated computer, separate from the ITLM Administration server for performance reasons. This IBM DB2 UDB server will be placed in the enterprise's main data center. Additional databases will have to be defined at ITLM Runtime server level and will be installed on the same physical machine as the ITLM Runtime server.

5.1.5 ITLM Catalog Manager and other management tools

As described in Chapter 3, "IBM Tivoli License Manager general overview" on page 59, the full version of IBM Tivoli License Manager provides a tool for managing the ITLM Master Catalog. As a general practice, the ITLM Catalog Manager tool and the ITLM Administration server are installed on separate servers. The common rule is to install the ITLM Catalog Manager tool on the ITLM Administrator's workstation that is responsible for managing the ITLM Master Catalog. Refer to *IBM Tivoli License Manager Planning, Installation, and Configuration*, SC32-1431 for a complete list of supported platforms and prerequisites of the ITLM Catalog Manager. Additional details on the ITLM Catalog manager and related topics is provided in Chapter 6, "IBM Tivoli License Manager logical design" on page 185.

IBM Tivoli License Manager physical components rely on prerequisite applications like IBM DB2 UDB and IBM WebSphere Application Server. These applications have their own management tools, such as the IBM DB2 Control Center and the IBM WebSphere Administrative Console.

For large environments, a dedicated server is commonly used for management purposes. This server, named the ITLM Management server is used to host the prerequisite management tools described above, as well as the ITLM Catalog Manager tool. The use of the ITLM Management server simplifies management operations and provides a good level of security. Considering that the ITLM Administration server database resides in a network protected area in the data center, only incoming traffic from the ITLM Management server would be authorized through the firewall.

Another common use for the ITLM Management server is as a stand by ITLM Administration server that is configured to access the ITLM Administration server database and manage the IBM Tivoli License Manager environment, in case the production ITLM Administration server is unavailable for any reason.

Architectural decision example: A dedicated server will be allocated to host the ITLM Catalog Manager tool as well other required management tools, such as the IBM DB2 UDB Control Center. This dedicated server will also host a backup ITLM Administration server in standby mode. This backup ITLM Administration server will be active in case the production ITLM Administration server is not available.

5.2 Communication type and security options

This section describes the considerations regarding overall communication between the physical components. It summarizes the rationale behind each type of communication, protocol, and ports defined.

IBM Tivoli License Manager V2.2 is compliant with the Federal Information Processing Standards (FIPS) 140-2, which are standards and guidelines issued by the National Institute of Standards and Technology (NIST) for federal government computer systems. FIPS 140-2 is the standard that defines the security requirements for cryptographic modules that are used within a system that handles sensitive but unclassified information.

IBM Tivoli License Manager mainly uses the HyperText Transfer Protocol (HTTP) for communication between the physical components. The default TCP port used in this case is 80. Communication between components can also be secured using Secure Sockets Layer (SSL) communications, and to select SSL cipher-suites that are FIPS 140-2 certified when information is transmitted between elements of the infrastructure. For additional information, refer to *IBM Tivoli License Manager Security Manager*, SC32-1502.

Figure 5-1 shows the relationships and communication between the physical components of an IBM Tivoli License Manager solution.

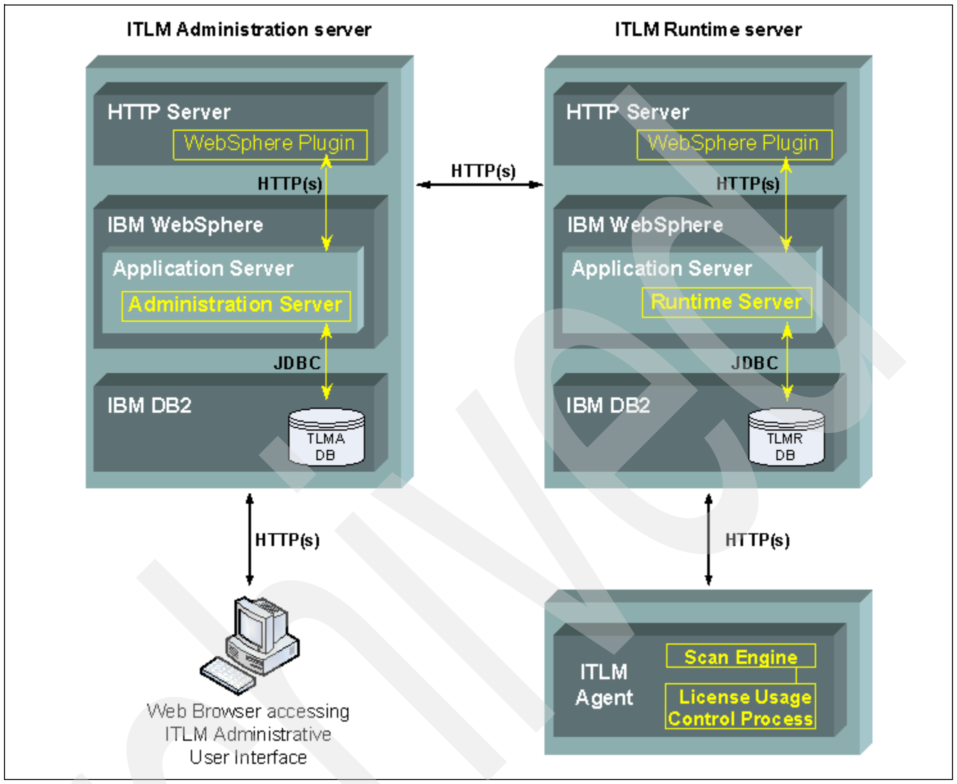


Figure 5-1 Physical components communication

Table 5-4 can be used as reference when configuring secure communications between IBM Tivoli License Manager components. It provides the nature of the request, and the initiators and the target of the communication request.

Table 5-4 Communication type references for IBM Tivoli License Manager

Communication	Initiator	Target
Runtime services	ITLM Runtime server	ITLM Administration server
Agent services	ITLM Agent	ITLM Runtime server
Administrative services	ITLM Administrator's Web Browser	ITLM Administration server

In Table 5-4 on page 100, Agent services refer to service requests from the ITLM Agent, such as a request to upload software usage information, or a request to obtain the portion of the ITLM Master Catalog controlled by its ITLM Runtime server. Runtime services refer to service requests from the ITLM Runtime server, such as a request to download the portion of the ITLM Master Catalog that concerns the ITLM Runtime server or a request to upload software usage information collected from the ITLM Agents.

In terms of a physical design solution of an IBM Tivoli License Manager implementation, security mainly refers to the following:

- Confidentiality

  No one can read the exchanged data.

- Integrity

  Data is sent and received without modifications.

- Authentication

  One entity is sure of the identity of the other one (can be mutual or not).

- Authorization

  An authenticated entity can access or not a requested resource.

The following sections go into details on how the above security topics can be addressed when preparing a physical design of an IBM Tivoli License Manager implementation:

- 5.2.1, "Security levels between physical components communications" on page 101
- 5.2.2, "Secure Sockets Layer communications" on page 105
- 5.2.3, "Administrator's authentication options" on page 106
- 5.2.4, "Proxy server" on page 107
- 5.2.5, "Firewall communications" on page 107

5.2.1 Security levels between physical components communications

As presented in Table 5-4 on page 100, there are three types of communication services; each of those types provide different security levels for communication that is independent from each other.

The following sections go into details about the security levels of each of the communication types.

Runtime services communication

IBM Tivoli License Manager provides the following security levels for communications between the ITLM Runtime server and the ITLM Administration server:

- Minimum

  This communication security level provides no confidentiality and does not ensure integrity of data. The communication is initiated by the ITLM Runtime server using the HTTP protocol. The ITLM Administration server accepts communication requests without performing authentication.

- Maximum

  This communication security level provides confidentiality and ensures integrity of data. Data is only exchanged after a mutual authentication. ITLM Runtime servers authenticate to the ITLM Administration server using a shared password. ITLM Administration server authenticates to the ITLM Runtime server using a SSL server certificate.

In all communications between these physical components, communication is initiated by the ITLM Runtime server, as presented in Figure 5-2 on page 103. Notice that when security level is set to maximum, communications use SSL and include a password for authentication. A truststore on each server must be set to store the password for authentication. This is further explained in 5.2.2, “Secure Sockets Layer communications” on page 105.

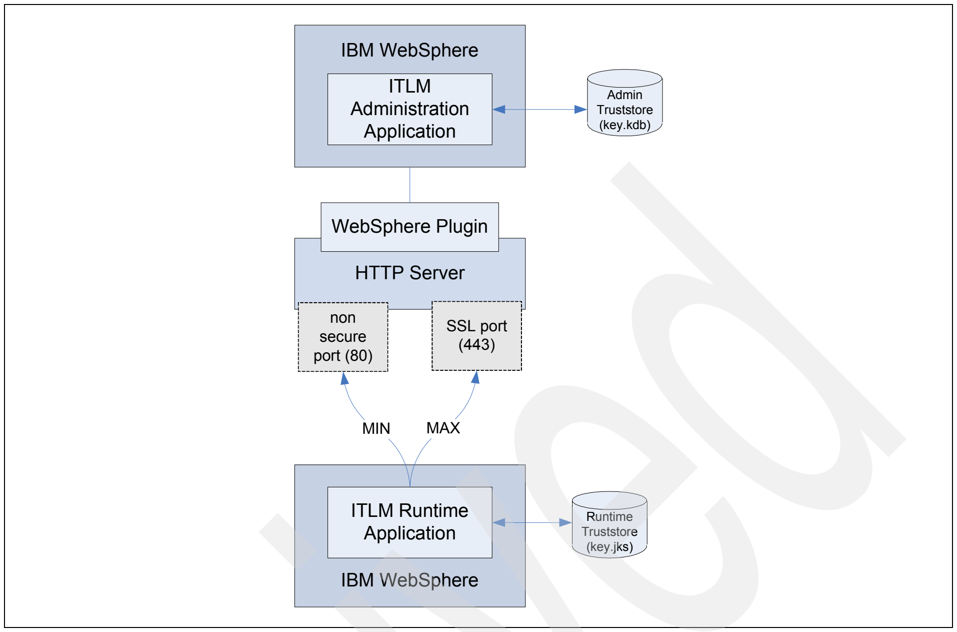


Figure 5-2 Runtime services communication security levels

Agent services communication

IBM Tivoli License Manager provides the following security levels for communications between ITLM Runtime server and ITLM Agents:

- Minimum

  This communication security level provides no confidentiality and does not ensure the integrity of data. The communication is initiated by ITLM Agents using plain HTTP protocol. The ITLM Runtime server accepts communication requests without performing authentication.

- Medium

  This communication security level provides confidentiality and ensures the integrity of data. Data is only exchanged after a server authentication. After the ITLM Agent initiates the communication request, the ITLM Runtime server authenticates to the ITLM Agent using an SSL server certificate.

- Maximum

  This communication security level provides confidentiality and ensures the integrity of data. Data is only exchanged after a mutual authentication. ITLM Runtime servers authenticate to the ITLM Agent using an SSL server certificate. ITLM Agent authenticates to the ITLM Runtime server using an SSL client certificate.

In all communications between these physical components, communication is initiated by the ITLM Agent (Figure 5-3). Notice that when the security level is set to medium or maximum, communications use SSL certificates. A truststore on both ITLM Agent and ITLM Runtime server must be set to store the password for authentication. This is further explained in 5.2.2, “Secure Sockets Layer communications” on page 105.

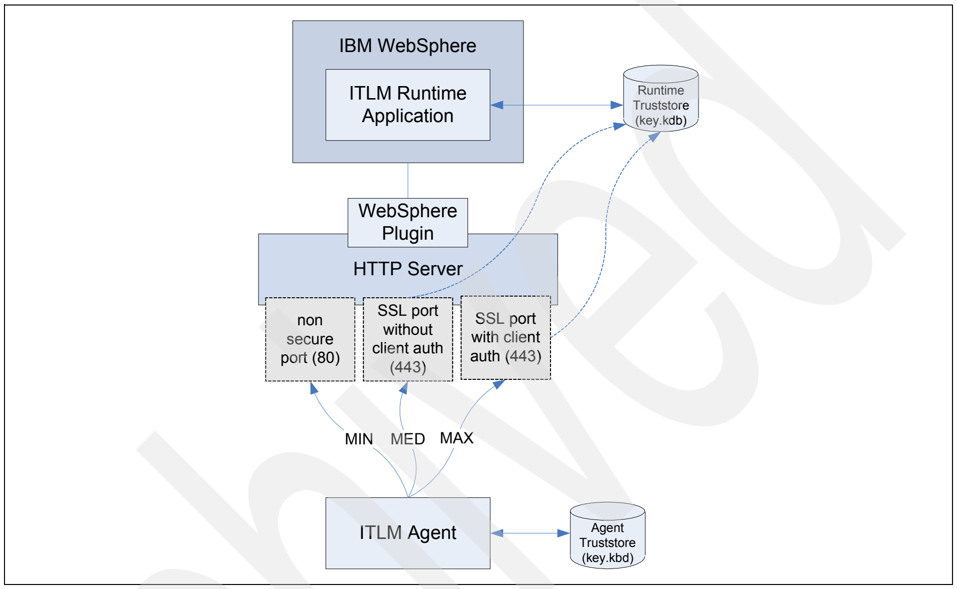

IBM WebSphere
ITLM Runtime Application
Runtime Truststore (key.kdb)
WebSphere Plugin
HTTP Server
non secure port (80)
SSL port without client auth (443)
SSL port with client auth (443)
MIN
MED
MAX
ITLM Agent
Agent Truststore (key.kbd)

Figure 5-3 Agent requests communication security levels

The security level defined for a ITLM Runtime server applies to all the ITLM Agents that communicate with the server. These security settings are some of the parameters downloaded to each ITLM Agent when the newly deployed ITLM Agent first connects to the ITLM Runtime server. Any change to the security level in the ITLM Runtime server configuration is downloaded to the ITLM Agent as part of the regular parameter download.

However, The ITLM Agent can only obtain these security settings after the first successful communication with its ITLM Runtime server. The security level defined to the newly deployed ITLM Agent for the first communication *must* match the security level of the ITLM Runtime server at the time of the ITLM Agent deployment.

This initial security setting for the ITLM Agent is of crucial importance during the ITLM Agents rollout phase of the physical design implementation.

Administrative services communication

IBM Tivoli License Manager provides the following security levels for communications between ITLM Administration server and administrators using the Web interface:

- Secure communications for all data transmissions
- Secure communications for sensitive data only
- Communications using plain HTTP protocol

5.2.2 Secure Sockets Layer communications

Secure communications using Secure Sockets Layer (SSL) can be enabled in IBM Tivoli License Manager between the ITLM Administration server and ITLM Runtime server and between ITLM Runtime server and ITLM Agents. In this case, the default TCP port used is 443. Encryption requires both parties to the communication to do extra work in exchanging handshakes and encrypting and decrypting the messages, making this form of communication slower than pure HTTP communications.

Secure Sockets Layer is a message transportation protocol that provides the following advantages:

- Authentication

  The origin of all messages is assured. The recipient of the message has the assurance that the received message came from the expected sender.

- Reliable

  The message transport uses a message integrity check (using a MAC) that ensures the quality of the data being transmitted.

- Private

  Messages between the components are encrypted, after a handshake to define a secret key. This ensures that the contents of the messages cannot be read by a third party.

If all of the components are behind a firewall, or some other means of protection, and do not require encryption, privacy can be disabled without comprising the authentication and reliability aspects of SSL.

Secure Sockets Layer passwords

Each ITLM Administration server and ITLM Runtime server of the IBM Tivoli License Manager solution that is enabled to use SSL communications requires a password to access its SSL key store. Whether or not SSL is enabled, the IBM Tivoli License Manager Server will be installed with a key store, and the password used to access that keystore will have a default value. Thus, this value applies, either if SSL is enabled at installation, or if it is enabled at some point after installation. The default password is *common* to all IBM Tivoli License Manager users, and should be changed after installation, for security reasons.

The password is only used by the server itself, so different values can be selected for each server. A file called passwd.properties is stored on the ITLM Administration server and on each ITLM Runtime server. This file is used to store encrypted passwords that the servers must use during inter-component communications.

The passwd.properties file must never be edited manually. The IBM Tivoli License Manager server command line interfaces provide commands that enable you to change the value stored in the passwords.

5.2.3 Administrator’s authentication options

Another aspect of security in regards to physical design architectural decisions is to determine the infrastructure to be used for authenticating IBM Tivoli License Manager Administrators accessing the ITLM Administration server Web user interface.

IBM Tivoli License Manager provides two methods of authenticating the user credentials provided by the IBM Tivoli License Manager Administrator at login time:

- User ID credentials of IBM Tivoli License Manager Administrators are stored in the ITLM Administration server’s database. This is the default authentication method.
- User ID credentials of IBM Tivoli License Manager Administrators are stored in Lightweight Directory Access Protocol (LDAP).

Using the LDAP authentication method is recommended in case an existing LDAP infrastructure is already in place or in the plans. However, all User ID credentials defined in the LDAP structure must also have corresponding definitions in the ITLM Administration server database. This is due to the fact that, although authentication is handled by LDAP, IBM Tivoli License Manager is still responsible for checking for the IBM Tivoli License Manager Administrator’s authorization role.

Regardless of the authentication method of choice, an IT process must be defined, or changed, to include the method for defining IBM Tivoli License Manager Administrators. If LDAP is to be used, the IT process must include definition on both the LDAP and ITLM Administration servers.

5.2.4 Proxy server

A proxy server is a server that acts as an intermediary between a workstation requester and the Internet or intranet so that the enterprise can ensure security, administrative control, and caching service. It is associated with or part of a gateway server that separates the enterprise network from the outside network or from other enterprise networks and a firewall server that protects the enterprise network from outside intrusion.

A proxy server can be used either to filter the request from a ITLM Runtime server to the ITLM Administration server or the request from an ITLM Agent to its ITLM Runtime server. The proxy server receives a request for an Internet service from a ITLM Runtime server or an ITLM Agent and if the request passes filtering requirements, the proxy server, acting as a client on behalf of the requester, uses one of its own IP addresses to pass the information to the ITLM Administration server or ITLM Runtime server. When the information is returned, the Proxy server relates it to the original request and forwards it on to the ITLM Runtime server or the ITLM Agent.

IBM Tivoli License Manager allows the use of any proxy server for this enhanced security communications

5.2.5 Firewall communications

A firewall environment is basically a security solution operating between one or more secure, internal private networks and other (non-secure) networks or the Internet. The main objective of a firewall is to prevent unwanted or unauthorized communication into or out of the secure network. The concept of firewalls started with this basic objective, but has extended its usage and functionality to the changing needs of this corporate world.

A firewall environment is composed of an important number of tools and components. This section only refers to the main components that have or could have an impact on the IBM Tivoli License Manager solution.

Packet filters are tools that inspect the information coming in and going out of the network on a packet by packet basis. Packet filters inspect packets at the session level based upon multiple criteria, such as time of day, source/destination IP address and port number, packet type, and subnet.

Packet filtering provides the basic protection mechanism for the firewall. It allows you to determine what kind of traffic can pass across the firewall based on IP session details, thereby protecting the secure network from external threats, such as scanning for secure servers or IP address spoofing. Packet filters act as the base on which the other higher layer firewall tools can be constructed.

Some firewalls in the market do implement packet filtering based on state of the session and hence the packet filters being stateful. Every time the connection is established/attempted, the firewall maintains the session details and the state of connection for that session and packet filtering happens, depending on the state diagram for that particular protocol (that is, deciding on what kinds of packets to allow, depending on the state diagram, unlike the stateless packet filters, which just allow any permit rule match). But these kinds of stateful packet filters are more prone to attacks, as compared with stateless packet filters.

In addition to firewalls implemented in the enterprise's network (corporate firewalls), personal firewalls are also widely implemented to protect enterprise's individual nodes, servers, and workstations. As communications between ITLM Agents and their respective ITLM Runtime server use HTTP (or HTTPS) communications, personal firewalls must be configured using the same approach as the corporate firewalls.

Table 5-5 summarizes the communication protocol and port between the different physical components of the IBM Tivoli License Manager solution. It provides default ports information.

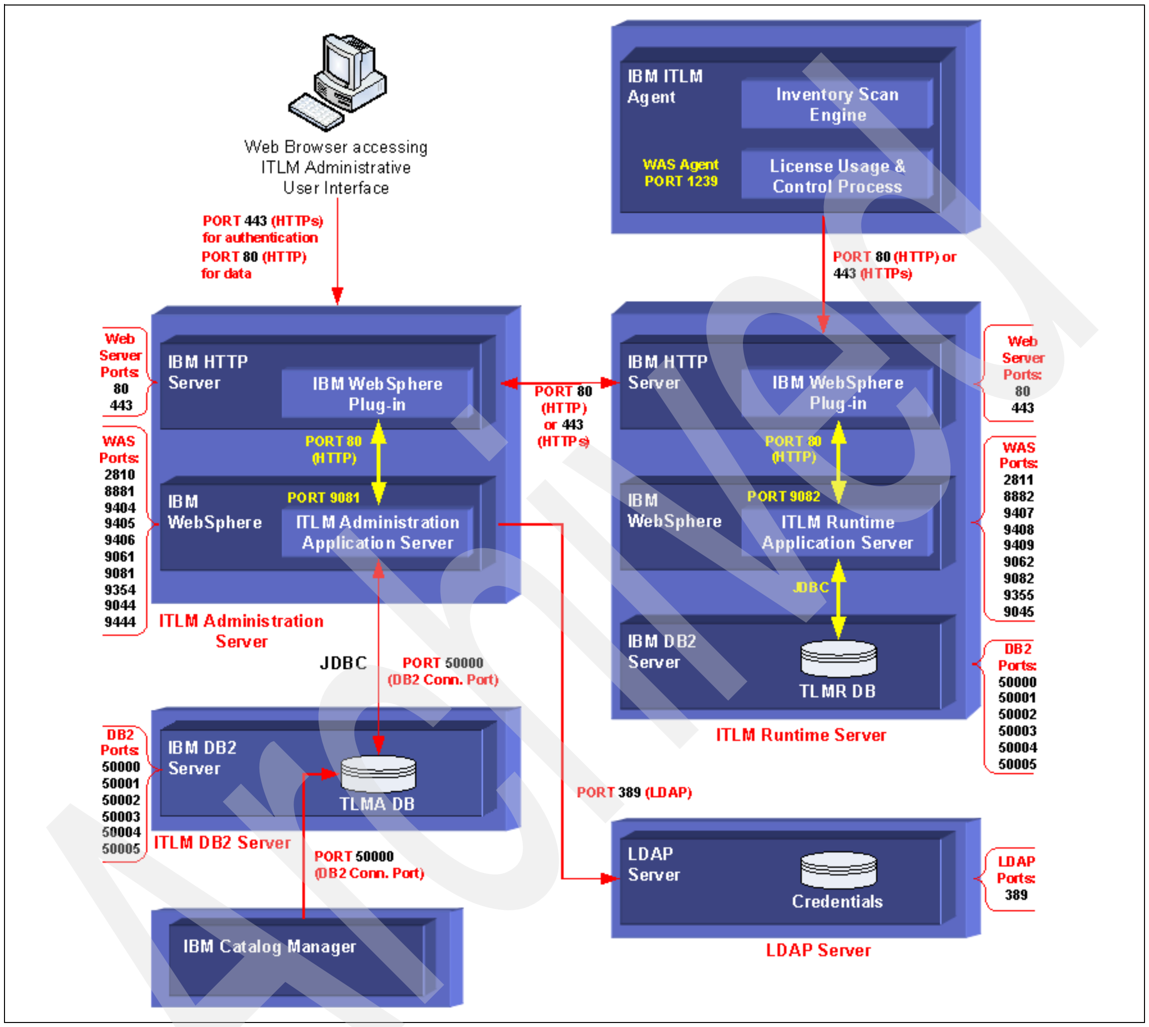
Table 5-5 Inter components communication ports

Component	Component	Protocol	Port
ITLM Administration Application server	IBM WebSphere Application Server	TCP	9081
ITLM Runtime Application server	IBM WebSphere Application Server	TCP	9082
IBM WebSphere Application Server	IBM DB2 Server hosting the ITLM Administration server database	JDBC	50000
IBM WebSphere Application Server	IBM DB2 Server hosting the ITLM Runtime server database	JDBC	50000
ITLM Administration server	ITLM Runtime server	HTTP(S)	80 (443)
ITLM Runtime server	ITLM Agent	HTTP(S)	80 (443)
ITLM Administration server	LDAP Server	LDAP	389

Component	Component	Protocol	Port
Web Browser (data) - ITLM Administrative User Interface	ITLM Administration server	HTTP	80
Web Browser (passwords) - ITLM Administrative User Interface	ITLM Administration server	HTTPS	443

The IBM DB2 Instances hosting the IBM Tivoli License Manager databases uses the 50000 TCP/IP port. In case the IBM DB2 Server hosting the IBM Tivoli License Manager databases is installed on a separate machine, IBM Tivoli License Manager V2.2 does not require the installation of IBM DB2 UDB Client on the IBM Tivoli License Manager servers. It makes use of JDBC implemented by the com.ibm.db2.jcc.DB2ConnectionPoolDataSource class.

Figure 5-4 depicts the communication between the different components of the IBM Tivoli License Manager solution, including default protocols and ports configured.



Web Browser accessing ITLM Administrative User Interface
PORT 443 (HTTPs) for authentication
PORT 80 (HTTP) for data
IBM ITLM Agent
Inventory Scan Engine
WAS Agent PORT 1239
License Usage & Control Process
PORT 80 (HTTP) or 443 (HTTPs)
Web Server Ports: 80 443
WAS Ports: 2810 8881 9404 9405 9406 9061 9081 9354 9044 9444
IBM HTTP Server
IBM WebSphere Plug-in
PORT 80 (HTTP)
IBM WebSphere
PORT 9081
ITLM Administration Application Server
ITLM Administration Server
PORT 80 (HTTP) or 443 (HTTPs)
IBM HTTP Server
IBM WebSphere Plug-in
PORT 80 (HTTP)
IBM WebSphere
PORT 9082
ITLM Runtime Application Server
JDBC
IBM DB2 Server
TLMR DB
ITLM Runtime Server
Web Server Ports: 80 443
WAS Ports: 2811 8882 9407 9408 9409 9062 9082 9355 9045
DB2 Ports: 50000 50001 50002 50003 50004 50005
JDBC
PORT 50000 (DB2 Conn. Port)
DB2 Ports: 50000 50001 50002 50003 50004 50005
IBM DB2 Server
TLMA DB
ITLM DB2 Server
PORT 50000 (DB2 Conn. Port)
IBM Catalog Manager
PORT 389 (LDAP)
LDAP Server
Credentials
LDAP Server
LDAP Ports: 389

Figure 5-4 Components communication ports and protocols

5.3 Communication flow between components

A thorough understanding of the communication flows between the various physical components of an IBM Tivoli License Manager solution is essential to produce a high quality physical design, as the way these components communicate and exchange data have an impact on the overall design of the solution.

This section provides information regarding the data management and communication flow of an IBM Tivoli License Manager solution. The following topics are covered:

5.3.1 ITLM Administration server Web user interface data

Figure 5-5 depicts the flow of data when an ITLM Administrator logs in to the ITLM Administration server Web user interface.

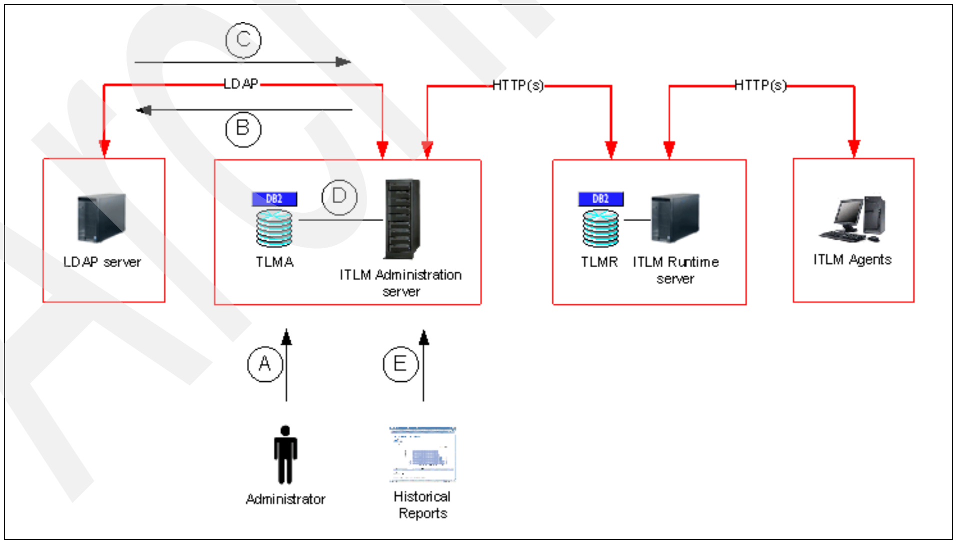


Figure 5-5 Web user interface data flow

Where:

A	ITLM Administration server Administrator login request.
B	The ITLM Administration server perform authentication based on data in the ITLM Administration server database, or optionally, forwards the authentication request to the LDAP server.
C	The LDAP server sends an authentication response to the ITLM Administration server (optional).
D	The ITLM Administrator server verifies the role of the ITLM Administrator and configures the ITLM Web user interface based on the authorization defined for the ITLM Administrator.
E	The ITLM Administrator performs related tasks, such as procurement and licenses definitions and extract reports

The following types of data are transferred:

- User ID and password (encrypted using SSL communication)
- ITLM Topology information and configuration
- Contract and Procured License information (only information that is needed to define license compliance).
- Software inventory and Usage information for the ITLM Reports

5.3.2 Configuration data

Figure 5-6 on page 113 depicts the flow of the IBM Tivoli License Manager configuration.

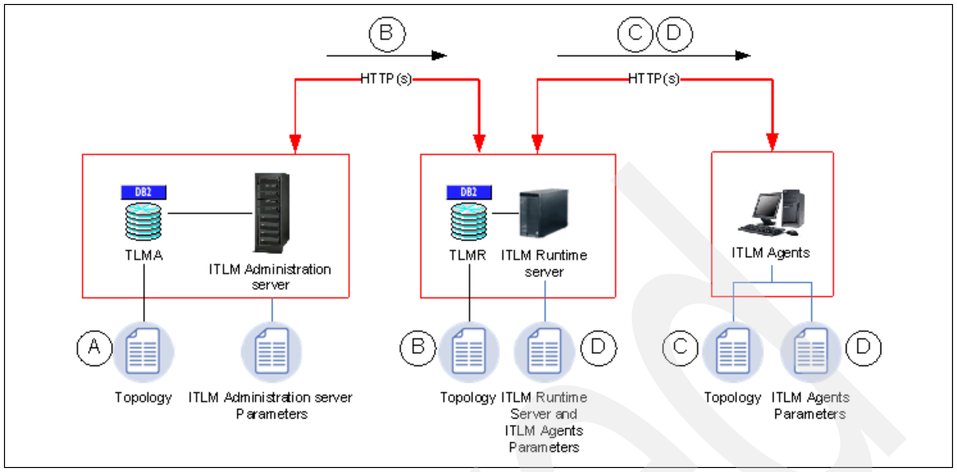


Figure 5-6 Configuration data flow

Where:

A	The ITLM Administration Server maintains the topology information.
B	The ITLM Topology information is periodically downloaded to the ITLM Runtime server. This is done at ITLM Runtime server's request based on the configuration of the system.properties file, adminDownloadPeriod parameter at the ITLM Runtime server level.
C	A copy of the ITLM Topology information is downloaded to the ITLM Agent. It is the ITLM Agent that requests an update based on the frequency defined in the system.properties file downloadParametersPeriod parameter at the ITLM Runtime server level.
D	Specific ITLM Agents parameters are defined in the system.properties file on the ITLM Runtime server. If this information is updated, the updates are sent to the ITLM Agent. It is the ITLM Agent that initiates a request for an update based on the frequency defined in the downloadParametersPeriod parameter.

5.3.3 IBM Tivoli License Manager Software Catalog data

Figure 5-7 depicts the flow of data for the IBM Tivoli License Manager Master Catalog information.

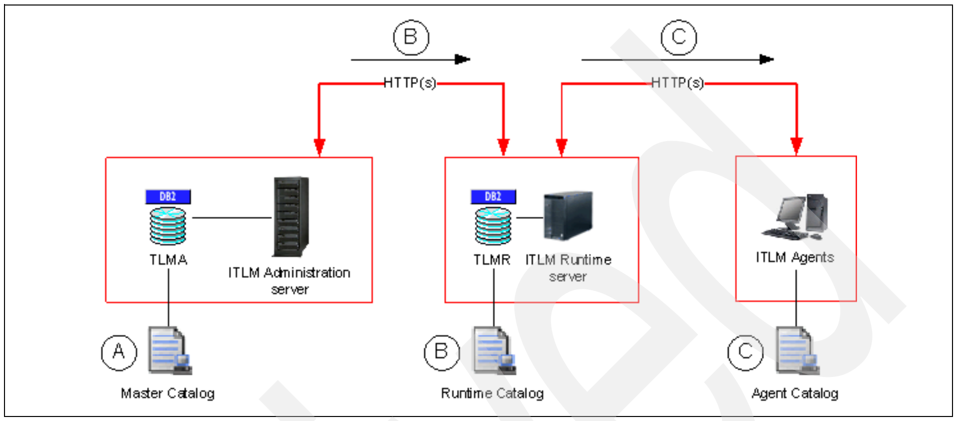


Figure 5-7 ITLM Catalog data flow

Where:

A	The ITLM Administration server maintains a Master Catalog to store details of products to be monitored.
B	A copy is periodically downloaded to the ITLM Runtime server; it is called the Runtime Catalog. This is done at the ITLM Runtime server's request based on the configuration of the system.properties file adminDownloadPeriod parameter at the ITLM Runtime server level.
C	A subset of the Runtime catalog that contains only software information related to the ITLM Agent platform; it is called the Agent Catalog and is downloaded to the ITLM Agent on a regular basis. It is the ITLM Agent that requests an update based on the frequency defined in the system.properties file downloadParametersPeriod parameter at the ITLM Runtime server level.

5.3.4 Software inventory data

Figure 5-8 on page 115 depicts that flow of software inventory data.

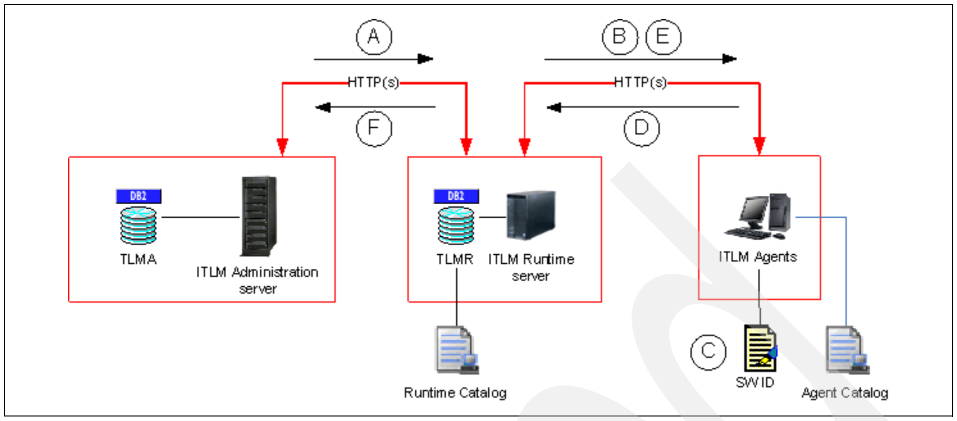


Figure 5-8 Software inventory data flow

Where:

A	Software Inventory scan is scheduled on the ITLM Administration server and the schedule automatically flows to the ITLM Runtime server(s) when the ITLM Runtime server(s) requests a configuration update. This is done at the ITLM Runtime server's request based on the configuration of the system.properties file adminDownloadPeriod parameter at the ITLM Runtime server level.
B	The ITLM Runtime server sends the inventory scan schedule to the ITLM Agents when the ITLM Agents requests a configuration update. It is the ITLM Agent that requests an update based on the frequency defined in the system.properties file downloadParametersPeriod parameter at the ITLM Runtime server level.
C	ITLM Agents perform scan schedule and identify products using a subset of the ITLM Master Catalog (the Agent Catalog).
D	Each ITLM Agent sends the list of identified products back to the ITLM Runtime server based on the frequency defined for uploading service data defined by the offlineUsagePeriod parameter at the ITLM Runtime server level.
E	The ITLM Runtime server acknowledges delivery of software inventory information to the ITLM Agent.

F	The ITLM Runtime server sends validated inventory data to the ITLM Administration server at a predefined interval configured in its system.properties file adminUploadPeriod parameter.

Note that the software inventory can also be started by the ITLM Agent through the command-line interface. In this case, items A and B do not occur.

5.3.5 Software usage data

Figure 5-9 depicts that flow of software usage data.

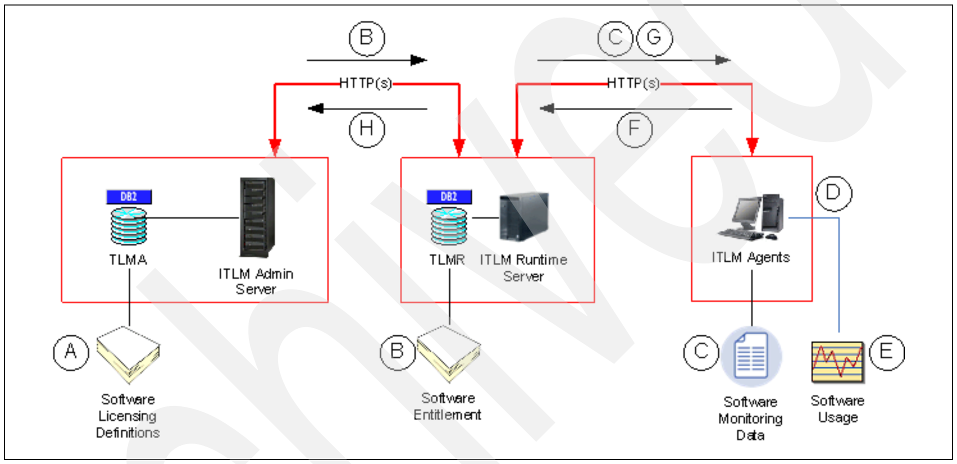


Figure 5-9 Software usage data flow

Where:

A	Software license data is defined on the ITLM Administration server.
B	All the software license entitlement information automatically flows to ITLM Runtime server(s). This depends on the configuration of the systems.properties file adminDownloadPeriod parameter at the ITLM Runtime server level.
C	Based on the software license entitlements definition, software monitoring data is delivered to the ITLM Agents at the time ITLM Agents requests a configuration update. It is the ITLM Agent that requests an update based on the frequency defined in the system.properties file downloadParametersPeriod parameter at the ITLM Runtime server level.

D	Execution of a software product is intercepted by the ITLM Agent. The product executable is only recognized as a known product if an entry in the Agent Catalog is found or if the software product has been identified previously by a software inventory scan. Otherwise, a new entry in the potential signatures file is created.
E	The ITLM Agent records software usage information based on received software monitoring data.
F	The ITLM Agent periodically uploads software usage data to its ITLM Runtime server.
G	The ITLM Runtime server acknowledges delivery of software usage information to ITLM Agent.
H	The ITLM Runtime server adds license info and uploads use data to the ITLM Administration server based on the value of the adminUploadPeriod parameter defined in the system.properties file in the ITLM Runtime server.

5.3.6 Potential signatures data

Figure 5-10 depicts the flow of software usage data.

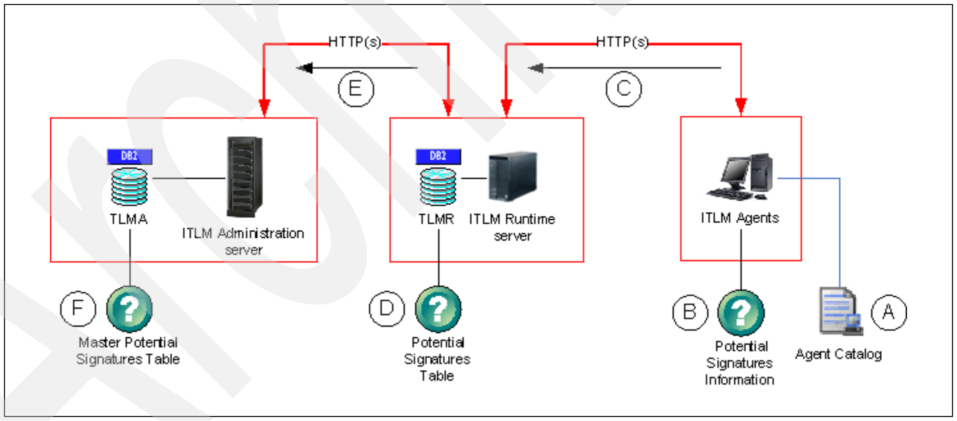

HTTP(s)
HTTP(s)
E
C
DB2
TLMA
ITLM Administration server
DB2
TLMR
ITLM Runtime server
ITLM Agents
F
Master Potential Signatures Table
D
Potential Signatures Table
B
Potential Signatures Information
Agent Catalog
A

Figure 5-10 Potential software signatures data flow

Where:

A	ITLM Agent identifies product usage and checks against the Agent Catalog if the software is a known software. Software signatures not included in the Agent Catalog go to the potential signatures information file.

B	At scheduled intervals, based on the configuration in the ITLM Runtime server' system.properties file uploadMinTime parameter, the ITLM Agent sends potential signatures information to the ITLM Runtime server.
C	The ITLM Runtime server compiles information from all ITLM Agents and consolidates it into the UNKNOWN table for potential signatures in the ITLM Runtime server database.
D	At scheduled intervals, the ITLM Runtime server sends potential signatures table contents to the ITLM Administration server according to the filtering options defined in the ITLM Runtime server file unknownFiles.properties.
E	The ITLM Administration server consolidates information from all ITLM Runtime servers and creates a master potential signatures collection and stores it into the UNKNOWN table of the ITLM Administration server database. The ITLM Runtime server is the one that initiates the upload of this information based on the adminUploadPeriod parameter in its system.properties file.

5.3.7 Complete data flow overview

Figure 5-11 on page 119 depicts a complete overview of the communication data flows for all IBM Tivoli License Manager physical components.

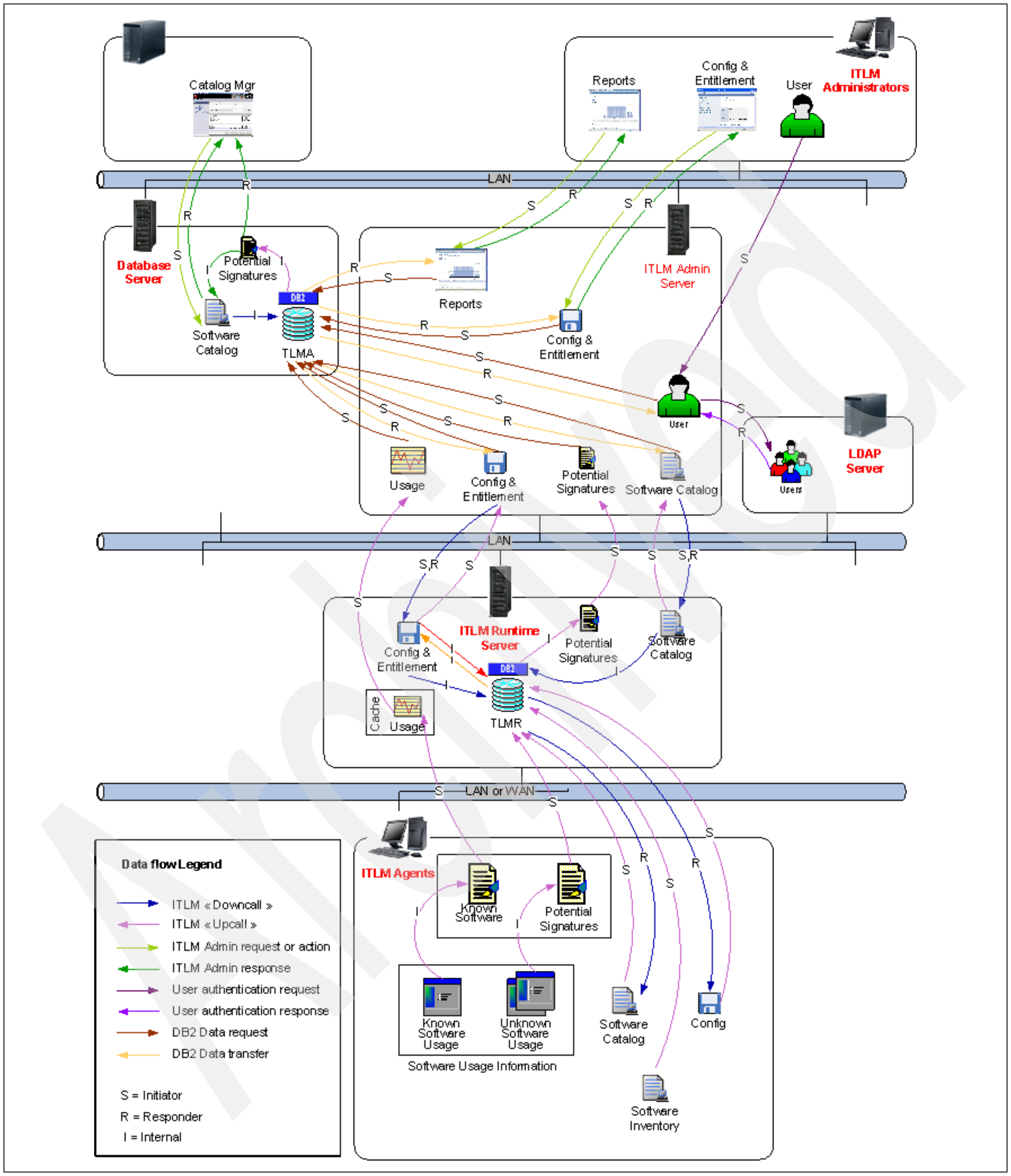


Figure 5-11 Overall IBM Tivoli License Manager communication flow

5.4 Operating System considerations

This section provides considerations for Operating Systems running on the IBM Tivoli License Manager servers.

While hardware configurations may differ among ITLM Administration servers, ITLM Runtime servers, and database servers, Operating System settings must be, as much as possible, standardized. This assures a consistent and performing environment is in place and facilitates change, support, and problem determination activities.

It is recommended that the Operating System configuration of each server in the IBM Tivoli License Manager environment be documented and maintained. In addition to that, any required changes in terms of Operating System settings must be performed in every server with the same role, for example, a security patch applied simultaneously to every ITLM Runtime server running the Windows Operating System.

The following topics are discussed in this section:

- 5.4.1, “Supported Operating Systems” on page 120
- 5.4.2, “General performance considerations” on page 121
- 5.4.3, “File system and directory structure” on page 124

5.4.1 Supported Operating Systems

The supported Operating Systems for IBM Tivoli License Manager servers include:

- Microsoft® Windows Server 2003 Standard or Enterprise Edition, 2000 Advanced Server, and 2000 Server
- IBM AIX 5L™ Version 5.3 (32-bit and 64-bit) and Version 5.2 (32-bit and 64-bit)
- HP/UX 11i
- Red Hat Enterprise Linux Version 4.0 Enterprise Server or Advanced Server Version 3.0 Enterprise Server or Advanced Server
- SUSE LINUX Enterprise Server Versions 9 and 8
- Sun™ Solaris Version 10 and 9

The supported Operating Systems for ITLM Agents include:

- Microsoft Windows Server 2003 Standard or Enterprise Edition
- Microsoft Windows 2000 Server, Advanced Server, and Professional
- Microsoft Windows XP Professional

- IBM AIX 5L Version 5.3 (32-bit and 64-bit), and Version 5.2 (32-bit and 64-bit)
- HP/UX 11i
- Red Hat Enterprise Linux Version 4.0 and Enterprise Server, Advanced Server, or Workstation
- Red Hat Enterprise Linux Version 3.0 and Enterprise Server, Advanced Server, or Workstation
- SUSE LINUX Enterprise Server Versions 9 and 8
- Sun Solaris Version 10 and 9
- OS/400 V5R2 and i5/OS® V5R3 (i5/OS is formerly known as OS/400.)

The ITLM Agent supports the following partitioning technologies:

- LPAR, DPAR, and Virtualization Engine™ on IBM @server® pSeries
- VMWare ESX Server 2.5, GSX Server 3.1, and Microsoft Virtual Server 2005 on Intel x86
- nPAR and vPAR on HP
- Dynamic System Domains and Containers on Sun

Note: The list provided above represents the information obtained at the time of the writing of this redbook. For an up to date list of supported platforms, refer to *IBM Tivoli License Manager v2.2 Planning, Installation, and Configuration*, SC32-1431 manual.

Based on deployments of previous versions of IBM Tivoli License Manager on large environments, UNIX Operating Systems have been used more often for deploying the ITLM Administration server and database server hosting the ITLM Administration server database, while ITLM Runtime servers have been deployed using the Microsoft Windows platform more often.

The choice of operating systems depends, among other things, on reliability, security, performance, or even the current platform of choice already adopted by the enterprise.

5.4.2 General performance considerations

The following system components usually affect overall system performance:

- CPU
- Memory
- Disk I/O
- Network

Those are the components that can introduce a performance bottleneck and are therefore specifically tailored to a given workload. Many of the IBM Tivoli License Manager V2.2 supported Operating Systems have sophisticated, built-in functions to optimize resource usage. On IBM AIX, for example, a standard system installation should be able to handle most IBM Tivoli License Manager workloads.

A general initial piece of advice is not to change any performance related Operating System parameters unless the workload of the IBM Tivoli License Manager environment is well known. In an environment where a single application dominates the system and is well known, initial modifications are usually done on a best practices approach.

For an IBM Tivoli License Manager environment, the performance of the database server hosting the ITLM Administration server database is crucial. This section describes Operating Systems tuning recommendations with this fact in mind. These recommendations are best practices values and have been proven to be the best choice in many different installations. Nevertheless, site specific recommendations or rules can have a higher precedence.

CPU

Depending on the Operating System of choice, there are a few parameters to adjust the Operating System to the hardware platform.

The kernel of UNIX Operating Systems is self tunable to a certain degree. We recommend only changing very basic system parameters and leave other more advanced parameters as the default values. As a general recommendation, enable the 64-bit version of the selected Operating System whenever possible.

Memory

System memory, whether paging space or real memory, must carefully be managed to make the best use out of available memory resources. We again recommend that no changes are made until experience with the actual workload exists. Many Operating Systems provide a memory manager tool to accomplish that. IBM AIX, for example, uses the Virtual Memory Manager (VMM) to control real memory and paging space utilization on a system.

Operating Systems use file system buffers and memory mapped files to avoid extensive disk I/O. The Operating System could read and write data directly to and from the disk for each request, but the response time and throughput would be poor due to slow disk access times. The Operating System therefore attempts to minimize the frequency of disk accesses by buffering data in main memory, within a structure called the *file system buffer cache*.

This implementation is designed for applications that make general I/O in file system structures. Typical RDBMS, such as IBM DB2 UDB, implement their own buffering mechanism and do not take advantage of file system buffers.

In case the IBM Tivoli License Manager environment uses a dedicated system running the database server hosting the ITLM Administration server database (ITLM Administration server and database server running on different systems, and database server dedicated to IBM DB2 UDB), we highly recommend limiting or even eliminating file system buffering and reducing the amount of memory that is occupied by mapped files.

As a general recommendation, the virtual memory size or swap size must be set to double the physical RAM, up to 2 GB of RAM, and to one and half of the physical RAM, above 2 GB of RAM, to avoid a machine crash in case of RAM shortage.

Disk I/O

Tuning the I/O subsystems is the most effective but at the same time the most challenging approach to get the best results out of the available resources. Because of the large number of subsystems that cooperate with each other at quite different speeds and capacities the best setting depends a lot on the current system layout and workload. Some advice exists for the hardware configuration and the way applications should perform I/Os. We recommend only changing the system configuration after experience with the actual workload exists with clear evidence of performance problems.

Most modern disk subsystems deliver high throughput, perform extremely high I/O rates per second, and hide the underlying disk organization completely to the end user. The Operating System itself is only presented as a logical unit (LU) of some specific size. The LUs are mostly organized into a RAID (Redundant Array of Independent Disks) configuration, typically RAID 5 or RAID 10 configurations. A modern storage subsystem additionally provides a non-volatile, memory-based read and write cache. Applications therefore perform I/O against the large memory buffers. Transaction oriented applications especially can experience a significant performance improvement.

IBM Tivoli License Manager V2.2 supports IBM DB2 UDB Enterprise Server Edition server Version 8.2. This version of IBM DB2 takes advantage of the ability to overlap processing and I/O (Asynchronous I/O) and has built in support for Direct I/O and Concurrent I/O.

Network

Knowledge of the network topology is necessary to understand which of the network options must be changed to avoid bottlenecks on the communications path to clients or other servers. This is, most of the time, very complex and difficult and can only be accomplished with a full understanding of the way the various components of the IBM Tivoli License Manager solution communicate.

The general approach is again to implement only changes to the network options at the Operating System level once specific problems are determined. In multi-user systems, changes are then usually implemented only on interface specific network options.

One of the most critical performance factors on IBM Tivoli License Manager environments of any size is the communication between the ITLM Administration server or ITLM Runtime server and their respective database servers. If the database server is installed on a different computer, ensure that a high-speed connection is used between the IBM Tivoli License Manager server components and its database server.

5.4.3 File system and directory structure

This section describes recommendations for file systems and directory structures for the servers of an IBM Tivoli License Manager environment.

While hardware configurations may differ among ITLM Administration servers, ITLM Runtime servers, and database servers, file system and directory structure settings must be, as much as possible, standardized. This assures a consistent and performing environment is in place and facilitates change, support, and problem determination activities.

The recommendations in the following sections are to be used as a reference only and the architectural decisions regarding file system and directory structure, including size, must be done depending on the IBM Tivoli License Manager environment size, and respect current policies in place.

Database server hosting the IBM Tivoli License Manager database

This section provides recommendations for the file system and directory structure for the database server hosting the IBM Tivoli License Manager databases on both UNIX and Microsoft Windows Operating Systems.

Even if you do not plan to install the ITLM Administration server and the database server hosting its database on the same server, it is a good practice to have these two servers with the same file system and directory structure. This decision allows for a more flexible architecture in case of a server consolidation action request or scalability issues, and facilitates server setting and deployment.

The initial size of both ITLM Administration server and ITLM Runtime server databases is dependent on the number of ITLM Agents of the IBM Tivoli License Manager environment, as follows:

- ITLM Administration server

  The initial size is 500 MB plus a quota for each ITLM Agent. The estimated quota per ITLM Agent is 400-500 KB:

  ITLMADM_DB_SIZE = 500 MB + (500 KB * AGENTS)

  Where AGENTS represents the total number of ITLM Agents on the IBM Tivoli License Manager environment.

- ITLM Runtime server

  The initial size is 200 MB plus a quota for each ITLM Agent. The estimated quota per ITLM Agent is 300-400 KB:

  ITLMRTS_DB_SIZE = 200 MB + (400 KB * AGENTS)

  Where AGENTS represents the total number of ITLM Agents managed by the ITLM Runtime server.

Table 5-6 presents recommendations for UNIX.

Table 5-6 Database server running on UNIX

Type	Name	Size	Description
File system	/usr/opt/db2_08_01	2 GB	It contains IBM DB2 Server binaries.
Directory structure	/home/db2tlm	0.5 GB	It is the home directory of the IBM DB2 Instance owner of the IBM Tivoli License Manager database.
File system	/db2tlmdata	100 GB (Admin[a]) 10 GB (Runtime[b])	It contains the IBM Tivoli License Manager database data.
File system	/db2tlmtmp	20 GB	It contains the temporary Tablespace for the IBM Tivoli License Manager database.
File system	/db2tlmlogs	2 GB	It contains IBM DB2 log file for the IBM Tivoli License Manager database.
Directory structure	/var/ibm/db2/diaglogs	200 MB	It contains error, log, trace, dump, event log, and alert log files for IBM DB2.
Directory structure	/tmp	200 MB	It contains temporary IBM Tivoli License Manager files.

a. Size estimate based on an IBM Tivoli License Manager environment with 45,000 ITLM Agents and 12 months usage information.
b. Size estimate based on 5,000 ITLM Agents being managed by the ITLM Runtime server and 12 months usage information.

Table 5-7 presents recommendations for Microsoft Windows.

Table 5-7 Database server running on Microsoft Windows

Type	Name	Size	Description
File system	\IBM\DB2\SQLLIB	1 GB	It contains IBM DB2 Server binaries.
File system	\DB2TLM	100 GB (Admin[a]) 10 GB (Runtime[b]	It contains the IBM Tivoli License Manager database data.
Directory structure	\Program Files\ibm\db2\diaglogs	200 MB	It contains error, log, trace, dump, event log, and alert log files for IBM DB2.
Directory structure	\temp	200 MB	It contains temporary IBM Tivoli License Manager files.

a. Size estimate based on an IBM Tivoli License Manager environment with 45,000 ITLM Agents and 12 months usage information.
b. Size estimate based on 5,000 ITLM Agents being managed by the ITLM Runtime server and 12 months usage information.

ITLM Administration server

This section provides recommendations for the file system and directory structure for the ITLM Administration server on both the UNIX and Microsoft Windows Operating Systems.

Note: The following tables assume that the ITLM Administration server and the database server hosting its database are installed on separate systems.

Table 5-8 presents recommendations for UNIX.

Table 5-8 ITLM Administration server running for UNIX

Type	Name	Size	Description
File system	/usr/WebSphere	1 GB	It contains the IBM WebSphere binary files.
Directory structure	/usr/WebSphere/AppServer	1 GB	It contains the IBM WebSphere Applications Servers files.
File system	/usr/ITLM	1 GB	It contains the ITLM Administration server binary files.
Directory structure	/usr/ITLM/admin/db	0.5 GB	It contains the ITLM Administration server configuration and binary files for accessing the ITLM Administration server database.
File system	/usr/IBMHTTPServer	0.5 GB	It contains the IBM HTTP Server binary files.
Directory structure	/var/ibm/tivoli/common	100 MB	The Tivoli common log and trace directory.
Directory structure	/tmp	200 MB	It contains temporary ITLM Administration server files.

Table 5-9 presents recommendations for Microsoft Windows.

Table 5-9 ITLM Administration server running on Microsoft Windows

Type	Name	Size	Description
File system	\IBM\WebSphere	1 GB	It contains the IBM WebSphere binary files.
Directory structure	\IBM\WebSphere\AppServer	1 GB	It contains the IBM WebSphere Applications Servers files.
File system	\IBM\ITLM	1 GB	It contains the ITLM Administration server binary files.
Directory structure	\IBM\ITLM\admin\db	0.5 GB	It contains the ITLM Administration server configuration and binary files for accessing the ITLM Administration server database.
File system	\IBM\HTTPServer	0.5 GB	It contains the IBM HTTP Server binary files.
Directory structure	\Program Files\ibm\tivoli\common	100 MB	The Tivoli common log and trace directory.
Directory structure	\temp	200 MB	It contains temporary ITLM Administration server files.

ITLM Runtime server

This section provides recommendations for the file system and directory structure for the ITLM Runtime server on both UNIX and Microsoft Windows Operating Systems.

Note: The following tables assume that the ITLM Runtime server and the database server hosting its database are installed on separate systems.

Table 5-10 presents recommendations for UNIX.

Table 5-10 ITLM Runtime server running for UNIX

Type	Name	Size	Description
File system	/usr/WebSphere	1 GB	It contains the IBM WebSphere binary files.
Directory structure	/usr/WebSphere/AppServer	1 GB	It contains the IBM WebSphere Applications Servers files.
File system	/usr/ITLM	1 GB	It contains the ITLM Runtime server binary files.
Directory structure	/usr/ITLM/runtime/db	0.5 GB	It contains the ITLM Runtime server configuration and binary files for accessing the ITLM Runtime server database.
File system	/usr/IBMHTTPServer	0.5 GB	It contains the IBM HTTP Server binary files.
Directory structure	/var/ibm/tivoli/common	100 MB	The Tivoli common log and trace directory.
Directory structure	/tmp	200 MB	It contains temporary ITLM Runtime server files.

Table 5-11 presents recommendations for Microsoft Windows.

Table 5-11 ITLM Runtime server running for Microsoft Windows

Type	Name	Size	Description
File system	\IBM\WebSphere	1 GB	It contains the IBM WebSphere binary files.
Directory structure	\IBM\WebSphere\AppServer	1 GB	It contains the IBM WebSphere Applications Servers files.
File system	\IBM\ITLM	1 GB	It contains the ITLM Runtime server binary files.
Directory structure	\IBM\ITLM\runtime\db	0.5 GB	It contains the ITLM Runtime server configuration and binary files for accessing the ITLM Runtime server database.
File system	\IBM\HTTPServer	0.5 GB	It contains the IBM HTTP Server binary files.
Directory structure	\Program Files\ibm\tivoli\common	100 MB	The Tivoli common log and trace directory.
Directory structure	\temp	200 MB	It contains temporary ITLM Runtime server files.

5.5 Database servers

The ITLM Administration server, ITLM Runtime server, and ITLM Catalog Manager applications rely on a database infrastructure for their functions. IBM Tivoli License Manager uses IBM DB2 Universal Database™ Enterprise Edition as a prerequisite for its database infrastructure. This prerequisite application is supplied with restricted-use licenses that allow them to be used only in association with the use of IBM Tivoli License Manager. The licenses also authorize you to download, install, and use any fixes issued on these products. By accepting the IBM Tivoli License Manager license agreement during the install process, the license agreements for the prerequisite software is also implicitly accepted, including the restricted use. This license does not include data manipulation or reporting by programs not provided as part of the IBM Tivoli License Manager solution.

As mentioned in 5.1, “Physical components considerations” on page 91, IBM Tivoli License Manager V2.2 supports IBM DB2 UDB Enterprise Edition Server Version 8.2 running on any platform supported by IBM DB2.

As far as access to the database server, IBM Tivoli License Manager V2.2 no longer requires IBM DB2 Client as a prerequisite for ITLM Administration server and ITLM Runtime server applications, as well as for the ITLM Catalog Manager.

Both ITLM Administration server and ITLM Runtime server applications access their databases using JDBC providers defined for each of their respective application servers on IBM WebSphere. This JDBC provider uses classes defined in the db2jcc.jar file, a JDBC Driver of type 4 Version 2.3.63, which is shipped with IBM DB2 UDB Enterprise Edition Server Version 8.2. The same database driver is used by the ITLM Catalog Manager application.

Tip: You can obtain the version of the JDBC Driver on IBM DB2 UDB Enterprise Edition Server Version 8.2 using the following command:

```
java -cp db2jcc.jar com.ibm.db2.jcc.DB2Jcc -version
```

The result of the above command should be similar to:

```
IBM DB2 JDBC Universal Driver Architecture 2.3.63
```

The following topics are discussed in this section:

- 5.5.1, “IBM DB2 UDB overview and terminology” on page 132
- 5.5.2, “IBM DB2 UDB Instance planning considerations” on page 137
- 5.5.3, “IBM DB2 UDB Database planning considerations” on page 145

5.5.1 IBM DB2 UDB overview and terminology

This section provides concepts and terminology that are relevant to create a performing database infrastructure for the IBM Tivoli License Manager environment. In addition, in order to better understand the architectural decisions that are described in the sections below, it is first important to have an overview of the IBM DB2 architecture. Figure 5-12 on page 133 gives the overall structure and dependencies between the different objects.

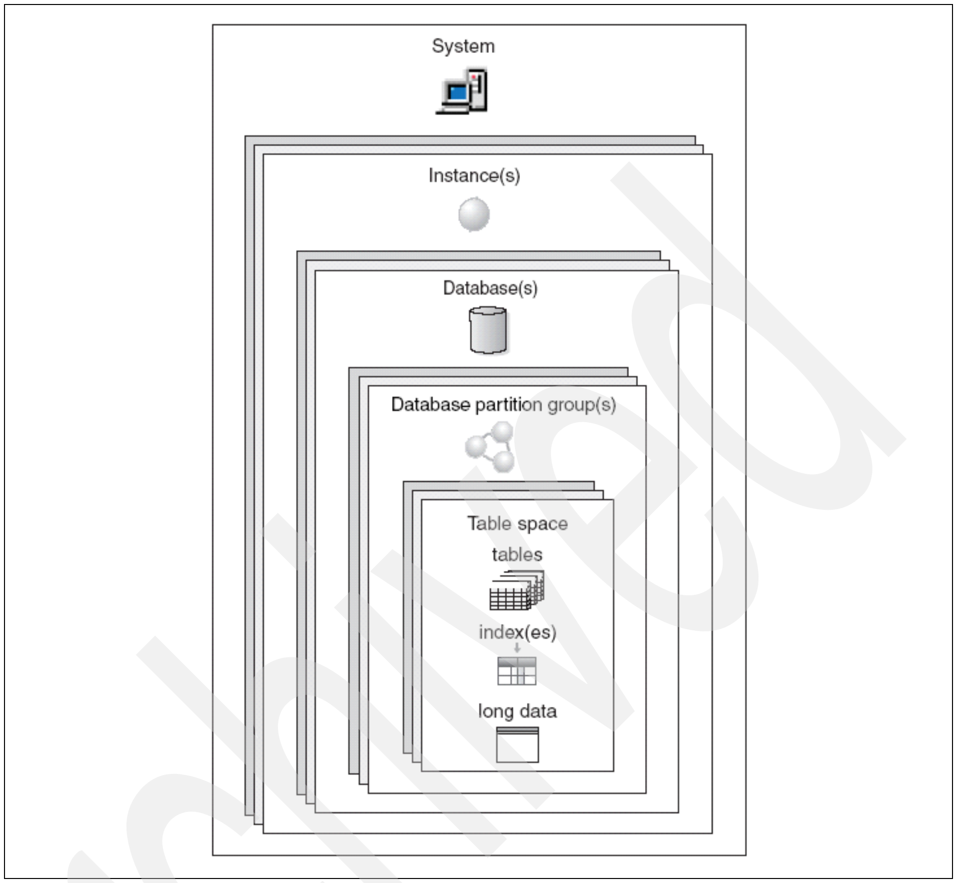


Figure 5-12 IBM DB2 architecture

As depicted in Figure 5-12, IBM DB2 has the main following objects:

- Instance

  An Instance, sometimes called a database manager, is IBM DB2 code that manages data. It controls what can be done to the data, and manages system resources assigned to it. Each Instance is a complete environment. It contains all the database partitions defined for a given parallel database system. An Instance has its own databases that other Instances cannot access, and all its database partitions share the same system directories. It also has separate security from other Instances on the same system.

- Database

  A relational database presents data as a collection of tables. A table consists of a defined number of columns and any number of rows. Each database includes a set of system catalog tables that describe the logical and physical structure of the data, a configuration file containing the parameter values allocated for the database, and a recovery log with ongoing transactions and archiveable transactions.

- Database partition group

  IBM DB2 allows a database to be partitioned. A database partition Group is a set of one or more database partitions.

- Tablespace

  A database is organized into parts called tablespaces. A tablespace is a place to store database tables. Certain objects, such as indexes and large object data, could be kept separately from the rest of the table data. A tablespace can also be spread over one or more physical storage devices. Tablespaces reside in database partition groups. Tablespace definitions and attributes are recorded in the database system catalog.

  A tablespace can be either system managed space (SMS), or database managed space (DMS). For an SMS tablespace, each container is a directory in the file space of the Operating System, and the Operating System's file manager controls the storage space. For a DMS tablespace, each container is either a fixed size pre-allocated file, or a physical device such as a disk, and the Database manager controls the storage space.

Other important IBM DB2 objects are as follows:

- Container

  A container is a physical storage device. It can be identified by a directory name, a device name, or a file name. A container is assigned to a tablespace. A single tablespace can span many containers, but each container can belong to only one tablespace. Figure 5-13 on page 135 shows the relationship between tables and a tablespace within a database, and the associated containers and disks.

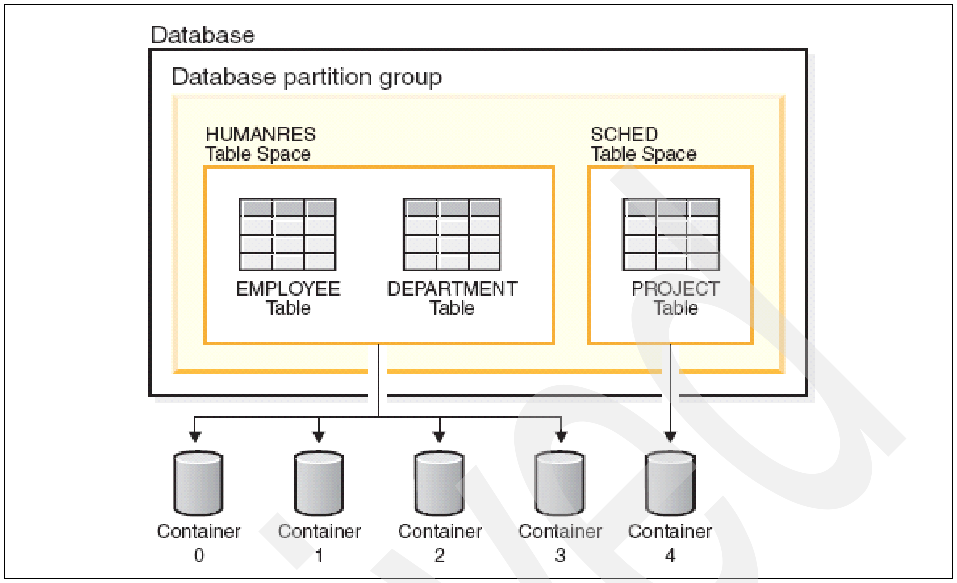


Figure 5-13 IBM DB2 containers

Data for any table will be stored on all containers in a tablespace in a round-robin fashion. This balances the data across the containers that belong to a given tablespace. The number of pages that the Instance writes to one Container before using a different one is called the extent size.

- Buffer pool

  A buffer pool is the amount of main memory allocated to cache table and index data pages as they are being read from disk, or being modified. The purpose of the buffer pool is to improve system performance. Data can be accessed much faster from memory than from disk; therefore, the fewer times the Instance needs to read from or write to a disk (I/O), the better the performance. The configuration of the buffer pool is the single most important tuning area, because you can reduce the delay caused by slow I/O. Figure 5-14 on page 136 shows the relationship between a buffer pool and containers.

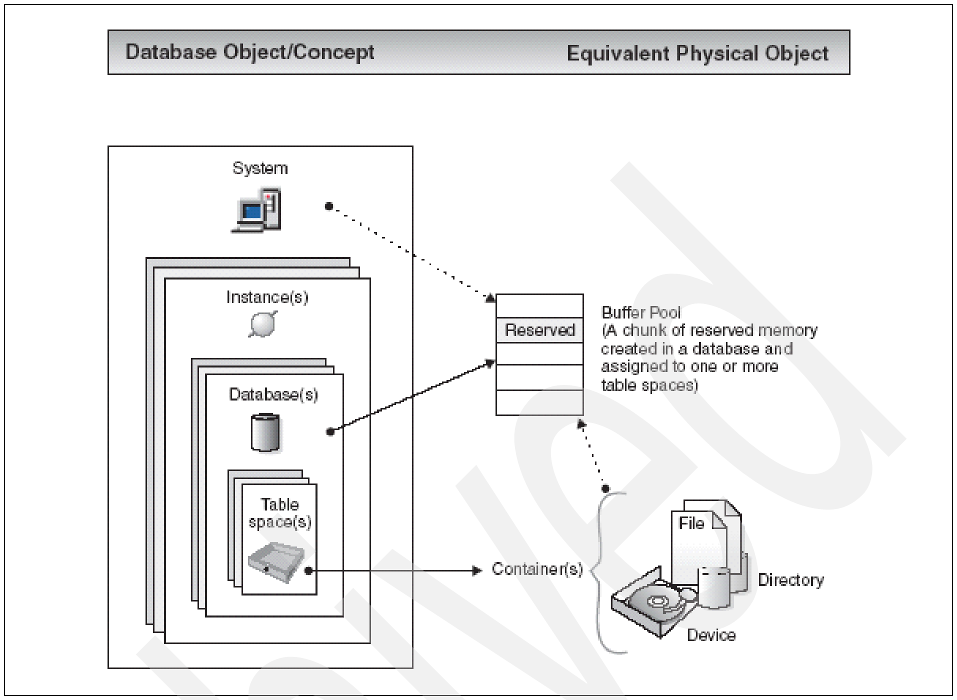

Database Object/Concept
Equivalent Physical Object
System
Instance(s)
Database(s)
Table space(s)
Reserved
Buffer Pool (A chunk of reserved memory created in a database and assigned to one or more table spaces)
Container(s)
File
Directory
Device

Figure 5-14 IBM DB2 buffer pools

- Table

  The data in the table is logically related, and relationships can be defined between tables. Data can be viewed and manipulated based on mathematical principles and operations called *relations*. Table data is accessed through Structured Query Language (SQL), a standardized language for defining and manipulating data in a relational database. A query is used in applications or by users to retrieve data from a database.

- View

  An efficient way of representing data without needing to maintain it is to use a view. A view is not an actual table and requires no permanent storage. A virtual table is created and used. A view can include all or some of the columns or rows contained in the tables on which it is based.

- System catalog tables

  Each database includes a set of system catalog tables, which describe the logical and physical structure of the data. IBM DB2 creates and maintains an extensive set of system catalog tables for each database. These tables contain information about the definitions of database objects, such as user tables, views, and indexes, as well as security information about the authority that users have on these objects.

- Schemas

  A schema is an identifier, such as a user ID, which helps group tables and other database objects. A schema can be owned by an individual, and the owner can control access to the data and the objects within it. A schema is also an object in the database. A schema name is used as the first part of a two-part object name. When an object is created, it could be assigned to a specific schema or to the default schema if none is specified. The second part of the name is the name of the object.

- Indexes

  In order to have more efficient access to rows in a table by creating a direct path to the data through pointers, indexes are created. The SQL optimizer automatically chooses the most efficient way to access data in tables. The optimizer takes indexes into consideration when determining the fastest access path to data. Unique indexes can be created to ensure uniqueness of the index key. An index key is a column or an ordered collection of columns on which an index is defined. Using a unique index will ensure that the value of each index key in the indexed column or columns is unique.

5.5.2 IBM DB2 UDB Instance planning considerations

This section provides Instance configuration settings recommendations that are relevant to creating a performing database infrastructure for the IBM Tivoli License Manager environment. This section does not present an exhaustive list of IBM DB2 configuration capabilities and only focuses on the configurations that can have an impact on the overall performance of the database environment used by IBM Tivoli License Manager.

This section contains the following topics:

- Instance directory
- Multiple Instances of the Database Manager
- Database partitioning and processor environments
- Parallelism
- Agents priority
- Application support layer heap size
- Java interpreter heap size

- Sort heap threshold

Instance directory

The Instance directory stores all information that pertains to a database Instance. The location of the Instance directory cannot be changed once it is created and used. The Instance directory contains:

- The Instance (database manager) configuration file
- The system database directory
- The node directory
- The node configuration file (db2nodes.cfg)

Multiple Instances of the database manager

Multiple Instances of the database manager may be created on a single server. This means several instances of the same product could be created on a physical machine, and have them running concurrently. This provides flexibility in setting up environments. Multiple Instances could be created for IBM Tivoli License Manager environment, for example:

- Separation of the IBM Tivoli License Manager verification and testing environment from the IBM Tivoli License Manager production environment.
- Separately tune each for the specific IBM Tivoli License Manager application servers the Instance will service.
- Protect sensitive information from administrators.

On IBM DB2 Servers running on UNIX, in order to prevent environmental conflicts between two or more Instances, each Instance should have its own home file system.

Each Instance of the same IBM DB2 server uses the same IBM DB2 program files. This means program files are not duplicated for each Instance created. Also, several related databases can be located within a single Instance.

In case the database server hosting the IBM Tivoli License Manager servers (Administration or Runtime server) database runs on a system that is not dedicated to the IBM Tivoli License Manager environment, there is a possibility that this database server already has several operational Instances. In this case, an exclusive Instance must be created to host IBM Tivoli License Manager servers' databases, in order to increase to security, and facilitate the management of the IBM Tivoli License Manager environment.

As a general rule, even if the IBM Tivoli License Manager server application and its database server run on the same physical system, we recommend that an exclusive Instance be created for the IBM Tivoli License Manager server application. This prevents the default Instance created during IBM DB2 installation from being shared with any IBM Tivoli License Manager server application.

Database partitioning and processor environments

A database partition is a portion of a database that consists of its own data, indexes, configuration files, and transaction logs. A database partition is sometimes called a node or a database node. The following are types of partitions:

- Single partition database

  A single partition database is a database having only one database partition. All data in the database is stored in that partition. In this case, database partition groups, while present, provide no additional capability.

- Multiple partition database

  A multiple partition database is a database with two or more database partitions. Tables can be located in one or more database partitions. When a Table is in a database partition group consisting of multiple partitions, some of its rows are stored in one partition, while some are stored in other partitions.

The following list describes the different possible database partitioning and processor environments:

- Single partition on a single processor (uniprocessor)
- Single partition with multiple processors (SMP)
- Multiple partition on a single processor (MPP)
- Multiple partition with multiple processors
- Logical multiple database partitions (also known as Multiple Logical Nodes (MLN)

In a partitioned database environment, most communication between database partitions is handled by the Fast Communications Manager (FCM). To enable the FCM for a Database partition and allow communication with other Database partitions, a service entry must be created in the partition's services file of the /etc directory.

Depending on the hardware configuration, IBM Tivoli License Manager environments have been deployed using a single partitioned database system. Nevertheless, in case of scalability issues, new processors or storage devices added to the IBM Tivoli License Manager servers, migration from single partition to a multiple partition database configuration could be necessary.

Parallelism

Components of a task, such as a database query, can be run in parallel to dramatically enhance performance. The nature of the task, the database configuration, and the hardware environment, all determine how IBM DB2 will perform a task in parallel. These considerations are interrelated, and should be considered together when working on the physical and logical design of a IBM Tivoli License Manager database environment. The following types of parallelism are supported by IBM DB2:

- Input/Output (I/O) parallelism

  While reading or writing data from and to tablespace containers, IBM DB2 may use parallel I/O. The degree of parallelism is determined by the prefetch size and extent size for the containers in the tablespace. Prefetch and extent will be discussed in 5.5.3, “IBM DB2 UDB Database planning considerations” on page 145.

  If the I/O parallelism is not explicitly configured via a dedicated registry variable, the degree of parallelism of any tablespace is the number of containers of the tablespace. If this registry variable is set, the degree of parallelism of the tablespace is the ratio between the prefetch size and the extent size of this tablespace.

- Query parallelism

  There are two types of query parallelism: inter-query parallelism and intra-query parallelism.

  Inter-query parallelism refers to the ability of multiple applications to query a database at the same time. Each query executes independently of the others, but IBM DB2 executes all of them at the same time.

  Intra-query parallelism refers to the simultaneous processing of parts of a single query, using either intra-partition parallelism, inter-partition parallelism, or both.

The intra-query and the utility parallelism could use different partition parallelism configurations. there are two types of partition parallelism, as follows:

- Intra partition parallelism

  This refers to the ability to break up a query into multiple parts. It subdivides what is usually considered a single database operation, such as index creation, database loading, or SQL queries, into multiple parts, many or all of which can be run in parallel within a single database partition. The main advantage is that the result is returned more quickly than if the operation were run in serial fashion. The degree of parallelism could be defined manually or by the system. The degree of parallelism represents the number of pieces of an operation running in parallel.

- Inter partition parallelism

  This refers to the ability to break up a query into multiple parts across multiple partitions of a partitioned database, on one machine, or multiple machines. The query is run in parallel. It subdivides what is usually considered a single database operation, such as index creation, database loading, or SQL queries, into multiple parts, many or all of which can be run in parallel across multiple partitions of a partitioned database on one machine or on multiple machines. The main advantage is that the results are returned more quickly than if the query were run in serial fashion on a single partition. The degree of parallelism is largely determined by the number of partitions defined and how a database partition group is defined.

- Simultaneous intra partition and inter partition parallelism

  Intra partition parallelism and inter partition parallelism could be used at the same time. This combination provides two dimensions of parallelism, resulting in an even more dramatic increase in the speed at which queries are processed.

Parallelism is often decided based on the behavior of the application accessing the data. If the application creates a lot of transactions running in parallel, using the parallelism configuration could harm the overall performance. As splitting up a query needs some CPU cycles, splitting a large number of queries could take time. If the application only generates few queries but is very complicated, for example, Data Warehouse, the parallelism is really efficient.

The IBM DB2 configuration parameter related to this subject is intra_parallel.

In terms of an IBM Tivoli License Manager environment, I/O parallelism is recommended for all databases.

At the ITLM Runtime server level, considering the normal utilization rate and number of managed ITLM Agents, there is no need for partition parallelism for improving performance. As the ITLM Administration server database is mainly dedicated to storing the historical information about software usage, reports generation is one of the main tasks done by administrators. These reports are built via a small number of complex queries. In order to improve the performance of reports generation, in the case of a single partitioned database, intra partition parallelism is recommended.

Agents priority

The priority given by the Operating System scheduler to all agents and other database manager Instance processes and threads can be controlled. In a partitioned database environment, this also includes coordinating subagents, the parallel system controllers, and the Fast Communication Manager (FCM), used for communication between different partitions' daemons. This priority determines how CPU time is given to the IBM DB2 processes, agents, and threads relative to the other processes and threads running on the system. Therefore, tuning the Agents Priority parameter allows controlling the priority with which the Database Manager processes and threads will execute on the machine.

As the configuration of this parameter has a direct impact on the attribution of the processes priority at the Operating System level, the setting of this Agents Priority parameter should be balanced with the other activity expected on the system.

When the parameter is set to -1, no special action is taken and the Database Manager is scheduled in the normal way that the Operating System schedules all processes and threads. When the parameter is set to a value other than -1, the Database Manager will create its processes and threads with a static priority set to the value of the parameter. The IBM DB2 configuration parameter related to this subject is agentpri.

In the context of a typical IBM Tivoli License Manager environment in which ITLM Administration server and its database server are installed on separate systems, and ITLM Runtime server and their databases are sharing the same hardware, the default value is used initially. This value provides a good compromise between response time to other users/applications and Database Manager throughput. If Database performance becomes a concern, specially at the ITLM Administration server level, benchmarking techniques must be used to determine the optimum setting for this parameter. Increasing the priority of the Database Manager processes and threads can have significant performance benefits. Nevertheless, even if the database server hosting the IBM Tivoli License Manager database is currently dedicated for the IBM Tivoli License Manager environment, increasing the priority of the Database Manager could severely degrade the performance of other user processes, especially when the CPU utilization is very high, and should be reviewed in case the database server becomes shared with other applications.

Application support layer heap size

The application support layer heap represents a communication buffer between the local application and its associated agent. This buffer is allocated as shared memory by each Database Manager agent that is started. If the request to the Database Manager, or its associated reply, does not fit into the buffer, they will be split into two or more send-and-receive pairs. The size of this buffer should be set to handle the majority of requests using a single send-and-receive pair.

The main recommendations to configure the application support layer heap size are if the application's requests are generally small and the application is running on a memory constrained system, we recommend reducing the value of this parameter. If the queries are generally very large, requiring more than one send and receive request, and the system is not constrained by memory, we recommend increasing the value of this parameter. The IBM DB2 configuration parameter related to this subject is aslheapsz.

The following formula can be used to calculate a minimum number of pages for aslheapsz:

aslheapsz >= (sizeof(input SQLDA) + sizeof(each input SQLVAR) + sizeof(output SQLDA) + 250) / 4096

where sizeof(x) is the size of x in bytes that calculates the number of pages of a given input or output value.

Again, for a typical IBM Tivoli License Manager environment, the default value is used initially. However, this parameter could be an important point to take into account if the IBM Tivoli License Manager servers' queries are facing some performance problems, more precisely at the ITLM Administration server level, on which reports queries could request the transfer of a large number of rows.

Java interpreter heap size

This parameter determines the maximum size of the heap that is used by the Java interpreter started to service Java DB2 stored procedures and user defined functions (UDF). There is one heap for each IBM DB2 process (one for each agent or subagent on UNIX-based platforms, and one for each instance on other platforms). There is one heap for each fenced UDF and fenced stored procedure process. There is one heap per agent (not including sub-agents) for trusted routines. In all situations, only the agents or processes that run Java UDFs or stored procedures ever allocate this memory.

In general, increasing the size of the Java heap improves throughput until the heap no longer resides in physical memory. After the heap begins swapping to disk, Java performance deteriorates drastically. Therefore, the maximum heap size needs to be low enough to contain the heap within physical memory.

The IBM DB2 configuration parameter related to this subject is java_heap_sz.

In the context of a typical IBM Tivoli License Manager environment, the default value is used initially. If the IBM Tivoli License Manager server's database performance decreases and processes start paging, before increasing the amount of memory on the machine, it will be important to tune first the Java interpreter heap parameter.

Sort heap threshold

Private and shared sorts use memory from two different memory sources. The size of the shared sort memory area is statically predetermined at the time of the first connection to a database based on the value of the sort heap threshold. The size of the private sort memory area is unrestricted. The sort heap threshold is used differently for private and shared sorts:

- For private sorts, this parameter is an instance-wide soft limit on the total amount of memory that can be consumed by private sorts at any given time. When the total private-sort memory consumption for an instance reaches this limit, the memory allocated for additional incoming private-sort requests will be considerably reduced.
- For shared sorts, this parameter is a Database-wide hard limit on the total amount of memory consumed by shared sorts at any given time. When this limit is reached, no further shared-sort memory requests will be allowed (until the total shared-sort memory consumption falls below the limit specified by sheapthres).

This parameter is very important in the sense that it is closely linked with the Sort Heap parameter of the databases managed by the Database Manager. The sort heap for the database will be discussed in 5.5.3, "IBM DB2 UDB Database planning considerations" on page 145. The IBM DB2 configuration parameter related to this subject is sheapthres.

From an IBM Tivoli License Manager perspective, this parameter should be set to a reasonable multiple of the largest sort heap parameter defined for the IBM Tivoli License Manager servers' databases. At a minimum, it should be at least three times the largest sort heap defined for any database within the instance defined for the ITLM Administration server and two times he largest sort heap defined for any database within the instance defined for the ITLM Runtime server

5.5.3 IBM DB2 UDB Database planning considerations

This section provides Instance configuration settings recommendations that are relevant to create a performing database infrastructure for the IBM Tivoli License Manager environment. This section does not present an exhaustive list of IBM DB2 configuration capabilities and only focuses on the configurations that can have an impact on the overall performance of the database environment used by IBM Tivoli License Manager.

This section contains the following topics:

- Database directory
- Tablespace
- Extent size
- Prefetching
- Buffer pool
- Database heap
- Sort heap
- Default degree
- Maximum database files open per application

Database directory

Information about the database, including default information, is stored in a directory hierarchy. In the directory specified during the database creation, a subdirectory that uses the name of the Instance is created. This subdirectory ensures that databases created in different Instances under the same directory do not use the same path.

Below the Instance-name subdirectory, a subdirectory named NODE0000 is created. This subdirectory differentiates partitions in a logically partitioned database environment. Below the node-name directory, a subdirectory named SQL00001 is created. This name of this subdirectory uses the database token and represents the database being created. SQL00001 contains objects associated with the first database created, and subsequent databases are given higher numbers: SQL00002, and so on. The directory structure appears as follows:

<Directory>/<Instance>/NODE0000/SQL00001/

The database directory contains the following files that are created as part of the database creation process:

- The files SQLBP.1 and SQLBP.2 contain buffer pool information. Each file has a duplicate copy to provide a backup.
- The files SQLSPCS.1 and SQLSPCS.2 contain Tablespace information. Each file has a duplicate copy to provide a backup.

- The SQLDBCON file contains database configuration information.
- The DB2RHIST.ASC history file and its backup DB2RHIST.BAK contain history information about backups, restores, loading of tables, reorganization of tables, altering of a tablespace, and other changes to a database.
- The DB2TSCHNG.HIS file contains a history of tablespace changes at a log-file level. For each log file, DB2TSCHG.HIS contains information that helps to identify which tablespace are affected by the log file. Tablespace recovery uses information from this file to determine which log files to process during tablespace recovery.
- The log control files, SQLOGCTL.LFH and SQLOGMIR.LFH, contain information about the active logs. Recovery processing uses information from this file to determine how far back in the logs to begin recovery. The SQLOGDIR subdirectory contains the actual log files.
- The log subdirectory should be mapped to different disks than those used for the data. A disk problem could then be restricted to the data or the logs but not both. This can provide a substantial performance benefit because the log files and database containers do not compete for movement of the same disk heads.
- The SQLINSLK file helps to ensure that a database is used by only one instance of the Database Manager.

The SQLT* subdirectories contain the default System Managed Space (SMS) tablespaces required for an operational Database. Three default tablespaces are created:

- SQLT0000.0 subdirectory contains the catalog tablespace with the system catalog tables.
- SQLT0001.0 subdirectory contains the default temporary tablespace.
- SQLT0002.0 subdirectory contains the default user data tablespace.

Each subdirectory or container has a file created in it called SQLTAG.NAM. This file marks the subdirectory as being in use so that subsequent tablespace creation does not attempt to use these subdirectories.

In addition, a file called SQL*.DAT stores information about each table that the subdirectory or container contains. For each SQL*.DAT file, there might be one or more of the following files, depending on the table type, the reorganization status of the table, or whether indexes, LOB, or LONG fields exist for the table:

- SQL*.BMP (contains block allocation information if it is an MDC table)
- SQL*.LF (contains LONG VARCHAR or LONG VARGRAPHIC data)
- SQL*.LB (contains BLOB, CLOB, or DBCLOB data)

- SQL*.LBA (contains allocation and free space information about SQL*.LB files)
- SQL*.INX (contains index table data)
- SQL*.DTR (contains temporary data for a reorganization of an SQL*.DAT file)
- SQL*.LFR (contains temporary data for a reorganization of an SQL*.LF file)
- SQL*.RLB (contains temporary data for a reorganization of an SQL*.LB file)
- SQL*.RBA (contains temporary data for a reorganization of an SQL*.LBA file)

Tablespace

A tablespace is a storage structure containing tables, indexes, large objects, and long data. Tablespaces reside in database partition groups. They allow assigning the location of database and table data directly onto containers. This can provide improved performance and more flexible configuration. A single tablespace can span several containers. It is possible for multiple containers to be created on the same physical disk (or drive). For improved performance, each container should use a different disk. A database must contain at least three tablespaces:

- One catalog tablespace, which contains the entire system catalog tables for the Database. This tablespace is called SYSCATSPACE, and it cannot be dropped. IBMCATGROUP is the default database partition group for this tablespace.
- One or more user tablespaces, which contain all user defined tables. By default, one tablespace, USERSPACE1, is created. IBMDEFAULTGROUP is the default database partition group for this tablespace.
- One or more temporary tablespaces, which contain temporary tables. Temporary tablespaces can be system temporary tablespaces or user temporary tablespaces. By default, one system temporary tablespace called TEMPSPACE1 is created. IBMTEMPGROUP is the default database partition group for this tablespace.

A table's page size is determined either by row size, or the number of columns. The maximum allowable length for a row is dependent upon the page size of the tablespace in which the table is created. Possible values for page size are 4 KB, 8 KB, 16 KB, and 32 KB. A tablespace with one page size for the base table could be defined, and a different tablespace with a different page size for long data.

There are two types of tablespace, both of which can be used in a single Database:

- System Managed Space (SMS)

  In an SMS tablespace, the operating system's file system manager allocates and manages the space where the table is stored. The storage model typically consists of many files, representing table objects, stored in the file system space. By controlling the amount of data written to each file, the Database Manager distributes the data evenly across the tablespace containers. A file is extended one page at a time as the object grows. If insert performance needs to be improved, multi page file allocation enablement could be considered. This allows the system to allocate or extend the file by more than one page at a time. Once multipage file allocation is enabled, it cannot be disabled.

  The main advantages of an SMS Tablespace are:

  – Space is not allocated by the system until it is required.

  – Creating a tablespace requires less initial work, because the containers do not need to be predefined

- Database Managed Space (DMS)

  In a DMS tablespace, the Database Manager controls the storage space. The storage model consists of a limited number of devices or files whose space is managed by DB2. The tablespace is essentially an implementation of a special purpose file system designed to best meet the needs of the Database Manager.

  The main advantages of a DMS Tablespace are:

  – The size of a tablespace can be increased by adding or extending containers. Existing data can be automatically rebalanced across the new set of containers to retain optimal I/O efficiency.

  – A table can be split across multiple tablespaces, based on the type of data being stored: Long field and LOB data, Indexes or Regular table data

  – The location of the data on the disk can be controlled, if this is allowed by the Operating System.

  – If all table data is in a single tablespace, a tablespace can be dropped and redefined with less overhead than dropping and redefining a table.

  – In general, a well-tuned set of DMS tablespaces will outperform SMS tablespaces.

In context of a typical IBM Tivoli License Manager environment, SMS tablespaces are normally used. This is mostly based on IBM DB2 best practices, which recommend defining a single SMS temporary tablespace with a page size equal to the page size used in the majority of the user tablespaces. By default the page size is set to 4 KB. For all IBM Tivoli License Manager databases, a minimum of three tablespaces is recommended: one as a temporary tablespace, one for the IBM Tivoli License Manager data, and one for the IBM Tivoli License Manager indexes. The combination of these numbers for tablespaces and page size should be adequate for typical IBM Tivoli License Manager environments and workloads.

Extent size

The extent size for a tablespace represents the number of pages of table data that will be written to a container before data will be written to the next container. When selecting an extent size, the following should be considered:

- The size and type of tables in the tablespace
- The type of access to the tables
- The minimum number of extents required

The IBM DB2 configuration parameter related to this subject is dft_extent_sz for the default size of a new Tablespace. A specific size of extent can be provided during creation time.

The default extent size defined for databases is reasonable and should be adequate for typical IBM Tivoli License Manager environments and workloads. These are defined as 64 pages of 4 KB for the ITLM Administration server and 32 pages of 4 KB for the ITLM Runtime server.

Prefetching

When considering the I/O cost of accessing data from a tablespace, the optimizer also considers the potential impact that prefetching data and index pages from disk can have on the query performance. Prefetching data and index pages can reduce the overhead and wait time associated with reading the data into the buffer pool.

Prefetching data into the buffer pools usually improves performance by reducing the number of disk accesses and retaining frequently accessed data in memory.

Prefetching pages means that one or more pages are retrieved from disk in the expectation that they will be required by an application. Prefetching index and data pages into the buffer pool can help improve performance by reducing the I/O wait time. In addition, parallel I/O enhances prefetching efficiency. There are two categories of prefetching:

- Sequential prefetch, a mechanism that reads consecutive pages into the buffer pool before the pages are required by the application.
- List prefetch, sometimes called list sequential prefetch. Prefetches a set of non-consecutive data pages efficiently.

These two methods of reading data pages are in addition to a normal read. A normal read is used when only one or a few consecutive pages are retrieved. During a normal read, one page of data is transferred.

Prefetching is important to the performance of intra-partition parallelism, which uses multiple subagents when scanning an index or a table. Such parallel scans introduce larger data-consumption rates, which require higher prefetch rates.

The cost of inadequate prefetching is higher for parallel scans than serial scans. If prefetching does not occur for a serial scan, the query runs more slowly because the agent always needs to wait for I/O. If prefetching does not occur for a parallel scan, all subagents might need to wait because one subagent is waiting for I/O.

Because of its importance, prefetching is performed more aggressively with intra-partition parallelism. The sequential detection mechanism tolerates larger gaps between adjacent pages so that the pages can be considered sequential. The width of these gaps increases with the number of subagents involved in the scan.

It is a good practice to explicitly set the prefetch size value as a multiple of the extent size value, and the number of tablespace containers. For example, if the extent size for an ITLM Administration server database is 64 pages and the tablespace has two containers, the prefetch quantity should be set to 128 pages.

Buffer pool

A buffer pool is memory used to cache table and index data pages as they are being read from disk or being modified. The buffer pool improves Database system performance by allowing data to be accessed from memory instead of from disk. Because memory access is much faster than disk access, the less often the Database Manager needs to read from or write to a disk, the better the performance. Because most data manipulation takes place in buffer pools, configuring buffer pools is the single most important tuning area. Only large objects and long field data are not manipulated in a buffer pool.

When an application accesses a row of a table for the first time, the Database Manager places the page containing that row in the buffer pool. The next time any application requests data, the Database Manager looks for it in the buffer pool. If the requested data is in the buffer pool, it can be retrieved without disk access, resulting in faster performance. Memory is allocated for the buffer pool when a database is activated or when the first application connects to the database.

To ensure that an appropriate buffer pool is available in all circumstances, IBM DB2 creates small buffer pools, one with each page size: 4 KB, 8 KB, 16 KB, and 32 KB. The size of each buffer pool is 16 pages. These buffer pools are hidden from the user. They are not present in the system catalogs or in the buffer pool system files. They cannot be used or altered directly, but IBM DB2 uses these buffer pools in the following circumstances:

- When a buffer pool of the required page size is inactive because not enough memory was available to create it after a CREATE BUFFERPOOL statement was executed with the IMMEDIATE keyword. If necessary, tablespaces are remapped to a hidden buffer pool. Performance might be drastically reduced.
- When the ordinary buffer pools cannot be brought up during a database connect. This problem is likely to have a serious cause, such as out-of-memory condition. Although IBM DB2 will be fully functional because of the hidden buffer pools, performance will degrade drastically.

Pages remain in the buffer pool until the database is shut down, or until the space occupied by a page is required for another page. The following criteria determine which page is removed to bring in another page:

- How recently the page was referenced
- The probability that the page will be referenced again by the last agent that looked at it
- The type of data on the page
- Whether the page was changed in memory but not written out to disk (Changed pages are always written to disk before being overwritten.)

In order for pages to be accessed from memory again, changed pages are not removed from the buffer pool after they are written out to disk unless the space is needed.

Because pages can be read into a buffer pool only if the tablespace page size is the same as the buffer pool page size, the page size of the tablespaces should determine the page size that are specified for buffer pools. The page size of the buffer pool cannot be altered after it is created, so a new buffer pool with a different page size must be created.

Although each database requires at least one buffer pool, several buffer pools might be created, each of a different size or with a different page size, for a single database that has tablespaces of more than one page size.

A new database has a default buffer pool called IBMDEFAULTBP with a size determined by the platform and a default page size of 4 KB. When a tablespace with a page size of 4 KB is created and not assigned to a specific buffer pool, the tablespace is assigned to the default buffer pool. The default buffer pool can be resized and its attributes changed, but it cannot be dropped.

In an IBM Tivoli License Manager environment, the configuration of the buffer pool is of high importance and is directly linked to the amount of RAM available for the database server. It must be the first item to look at when performance is an issue.

As described in “Tablespace” on page 147, a minimum of three tablespaces is recommended for a typical IBM Tivoli License Manager environment: one as a temporary tablespace, one for the IBM Tivoli License Manager data, and one for the IBM Tivoli License Manager indexes. As far as buffer pools, the recommendation is to have one buffer pool defined for each one of these tablespaces.

Page sizes for buffer pools

If a tablespace is created with a page size greater than 4 KB, such as 8 KB, 16 KB, or 32 KB, it must be assigned to a buffer pool that uses the same page size. If this buffer pool is currently not active, IBM DB2 attempts to assign the tablespace temporarily to another active buffer pool that uses the same page size, if there is one, or to one of the default special buffer pools that IBM DB2 creates when the first client connects to the Database. When the database is activated again, and the originally specified buffer pool is active, then IBM DB2 assigns the tablespace to that buffer pool.

Advantages of large buffer pools

Large buffer pools provide the following advantages:

- Enables frequently requested data pages to be kept in the buffer pool, which allows quicker access. Fewer I/O operations can reduce I/O contention, thereby providing better response time and reducing the processor resource needed for I/O operations.
- Provides the opportunity to achieve higher transaction rates with the same response time.
- Prevents I/O contention for frequently used disk storage devices, such as catalog tables and frequently referenced user tables and indexes. Sorts required by queries also benefit from reduced I/O contention on the disk storage devices that contain the temporary tablespaces.

Advantages of many buffer pools

The usage of more than one buffer pool should be considered for the following reasons:

- Temporary tablespaces can be assigned to a separate buffer pool to provide better performance for queries that require temporary storage, especially sort-intensive queries.
- If data must be accessed repeatedly and quickly by many short update-transaction applications, assignation of the tablespace that contains the data to a separate buffer pool should be considered. If this buffer pool is sized appropriately, its pages have a better chance of being found, contributing to a lower response time and a lower transaction cost.
- Some data could be isolated into separate buffer pools to favor certain applications, data, and indexes. For example, tables and indexes that are updated frequently could be managed by a buffer pool that is separate from those tables and indexes that are frequently queried but infrequently updated. This change will reduce the impact that frequent updates on the first set of tables have on frequent queries on the second set of tables.
- Smaller buffer pools could be used for the data accessed by applications that are rarely used, especially for an application that requires very random access into a very large table. In such a case, data need not be kept in the buffer pool for longer than a single query. It is better to keep a small buffer pool for this data, and free the extra memory for other uses, such as for other buffer pools.
- After separating different activities and data into separate buffer pools, good and relatively inexpensive performance diagnosis data can be produced from statistics and accounting traces.

Database heap

There is one database heap per database, and the Database Manager uses it on behalf of all applications connected to the database. It contains control block information for tables, indexes, tablespaces, and buffer pools. It also contains space for event monitor buffers, the log buffer, and temporary memory used by utilities. Therefore, the size of the heap will be dependent on a large number of variables. The control block information is kept in the heap until all applications disconnect from the database.

The minimum amount the Database Manager needs to get started is allocated at the first connection. The data area is expanded as needed up to the maximum specified by the Database heap parameter.

Nevertheless, if performance needs to be increased, the database system monitor could be used to track the highest amount of memory that was used for the Database heap, using the maximum Database heap allocated element (db_heap_top). The IBM DB2 configuration parameter related to this subject is dbheap.

In the context of a typical IBM Tivoli License Manager environment, the default value is used initially. In the IBM Tivoli License Manager server's database performance decreases, it will be important to tune the database heap parameter.

Sort heap

This parameter defines the maximum number of private memory pages to be used for private sorts, or the maximum number of shared memory pages to be used for shared sorts. If the sort is a private sort, then this parameter affects agent private memory. If the sort is a shared sort, then this parameter affects the database shared memory. Each sort has a separate sort heap that is allocated as needed by the Database Manager. This sort heap is the area where data is sorted. If directed by the optimizer, a smaller sort heap than the one specified by this parameter is allocated using information provided by the optimizer.

When working with the sort heap, the following should be considered:

- Appropriate indexes can minimize the use of the sort heap.
- Hash join buffers and dynamic bitmaps.
- Frequent large sorts are required.
- When increasing the value of this parameter, the sort heap threshold parameter in the Database Manager configuration file should maybe also be adjusted.

The sort heap size is used by the optimizer in determining access paths. Rebinding applications should be considered after changing this parameter. The IBM DB2 configuration parameter related to this subject is sortheap.

In the context of a typical IBM Tivoli License Manager environment, the default value is used initially. If the IBM Tivoli License Manager server's database performance decreases, it will be important to tune the sort heap parameter.

Default degree

This parameter specifies the value for the current degree of parallelism used for the database. The default value is 1. A value of 1 means no intra partition parallelism. A value of -1 means the optimizer determines the degree of intra partition parallelism based on the number of processors and the type of query.

The degree of intra-partition parallelism for an SQL statement is specified at statement compilation time using the current degree special register or the degree bind option. The maximum runtime degree of intra partition parallelism for an active application is specified using the **`set runtime degree`** command. The Maximum Query Degree of Parallelism configuration parameter specifies the maximum query degree of intra partition parallelism for all SQL queries.

The IBM DB2 configuration parameter related to this subject is dft_degree.

For a small IBM Tivoli License Manager environment, the default value should be used. For a large IBM Tivoli License Manager environment, intra parallelism is recommended at the ITLM Administration server level, meaning a default degree of -1.

Maximum database files open per application

This parameter specifies the maximum number of file handles that can be open for each database agent. If opening a file causes this value to be exceeded, some files in use by this agent are closed. If this parameter is too small, the overhead of opening and closing files so as not to exceed this limit will become excessive and may degrade performance.

Both SMS tablespaces and DMS tablespace file containers are treated as files in the Database Manager's interaction with the Operating System, and file handles are required. More files are generally used by SMS tablespaces compared to the number of containers used for a DMS file tablespace.

Therefore, if SMS tablespaces are used, this parameter needs a larger value compared to what a DMS file tablespaces would require. This parameter could also be used to ensure that the overall total of file handles used by the Database manager does not exceed the Operating System limit by limiting the number of handles per agent to a specific number; the actual number will vary depending on the number of agents running concurrently.

The IBM DB2 configuration parameter related to this subject is maxfilop.

In the context of a typical IBM Tivoli License Manager environment, and based on IBM DB2 implementation best practices, the default value of 64 is not enough. The recommendation is to increase the maxfilop value to 128 minimum.

5.6 HTTP and Web Application Servers

The ITLM Administration server and ITLM Runtime server applications rely on a Web server and a Web application server layer for their functions. IBM Tivoli License Manager uses both IBM HTTP Server and IBM WebSphere as prerequisite applications. These prerequisite applications are supplied with restricted-use licenses that allow them to be used only in association with the use of IBM Tivoli License Manager. The licenses also authorize you to download, install, and use any fixes issued on these products. By accepting the IBM Tivoli License Manager license agreement during the install process, the license agreements for these prerequisite applications is also implicitly accepted, including the restricted use. This license does not include data manipulation or reporting by programs not provided as part of the IBM Tivoli License Manager solution.

The following topics are described in this section:

- 5.6.1, "Supported Web application servers" on page 156
- 5.6.2, "IBM WebSphere overview and terminology" on page 157
- 5.6.3, "IBM WebSphere Application Server planning considerations" on page 164

5.6.1 Supported Web application servers

IBM Tivoli License Manager V2.2 supports the following web application servers:

- IBM WebSphere Application Server Version 6.0.x
- IBM WebSphere Application Server Version 5.1.1

IBM Tivoli License Manager only supports the Base Edition of WebSphere Application Server in stand-alone modality. Other editions, such as WebSphere Application Server Express and WebSphere Application Server Network Deployment, are not supported. IBM WebSphere Application Server Version 6.0.x is recommended.

IBM Tivoli License Manager V2.2 supports any Web server supported by the prerequisite version of IBM WebSphere Application Server. The list of Web servers include the recommended IBM HTTP Server Version 6. If IBM HTTP Server is the Web server of choice, the IBM HTTP Server WebSphere Application Server plug-in must be installed as well.

5.6.2 IBM WebSphere overview and terminology

This section provides concepts and terminology that are relevant to create a performing Web application server infrastructure for both the ITLM Administration server and ITLM Runtime server. In addition, in order to better understand the architectural decisions that are described in the sections below, it is first important to have an overview of the IBM WebSphere architecture.

The IBM WebSphere Application Server's primary role is to provide J2EE™ (Java 2 Enterprise Edition) applications with an environment that fulfills all the J2EE requirements plus some extensions that are necessary to operate and support the applications functionally. IBM WebSphere Application Server provides the necessary services to handle HTTP, Remote Method Invocation (RMI), and Internet InterORB Protocol (IIOP) traffic, and provides connections to back-end systems like databases or messaging server, and also caching, failover, and scalability capacities. Figure 5-15 outlines the architecture of IBM WebSphere Application Server.

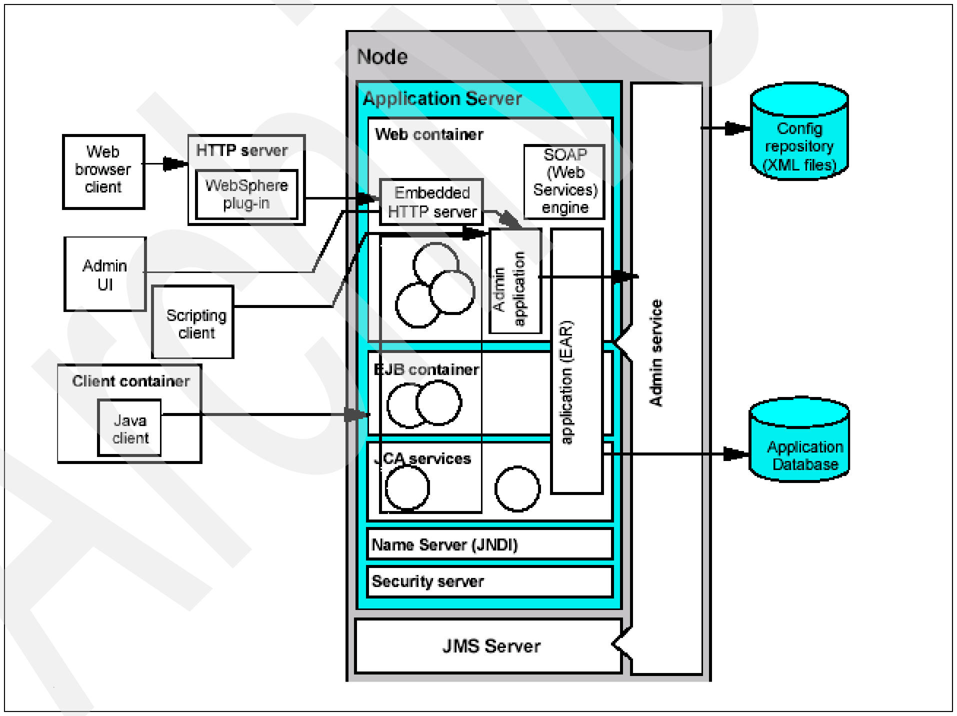


Figure 5-15 IBM WebSphere Application Server architecture

The following are important concepts, terminology and features of IBM WebSphere Application Server objects:

- Servers

  IBM WebSphere Application Server contains several servers, of which the most important one is the application server. This server provides the runtime environment for application code. It provides containers and services that specialize in enabling the execution of specific Java application components. It is the core J2EE runtime environment. Each application server runs in its own Java Virtual Machine (JVM™).

  Additional application servers can be created and federated or centrally administered by using WebSphere Application Server Network Deployment to enable scalability and failover.

  Another J2EE mandatory server is the Java Messaging Service server (JMS). The messaging support is provided through the use of one of the following JMS providers: WebSphere JMS provider (embedded), WebSphere MQ (Message Queue) JMS provider, or Generic JMS provider.

  The JMS include support for message-driven beans, point-to-point and publish/subscribe styles of messaging, and integration with the transaction management service. The JMS server runs in the same JVM as the application server. From a configuration standpoint, it is a part of the application server. The JMS server is separated from the application server and runs in a separate, dedicated JVM.

- Clustering

  WebSphere Application Server Network Deployment offers the option of using application server clustering to provide enhanced workload distribution. A cluster is a logical collection of application server processes, with the sole purpose of providing workload balancing. Application servers that belong to a cluster are members of that cluster and must all have identical application components deployed on them. Other than the applications configured to run on them, cluster members do not have to share any other configuration data.

  WebSphere Application Server Network Deployment also provides workload management. Clustering application servers that host Web containers automatically enables plug-in workload management for the application servers and the servlets they host. Routing of servlet requests occurs between the Web server plug-in and the clustered application servers using HTTP or HTTPS. Figure 5-16 on page 159 shows the routing process.

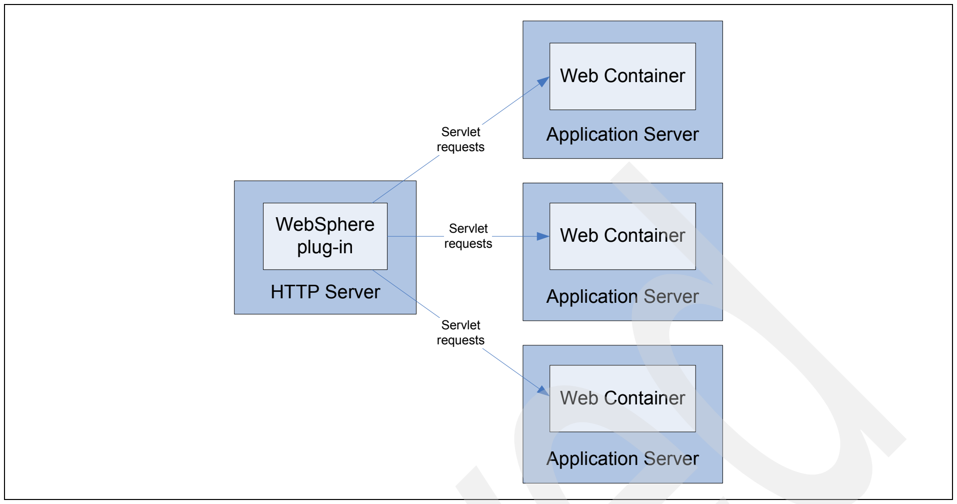


Figure 5-16 IBM WebSphere clustering

- Containers

  The J2EE specification defines the concept of containers to provide runtime support for applications:

  – Web container

    It process HTTP requests, servlets, JSPs and Web Service (SOAP). Each application server runtime has one logical Web container, which can be modified, but not created or removed.

  – EJB™ container

    It provides all the runtime services needed to deploy and manage enterprise beans. It is a server process that handles requests for both session and entity beans.

  – Client container

    In addition, there is an application client container that can run on the client machine. It allows the client to run applications in an EJB-compatible J2EE environment.

- Application Server Services

  Connection management for access to enterprise information systems (EIS) in IBM WebSphere Application Server is based on the J2EE Connector Architecture (JCA) specification, also referred to as J2C. The connection between the enterprise application and the EIS is done through the use of EIS-provided resource adapters, which are plugged into the application server. The architecture specifies the connection management, transaction management, and security contracts that exist between the application server and EIS.

  IBM WebSphere Application Server applications can use transactions to coordinate multiple updates to resources as one unit of work so that all or none of the updates are made permanent. Transactions are started and ended by applications or the container in which the applications are deployed. IBM WebSphere Application Servers applications can also be configured to interact with databases, JMS queues, and JCA connectors through their local transaction support when distributed transaction coordination is not required.

  The dynamic cache service improves performance by caching the output of servlets, commands, and JSP™ files. The dynamic cache works within an application server, intercepting calls to cacheable objects. The following caching features are available in IBM WebSphere Application Server:

  – Cache replication

    Cache replication among cluster members takes place using the IBM WebSphere internal replication service.

  – Cache disk off load

    When the number of cache entries reaches the configured limit for a given IBM WebSphere Application Server, eviction of cache entries takes place, allowing new entries to enter the cache service.

  – Edge Side Include (ESI) caching

    The ESI processor has the ability to cache whole pages, as well as fragments, providing a higher cache hit ratio.

  – External caching

    The dynamic cache has the ability to control caches outside of the application server.

  An Object Request Broker (ORB) manages the interaction between clients and servers, using Internet InterORB Protocol (IIOP). It enables clients to make requests and receive responses from servers in a network-distributed environment. IBM WebSphere Application Server uses an ORB to manage communication between client applications and server applications, as well as communication among product components.

- Session Management

  In many Web applications, users dynamically collect data as they move through the site based on a series of selections on pages they visit. Where the user goes next, and what the application displays as the user's next page (or next choice) may depend on what the user has chosen previously from the site. In order to maintain this data, the application stores it in a session. IBM WebSphere Application Server supports three approaches to track sessions:

  – SSL session identifiers

    SSL session information is used to track the HTTP session ID.

  – Cookies

    The application server session support generates a unique session ID for each user, and returns this ID to the user's browser using a cookie. The default name for the session management cookie is JSESSIONID. Using cookies is the most common method of session management.

  – URL rewriting

    Session data can be kept in local memory cache, stored externally on a database, or kept in memory and replicated among application servers.

- Web service

  Web services are self-contained, modular applications that can be described, published, located, and invoked over a network. IBM WebSphere Application Server can act as both a Web service provider and as a requester. As a provider, it hosts Web services that are published for use by clients. As a requester, it hosts applications that invoke Web services from other locations. IBM WebSphere Application Server supports SOAP-based Web service hosting and invocation.

- Security

  IBM WebSphere Application Server security sits on top of the Operating System security and the security features provided by other components, including the Java language, as shown in Figure 5-17 on page 162.

  – The Operating System security protects sensitive IBM WebSphere Application Server configuration files and authenticates users when the Operating System user registry is used for authentication.

  – Standard Java security is provided through the Java Virtual Machine (JVM) used by IBM WebSphere Application Server and the Java security classes.

  – The Java 2 Security API provides a means to enforce access control, based on the location of the code and who signed it. Java 2 security guards access to system resources such as file I/O, sockets, and properties.

– The Common Secure Interoperability (CSIv2) protocol adds additional security features that enable interoperable authentication, delegation, and privileges in a CORBA environment.

– J2EE security uses the security collaborator to enforce J2EE-based security policies and support J2EE security APIs. APIs are accessed from IBM WebSphere applications in order to access security mechanisms and implement security policies. J2EE security guards access to Web resources, such as servlets/JSPs and EJB methods based on roles defined by the application developer. Users and groups are assigned to these roles during application deployment.

– IBM Java Secure Socket Extension (JSEE) is the Secure Sockets Layer (SSL) implementation used by IBM WebSphere Application Server. It is a set of Java packages that enable secure Internet communications. It implements a Java version of SSL and Transport Layer Security (TLS) protocols and includes functionality for data encryption, server authentication, message integrity, and client authentication.

Figure 5-17 presents a general view of the logical layered security architecture model of IBM WebSphere Application Server. The flexibility of that architecture model lies in pluggable modules which can be configured according to the requirements and existing IT resources.

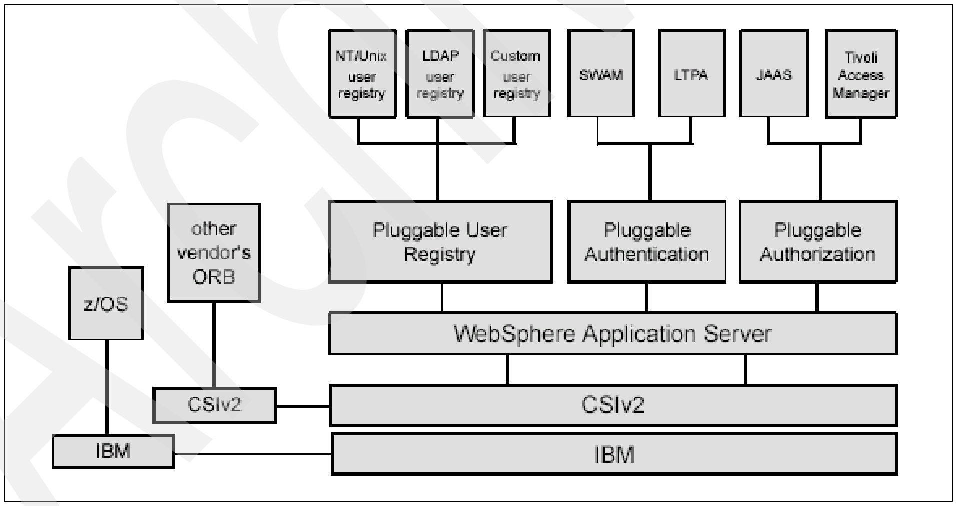


Figure 5-17 IBM WebSphere Application Server security architecture

The pluggable user registry allows configuring different databases to store user IDs and passwords that are used for authentication and authorization. The options are:

– Local Operating System user registry, but this cannot be used in a clustered environment.

– LDAP user registry, which is the most common in an enterprise environment.

– Custom user registry, which can be used to adapt to a specific environment.

An authentication mechanism in IBM WebSphere Application Server typically collaborates closely with a user registry when performing authentication. The authentication mechanism is responsible for creating a credential that is an IBM WebSphere Application Server internal representation of a successfully authenticated client user.

- Resource providers

  Resource providers define resources needed by running J2EE applications. Some of the most common resource providers supported by the IBM WebSphere Application Server configuration are:

  – Java DataBase and connectivity (JDBC)

    A data source represents a real-world data source, such as a relational database. When a data source object has been registered with a JNDI naming service, an application can retrieve it from the naming service and use it to make a connection to the data source it represents.

  – J2EE Connector Architecture (JCA)

    It defines a standard architecture for connecting the J2EE platform to heterogeneous Enterprise Information Systems (EIS), for example Enterprise Resource Planning (ERP), mainframe transaction processing, database systems, and existing applications not written in the Java programming language. The JCA resource adapter is a system-level software driver supplied by EIS vendors or other third-party vendors. It provides the connectivity between J2EE components (an application server or an application client) and an EIS.

  – Java Message Service

    The JMS functionality provided by IBM WebSphere Application Server includes support for three types of JMS provider:

    - IBM WebSphere JMS provider (embedded messaging)
    - IBM WebSphere MQ JMS provider
    - Generic JMS providers

5.6.3 IBM WebSphere Application Server planning considerations

This section provides configuration settings recommendations that are relevant to create a performing IBM WebSphere Application Server infrastructure for the IBM Tivoli License Manager environment. This section does not present an exhaustive list of IBM WebSphere Application Server configuration capabilities and only focuses on the configurations that can have an impact on the overall performance of the J2EE environment used by IBM Tivoli License Manager applications.

IBM Tivoli License Manager consists of two different J2EE applications as follows:

- SLM_Admin_Application identifies the ITLM Administration server application.
- SLM_Runtime_Application identifies the ITLM Runtime server application.

Both of them are Web applications, which means that they do not contain EJB or Client Application modules.

As the above applications are Web applications running on top of IBM WebSphere Application Server, the focus in terms of performance tuning is reduced to the handling of HTTP traffic and database connections. In terms of failover, the focus is to ensure that at least one instance of each application is up and running.

The following sections contain some of the considerations for these applications. The following topics are discussed:

- JVM sizing consideration
- Java Garbage Collector
- HTTP session management
- Threading
- Web container sizing
- Database connection tuning
- HTTP server tuning

JVM sizing consideration

For JVM sizing, it is important to keep in mind that moderate heap sizes work best since large heaps require longer garbage collection cycles. Since the HTTP session is stored in the JVM heap, its size should be configured accordingly; indeed, overflowed session retrieval can be costly (Java object serialization).

For some IBM Tivoli License Manager production systems, the JVM heap size is often initially set to be lower than the maximum allowed value in a 1:1.33 ratio. In addition, never set the maximum JVM heap size to be larger than the physical amount of memory of the server.

Java Garbage Collector

The JVM manages memory and the reuse of objects via the Java Garbage Collector. The JVM Garbage Collector is a routine running on the background that searches memory to reclaim space from program segments or inactive data and is know to be a CPU consumer.

By default, the JVM unloads a class from memory when there are no live instances of that class left, but this can degrade performance. Turning off class garbage collection eliminates the overhead of loading and unloading the same class multiple times. This is specially important when the ITLM Administration server or the ITLM Runtime server run on shared IBM WebSphere Application serves.

The recommendation is to have the Java Garbage Collector disabled on both the ITLM Administration server and ITLM Runtime server.

HTTP session management

Since HTTP is a stateless protocol, IBM WebSphere allows the application developer to manage the connection state via HTTP session management.

There are many aspects of managing HTTP sessions, which could include specifying a session tracking mechanism, setting maximum in-memory session count, controlling overflow, and configuring session time out.

Note that to preserve performance, in the context of IBM Tivoli License Manager, we recommend reducing the value of session time out (the default is 30 minutes). Reducing this value to a lower number can help reduce memory consumption requirements, allowing a higher user load to be sustained for longer periods of time. This is especially important when ITLM Agents are uploading software usage and inventory information to ITLM Runtime servers.

Threading

As a sequential flow of control, a thread must carve out some resources within a running program. A thread pool enables components of the server to reuse threads to eliminate the need to create new threads at run time. Creating new threads expends time and resources.

Figure 5-18 shows a series of executions using threads pools for a generic JE22 application running on IBM WebSphere Application Server. Not all Web requests need to access information about the database, and may not use the EJB container to do so, which is the case of IBM Tivoli License Manager applications.

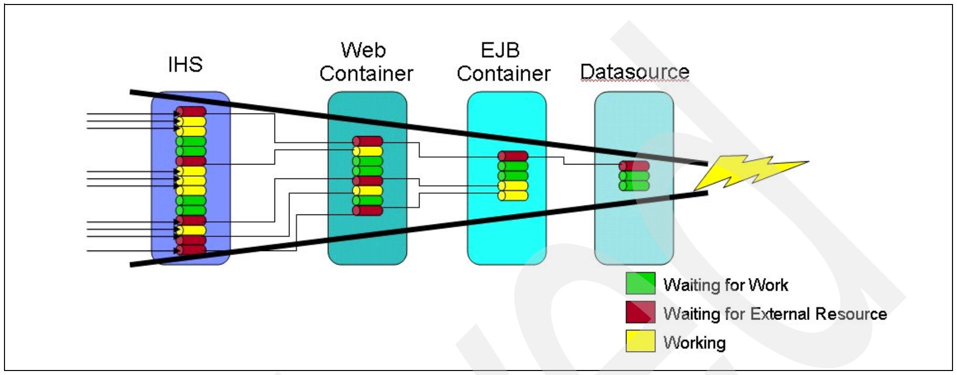


Figure 5-18 IBM WebSphere funnel request processing

Information related to software usage is directly managed in the cache of the ITLM Runtime server, thus decreasing the number of access to the ITLM Runtime server database. On the other hand, the ITLM Administration server does not manage real time information, and the majority of configuration and historical information received from and sent to ITLM Runtime server is maintained on the ITLM Administration server database.

We highly recommend using the WebSphere Runtime Performance Advisor to help tune the ITLM Administration server and ITLM Runtime server systems for optimal performance after initial deployment of the IBM Tivoli License Manager infrastructure to adapt the thread pools according to the effective load.

The WebSphere Runtime Performance Advisor uses the Performance Monitoring Infrastructure (PMI) data to suggest configuration changes to Object Request Broker (ORB) service thread pools and Web container thread pools.

Web container sizing

When sizing the Web container, it is important to notice that the thread pool determines the maximum concurrency. The pool sweet spot is around 50 to 75 threads. To keep the pool constant and avoid unnecessary allocation and cleaning, the minimum should be equal to the maximum. Unbounded growth should be avoided since it can lead to unbounded consumption of resources. Also, too many threads should be avoided; there will be no additional throughput gain, but this will increase the JVM thread management burden.

The thread pool size determines the maximum concurrency.

Note that setting too many threads is harmful, as it increases the JVM thread management burden and does not provide additional throughput gains.

For example, in the context of a medium to large IBM Tivoli License Manager environment, we may assume the ITLM Administration server handles an average value of five requests per second. A good start point for the minimum and maximum thread initial pool size settings for the ITLM Administration server would be as follows:

Minimum Thread Pool size	5 * 2/3 = 3
Maximum Thread Pool size	5 * 3 = 15
Thread inactivity time out	60000 ms/5 * 3 = 12000 ms
Is Growable	NO

Making the assumption of having 5.000 connections to the ITLM Runtime server with 4 requests in one hour, this results in 20.000 requests per hour in peak time. That is 5.5 per second with an even load. The assumption that could be done is to have up to three times this load in rare cases, that is, about 15 requests per second. Thus, the ITLM Runtime server needs to be able to address 15 simultaneous connections to the database in peak time. The thread initial pool size settings for the ITLM Runtime server would be as follows:

Minimum Thread Pool size	15 * 2/3 = 10
Maximum Thread Pool size	15 * 3 = 45
Thread inactivity time out	60000 ms/15 * 3 = 4000 ms
Is Growable	NO

Database connection tuning

Database performance tuning can dramatically affect the throughput of IBM Tivoli License Manager applications. For example, the ITLM Administration server application requires multiple, simultaneous interactions with a back-end ITLM Administration server database, and an improperly tuned database and database connection can result in a bottleneck. Database access threads accumulate in a backlog when the database is not configured to accept a sufficient number of incoming requests.

Database connection tuning involves two aspects:

- Data source tuning

  From an IBM Tivoli License Manager perspective, data source tuning is related to statement cache size settings. This specifies the number of statements that can be cached per connection. The IBM WebSphere data source optimizes the processing of prepared statements and callable statements by caching those statements that are not being used in an active connection. Both statement types help reduce overhead for transactions with back-end data.

- Connection pool tuning

  From an IBM Tivoli License Manager perspective, connection pool settings are related to maximum connections settings. This specifies the maximum number of physical connections that can be created in the pool. These are the physical connections to the back-end data store. When this number is reached, no new physical connections are created; requestors must wait until a physical connection that is currently in use is returned to the pool.

  For both ITLM Administration server and ITLM Runtime server, for optimal performance, set the value for the connection pool lower than the value for the Web container threadpool size (described in "Web container sizing" on page 166).

For example, in the context of a medium to large IBM Tivoli License Manager environment, we may assume the ITLM Administration server executes an average of 15 requests per second and they are long-running requests on many DB tables. A good start point for the minimum and maximum initial connection pool size as well as statements cache settings for the ITLM Administration server would be as follows:

Minimum pool size	15
Maximum pool size	20 + 50% margin = 30
Connection time out	60 s.
Idle time out	1200
Orphan time out	1200
Statement cache size	200
Disable auto connection cleanup	Not selected

Making the assumption that the ITLM Runtime server handles three times more HTTP requests in peak time then the ITLM Administration server, we have an average of 45 requests per second with an even load. Assuming also that a HTTP session could wait three minutes to get a connection and after 30 minutes, the session could be freed, a good start point for the minimum and maximum initial connection pool size as well as statements cache settings for the ITLM Runtime server would be as follows:

Minimum pool size	5
Maximum pool size	45 + 50% margin = 60
Connection time out	180 s.
Idle time out	1800
Orphan time out	1800
Statement cache size	20
Disable auto connection cleanup	Not selected

HTTP server tuning

The HTTP Server is the front end to IBM WebSphere Application Servers via the Web server plug-in. Each IBM Tivoli License Manager supported HTTP server operating system combination has specific tuning parameters that affect the both ITLM Administration server and ITLM Runtime server applications performance. All of these HTTP servers allocate a thread to handle each client connection. Ensuring that enough threads are available for the maximum number of concurrent client connections helps prevent this tier from being a bottleneck.

Every communication exchanged between the ITLM Agents and their ITLM Runtime servers and between ITLM Runtime servers and the central ITLM Administration server is issued by the HTTP server.

This section focuses on the IBM HTTP Server Version 6 and discusses some of the parameters that can be adjusted to improve performance. The following is not an exhaustive list and are provided here for reference and initial performance tuning activities:

- Persistent requests

  Specifies the maximum number of persistent (keep-alive) requests that are allowed on a single HTTP connection. If the parameter MaxKeepAliveRequests has a value of 0 (zero), only one request is allowed per connection. If a value of -1 is specified, an unlimited number of requests is allowed per connection. The default value of 100 is used initially on most medium sized IBM Tivoli License Manager implementations.

  As for the time out value of such requests, the KeepAliveTimeout parameter specifies the number of seconds to wait for the next request from the same client on the same connection. It can be helpful to increase the time out value of persistent requests; however, it is important to note that a higher time out may increase contention for HTTP server processes. A good starting point is between 5 and 15 seconds. The default value of 10 is used initially on most medium sized IBM Tivoli License Manager implementations for both ITLM Administration server and ITLM Runtime servers.

- Number of concurrent threads at a time

  The IBM WebSphere and IBM HTTP server contains interrelated components that must be harmoniously tuned to support the needs of applications. The HTTP server also needs to be configured to avoid having more inbound requests than you have HTTP threads to handle those requests. The number of concurrent threads at a time is controlled by the MaxClients (UNIX platforms) or ThreadsPerChild (Windows platforms) parameter. The default value of 250 is used initially on most medium sized IBM Tivoli License Manager implementations for both ITLM Administration server and ITLM Runtime servers.

- Maximum request per child

  The IBM HTTP server always creates one child process to handle requests. If it dies, another child process is created automatically. Within the child process, multiple threads handle incoming requests. The MaxRequestsPerChild parameter sets the limit on the number of requests that an individual child server process will handle. After MaxRequestsPerChild requests, the child process will die. If MaxRequestsPerChild is 0, then the process will never expire.

Setting MaxRequestsPerChild to a non-zero limit has two beneficial effects:

– It limits the amount of memory that process can consume by (accidental) memory leakage.

– By giving processes a finite lifetime, it helps reduce the number of processes when the server load reduces.

The MaxRequestsPerChild value of 10000 is used initially on most medium sized IBM Tivoli License Manager implementations for ITLM Administration server and MaxRequestsPerChild value of 20000 for ITLM Runtime servers.

5.7 IBM Tivoli License Manager servers settings

This section provides configuration options for the physical components of a IBM Tivoli License Manager solution. The optimum settings for these configuration options are based on the architectural decisions for the physical design and in accordance with topics described in 5.1, “Physical components considerations” on page 91.

As set of parameters is available to configure and define the behavior of all IBM Tivoli License Manager servers, including the behavior of the administrative Web interface, the handling of e-mail notifications, the behavior of the ITLM Administration server and ITLM Runtime server in terms of scalability and performance, as well as global ITLM Agents settings that will define the general behavior of these agents.

This set of parameters are contained in configuration files. The system.properties file is the main configuration file of IBM Tivoli License Manager servers and the following sections go into details of some of the parameters in these configuration files.

Refer to the Appendix A, “IBM Tivoli License Manager V2.2 servers system configuration file” on page 239 for a description of the system.properties parameters for both ITLM Administration server and ITLM Runtime server, as well as the ITLM Agent.

The following topics are discussed in this section:

- 5.7.1, “Additional security topics” on page 172
- 5.7.2, “Timing of IBM Tivoli License Manager server services” on page 173
- 5.7.3, “ITLM Administration server system parameters” on page 174
- 5.7.4, “ITLM Runtime server system parameters” on page 176
- 5.7.5, “ITLM Agent system parameters” on page 178

5.7.1 Additional security topics

The following sections discuss additional aspects of security on an IBM Tivoli License Manager environment. Refer also to 5.2, “Communication type and security options” on page 99 for a general overview of security options of a IBM Tivoli License Manager solution.

IBM Tivoli License Manager servers database authentication

The install process of both ITLM Administration server and ITLM Runtime servers requires the creation of an user ID for their application to have authenticated access to their respective databases. In all cases, the user ID is *tlmsrv*. The tlmsrv user ID can be defined prior to installation time, but we highly recommend that it be created during the installation process, as it ensures that the proper authorization levels are granted to the user ID.

In order to simplify password management of this unattended user ID, we highly recommend having the same password for the tlmsrv user ID throughout the IBM Tivoli License Manager environment. In addition, this password must be compliant with existing password management policies in place. To ease this task, IBM Tivoli License Manager provides a script to automate tlmsrv user ID password management.

ITLM Runtime server registration password

When the level of security for communications between the ITLM Runtime server and ITLM Administration server is set to Maximum, IBM Tivoli License Manager requires that SSL communication be used. In this case, not only does both the ITLM Runtime server and ITLM Administration server require a password for accessing their respective key stores (as described in 5.2.2, “Secure Sockets Layer communications” on page 105), but also a password to authenticate the identity of the ITLM Runtime server to the ITLM Administration server.

This password is stored in the passwd.properties file encrypted using a FIPS-approved algorithm and its value must match the password specified when the ITLM Runtime server details are registered to the ITLM Administration server at initial plug-in time.

Although the communications password is not linked to any user ID, it still may also follow existing password management policies.

5.7.2 Timing of IBM Tivoli License Manager server services

The timing of IBM Tivoli License Manager servers services and events is determined by the following:

- The service or event start time

  The start time is either determined by the time that the server last started, or by the parameters <server>BaseTime defined in the system.properties file of each server, where <server> is either admin or runtime: adminBaseTime for ITLM Administration server and runtimeBaseTime for ITLM Runtime servers. The <server>BaseTime parameter can take any integer value between 0 and 23.

- The interval between services or events

  Each service or event has a parameter defined in the system.properties file that determines how often a service or event will be performed. Only parameters suffixed by the term *period* are affected by the <server>BaseTime. For example, the cleanUsagePeriod parameter determines the frequency of non aggregated software use database tables cleanups.

The start time of an event or service is either determined by the time that the IBM Tivoli License Manager server last started, or by the parameter. If the frequency of an event or service is exactly 1, 2, 3, 4, 6, 8, 12, or 24 hours, independently of the measurement unit, that event or service will use the appropriate <server>BaseTime parameter as its start time.

The following are recommendations on how the <server>BaseTime parameters of the IBM Tivoli License Manager servers should be set:

- The ITLM Administration server should not have the same BaseTime as any of the ITLM Runtime servers. This is to avoid having the ITLM Administration server performing its periodic database services while the ITLM Runtime servers are sending their data.
- The BaseTime on ITLM Runtime servers should be set so that the ITLM Administration server starts its services after all of ITLM Runtime server have finished sending their information.
- Ideally, ITLM Runtime servers should have different BaseTime settings.
- BaseTimes apply to local time, so in making the above calculations, zone differences between the locations of the servers must be taken into account.

ITLM Agents also run some regular tasks that are defined in the ITLM Runtime server system.properties configuration file. The BaseTime used by the ITLM Agent is initially the time when the ITLM Agent has been installed. If the ITLM Agents missed a service period because they were switched off or disconnected, they will start their services as soon as they will be connected. However, in order to avoid all ITLM Agents starting their services at the same time, there is a local algorithm preventing a connection overload.

5.7.3 ITLM Administration server system parameters

This section discusses some important ITLM Administration server configuration items. All these configuration items are manipulated through parameters in the system.properties file in the ITLM Administration server.

Software usage data management

IBM Tivoli License Manager collects software usage information based on software start or stop activities on a machine in which the ITLM Agent is running. This software usage information is stored in the ITLM Administration server database. It is important to control how fast the tables managing usage information grow and for how long the aggregated software usage should be maintained.

There are two aspects that need to be taken into account for software usage information: The software usage data sent by ITLM Runtime servers that has been received by the ITLM Administration server and has not been aggregated, and the software usage information that has already been aggregated.

Having this in mind, the following decisions must be made in order to have a performing environment:

- When the software usage data need to be aggregated and how often the aggregation service should run.
- How long to keep aggregated and non aggregated software usage information.
- When and how often software usage cleanup processes should run.

The software usage information is stored in the IBM Tivoli License Manager servers USAGE table. IBM Tivoli License Manager has an internal cleanup process that runs at a predefined interval. However, this process only cleans data that is older than a predefined number of days.

This usage information needs to be aggregated for historical storage. Based on the same principle as for the usage cleanup, IBM Tivoli License Manager has an internal process that aggregates the software usage information, meaning it consolidates information from the USAGE table to historical tables, and this process only aggregates information that is older than a predefined number of days.

The IBM Tivoli License Manager configuration parameters in the system.properties file related to this subject are cleanUsagePeriod, maxUsageAge, aggregateUsagePeriod, and maxAggregateUsageAge.

ITLM Runtime server management

Although communications with ITLM Administration server are always initiated and controlled by the ITLM Runtime server (this is further described in 5.7.4, “ITLM Runtime server system parameters” on page 176), IBM Tivoli License Manager provide parameters that define the behavior on how the ITLM Administration Server manages its ITLM Runtime servers.

The ITLM Administration server is able to check if changes have been made to information that a (or a group of) ITLM Runtime server may need. If changes have been made since the last contact with the ITLM Runtime server, the ITLM Administration server waits until that particular ITLM Runtime server requests information updates to send the updates. If the information needs to be sent to a group of ITLM Runtime servers, the ITLM Administration server keeps track of that and ensures the information is sent to every ITLM Runtime server in the group.

The ITLM Administration server also needs to frequently check the availability of its ITLM Runtime server. The standard behavior is the ITLM Runtime servers have to regularly initiate a communication with their ITLM Administration server. The ITLM Administration server keeps records of last update requests from all ITLM Runtime servers and can declare them inactive if there was no communication exchanged between a predefined period of time. The configuration parameter related to this is maxServerInactivity defined in the system.properties file of the ITLM Administration server.

Product inventory data management

ITLM Runtime servers upload software inventory information collected by their ITLM Agents to the ITLM Administration server. At defined intervals, the ITLM Administration server aggregates and reconciles the installed software information collected by the ITLM Agents, which identifies the software components that are installed on monitored computers, with the product information held on the ITLM Administration server database. In this way, the inventory of components is converted to an inventory of products, in which components are assigned according to the catalog information and the mappings of shared components.

The interval of time between consecutive builds of the software inventory on the ITLM Administration server is controlled by the productInventoryBuilderPeriod parameter in the system.properties file of the ITLM Administration server.

Software monitoring

Software monitoring enables IBM Tivoli License Manager to collect and report information about usage and installation of software applications. The default is Disabled. You should only enable product monitoring if you want to collect use statistics for the product to be monitored. When a product is assigned to a use license, the value of this property is automatically modified to Enabled.

It is important to have complete control over which software is enabled for monitoring, as the amount of information collected by ITLM Agents and sent to the ITLM Administration server via their ITLM Runtime server needs to be controlled to minimize the impact on the scalability of the IBM Tivoli License Manager solution.

5.7.4 ITLM Runtime server system parameters

This section discusses some important ITLM Runtime server configuration items. All these configuration items are manipulated through parameters in the system.properties file in the ITLM Runtime server.

ITLM Runtime server plug-in

After the installation or a restart of an ITLM Runtime server, the ITLM Runtime server must plug in to the ITLM Administration Server. This process is initiated directly after the installation or a restart. However, in case this plug-in request fails for any reason, the ITLM Runtime server will retry to connect to the ITLM Administration server based on a predefined frequency.

The configuration parameter related to this subject is runtimePluginPeriod, defined in the system.properties file of the ITLM Runtime server.

Data exchange management

The ITLM Runtime server is in fact in charge of managing the exchange of data, either the download or the upload, with the ITLM Administration server.

The type of information that needs to be uploaded from the ITLM Runtime server to the ITLM Administration server is:

- Software inventory and ITLM Agent information.
- The list of discovered applications installed on the ITLM Agents that are not part of the ITLM software catalog (also known as unknown files or potential signature files list) that have arrived from the ITLM Agents since the last update.
- Software monitoring information.
- The software usage snapshots. The snapshots are sent periodically to enable repair of the usage history if any software usage information is lost.

The type of information that needs to be downloaded by the ITLM Runtime server from the ITLM Administration server is:

- Updated license information
- ITLM Master catalog information
- Topology information

The configuration parameters related to this subject are adminDownloadPeriod and adminUploadPeriod, and are defined in the system.properties file on the ITLM Runtime server.

ITLM Agents management

One of the many tasks the ITLM Runtime server performs is to manage ITLM Agents, which is done by checking the availability of the ITLM Agent running on the nodes. As the ITLM Runtime server is not able to initiate connections to ITLM Agents, the ITLM Agents are configured to contact the ITLM Runtime server on a regular basis either for availability check or for download requests.

If the ITLM Runtime server does not receive any contact from an ITLM Agent in a predefined period of time, it will consider the ITLM Agent as inactive. This means the information related to this ITLM Agent will not be included any more in the compliance report, so this will release all licenses used by this Agent.

This period must not be lower than the downloadParametersPeriod defined in the Agent settings section of the system properties file.

The configuration parameter related to this subject is maxAgentInactivity, and is defined in the system.properties file on the ITLM Runtime server.

Privacy policy management

An important consideration for an IBM Tivoli License Manager solution is that some countries do not allow you to store or disclose user information either in the country or outside the country the user is located.

The IBM Tivoli License Manager configuration allows for the implementation and enforcement of data privacy policies.

The configuration parameter related to this subject is storeUser, and is defined in the system.properties file on the ITLM Runtime server. Depending on its value, information regarding user identification is recorded with software usage information or not.

5.7.5 ITLM Agent system parameters

This section discusses some important ITLM Agent configuration items. All these configuration items are manipulated through parameters in the system.properties file in the ITLM Runtime server. These ITLM Agent settings are effective for all the ITLM agents registered with the ITLM Runtime server. ITLM Agent settings are downloaded from the ITLM Runtime server to the ITLM Agent at defined intervals.

Communications management

The ITLM Runtime server has to manage the status of the ITLM Agents, in order to define if an ITLM Agent is either active or inactive. As the ITLM Runtime server is not able to contact directly an ITLM Agent to verify its status, the ITLM Agent is in charge of frequently providing status information to the ITLM Runtime server.

An important configuration parameter is the ability to define the frequency of such status' reports. The ITLM Agent communicates with only one ITLM Runtime server. As ITLM Runtime servers may control many ITLM Agents, it is important to minimize status' reports from ITLM Agents to an optimum frequency to avoid unnecessary network traffic. The configuration parameter related to this subject is pingPeriod, and it is defined in the system.properties of the ITLM Runtime server.

Data exchange management

The ITLM Agent is in fact in charge of managing the exchange of data, either the download or the upload of the information, with its ITLM Runtime server. It is important to notice that the ITLM Runtime server is not able to contact the ITLM Agent.

The type of information that needs to be uploaded from the ITLM Agent to the ITLM Runtime server is:

- Offline software usage
- Software monitoring information
- The potential signatures file information
- Software inventory information

The type of information that needs to be downloaded by the ITLM Agent from the ITLM Runtime server is:

- The parameters from the system.properties file of the ITLM Runtime server that are related to the ITLM Agent
- The software catalog information

The configuration parameters related to this subject are downloadParametersPeriod, downloadTopologyPeriod, offlineUsagePeriod, and uploadMinTime.

ITLM Agent self update

If a new version, Fix Pack, or interim fix that contains an upgrade to the ITLM Agent is available on the ITLM Runtime server, it is possible to automatically upgrade ITLM Agents running code. If enabled for self update, the ITLM Agent regularly checks with its ITLM Runtime server and if a new version of the ITLM Agent files is found, the ITLM Agent automatically downloads and deploys the new files.

The self update of ITLM Agents can be configured for one, a group, or all of the ITLM Agents controlled by a ITLM Runtime server. The configuration parameters related to this subject are updateAgentEnabled and updateAgentPeriod.

Software usage

The ITLM Agent maintains a list of software usage information named agent process list. Comparison of consecutive versions of the agent process list enables the ITLM Agent to determine which software applications have been executed or stopped during a period of time. The period of time between between checks of the agent process list is configurable using the processListPeriod defined in the systems.properties file on the ITLM Runtime server. Depending on the type of node the ITLM Agent is running, the processListPeriod should not be configured to be too long, as it may lead the ITLM Agent not to identify the start and stop of software applications and therefore the software monitoring and usage information collected by the ITLM Agent may not be accurate.

J2EE applications usage

In case of J2EE applications running on IBM WebSphere Application Server, the ITLM Agent makes use of a sub agent named WebSphere agent. If the following IBM WebSphere Application Server products are identified by the ITLM Agent on the Node, the WebSphere Application Server Agent is automatically executed:

- IBM WebSphere Application Server Version 6.0
- IBM WebSphere Application Server Version 5.1
- IBM WebSphere Application Server Version 5.0
- IBM WebSphere Portal

Refer to *IBM Tivoli License Manager Planning, Installation, and Configuration*, SC32-1431 for an up to date list of supported Web Application Server versions.

The ITLM Agent manages the execution and availability of the WebSphere Application Server Agent on a regular basis, based on the wasAgentCheckPeriod parameter defined in the systems.properties file on the ITLM Runtime server.

5.8 Backup and recovery considerations

Unless a recovery plan is defined and implemented, the recovery of an IBM Tivoli License Manager environment cannot be done within a reasonable period of time. The more preventative measures are added to the backup and recovery procedures, the lower the chances are of a situation resulting in a disaster to the organization. No matter how sophisticated the preventative measures are, however, there are always risks of outages.

The following are compelling reasons for defining a disaster recovery plan for an IBM Tivoli License Manager solution:

- The IBM Tivoli License Manager environment contains the compliance definition of the licenses agreements.
- The IBM Tivoli License Manager environment contains important financial information and license contracts definitions.

The creation of backups to protect the files from disaster is not just a question of taking copies of the data. The IBM Tivoli License Manager environment is spread over several computers, and relies on the synchronization of the data from various databases.

This section describes the backup and recovery requirements of each layer of the IBM Tivoli License Manager solution as follows:

- 5.8.1, “Database environment backup” on page 181
- 5.8.2, “HTTP and Web application servers configuration backup” on page 182
- 5.8.3, “IBM Tivoli License Manager servers configuration backup” on page 182

5.8.1 Database environment backup

The IBM Tivoli License Manager databases contain all of the defined procurement and topology information, as well as information collected by all ITLM Agents.

In order to keep the loss of data to a minimum, a regular backup plan of the IBM Tivoli License Manager database environment must be planned and implemented. The backup plan should only include the ITLM Administration server database, as we do not recommend backing up and restoring ITLM Runtime server databases in order to avoid data inaccuracy, as most of the information is managed centrally on the ITLM Administration server database.

In the event of a ITLM Runtime server database corruption, it could be easily reconstructed using information already stored in the ITLM Administration server database. In addition, as usage information is stored in the ITLM Runtime server cache, recovering a ITLM Runtime server database does not cause any loss of usage data sent by ITLM Agents.

Note: The ITLM Administration server must be stopped during the backup or restore of the its database. This has been noticed as a common mistake when the ITLM Administration server and its database are not on the same physical system, resulting in a corrupted database restore.

ITLM Runtime servers do not need to be stopped during the backup or restore of the ITLM Administration server database.

The recommendation is to have a full backup of the ITLM Administration server database on a weekly basis with a daily incremental backup.

5.8.2 HTTP and Web application servers configuration backup

As described in 5.6, “HTTP and Web Application Servers” on page 156, IBM Tivoli License Manager consists of two different J2EE applications as follows:

- SLM_Admin_Application identifies the ITLM Administration server application.
- SLM_Runtime_Application identifies the ITLM Runtime server application.

During installation time, the above applications are deployed and configured on top of a IBM WebSphere Application Server. These applications have their own configuration files that must also be part of the backup plan.

IBM WebSphere Application Server has utilities to back up the configuration of applications to a file. These files can then be backed up and used later in case of recovery. These utilities are backupConfig and restoreConfig and are described in details in the IBM WebSphere Application Server InfoCenter at `http://www.ibm.com/software/webservers/appserv/infocenter.html`.

HTTP servers configuration files are usually changed to improve performance and security requests. All IBM Tivoli License Manager servers must have their HTTP Server configuration files included in a formal backup plan. As the IBM HTTP Server does not have any built-in utility to back up the configuration files, the recommendation is to include all files stored in the <IBM_HTTP_INSTALL_DIR>/conf directory in the backup plan, where <IBM_HTTP_INSTALL_DIR> is the installation directory of IBM HTTP Server on both ITLM Administration server and all ITLM Runtime servers.

5.8.3 IBM Tivoli License Manager servers configuration backup

As described in 5.7, “IBM Tivoli License Manager servers settings” on page 171, IBM Tivoli License Manager provides extensive configuration facilities that are essential for a performing IBM Tivoli License Manager environment.

IBM Tivoli License Manager also provides commands that facilitate the backup and restore of IBM Tivoli License Manager server settings, that is, the **`backupconf`** and **`restoreconf`** commands.

The **`backupconf`** command is used by the install wizard during the server installation, so that a backup of the default settings is available should anything go wrong while initial changes occur in these files.

Whenever a configuration file is changed, a validation should be done to verify that the desired result has been achieved. If it has, the **`backupconf`** command could be run to indicate a desired state or IBM Tivoli License Manager server settings in case of a restore. The **`backupconf`** command copies all the configuration files to a predetermined backup directory, overwriting whatever is already there. If at any time the configuration files need to be restored, the appropriate restore command must be run, and all the files will be restored.

> **Note:** Many of the configuration files are inter-dependent. Thus, we do not recommend restoring files individually from the backup, as it may compromise the viability of the configuration. The correct procedure is to back up the complete set of files whenever a change is operated, and restore a complete set whenever it is needed.

The backup of the configuration files are stored in the following directories:

- For the ITLM Administration server:

  <ITLM_INSTALL_DIR>\admin\AdminBackupConf\

- For the ITLM Runtime server

  <ITLM_INSTALL_DIR>\runtime\RuntimeBackupConf\

where <ITLM_INSTALL_DIR> is the installation directory of the respective IBM Tivoli License Manager server.

6

IBM Tivoli License Manager logical design

This chapter addresses the logical design for the implementation of IBM Tivoli License Manager. The logical design identifies how the IBM Tivoli License Manager Topology will be configured. Although the logical design of an IBM Tivoli License Manager solution can be modified and enhanced more easily after it is in place than the physical design, careful thought should be given in developing the logical design. The design becomes more complex over time and therefore harder to modify. A good physical design could be easily invalidated if the logical design is not well planned and executed.

The following topics are discussed in this chapter:

Note: This redbook refers to the IBM Tivoli License Manager V2.2 product, which has been recently rebranded as IBM Tivoli License Compliance Manager V2.2.

The IBM Tivoli License Manager product name is still in use in this redbook as well as on all of the administrative user interfaces and in the documentation of the product.

Refer to the product's Release Notes as they provide the most current information about the IBM Tivoli License Compliance Manager V2.2 release of the product. The latest version of the Release Notes can be found on the Tivoli Information Center Web site:

```
http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?toc
=/com.ibm.itlm.doc/toc.xml
```

6.1 Logical components considerations

Understanding the IBM Tivoli License Manager logical components helps with the definition of the IBM Tivoli License Manager Topology, in support of the following operations:

- The management and organization, from a logical stand point, of elements that are to be monitored by IBM Tivoli License Manager.
- The definition of software entitlements and specification of rules to be used in monitored software usage.
- The configuration of elements in support of procurement management activities.

The IBM Tivoli License Manager Topology is made up of the following elements:

- ITLM Runtime servers
- ITLM Agents
- Divisions
- Nodes
- Application users

In addition to the components that are part of the Topology, the following list gives the IBM Tivoli License Manager logical components:

- IBM Tivoli License Manager Administrators
- Components that support procurement management, such as contract and licenses definitions
- Components that support software entitlements, such as software usage monitoring and software license compliance reports

As mentioned in 3.4, “Logical components” on page 66, IBM Tivoli License Manager uses a logical hierarchy structure for managing the above described logical components that is made up of two levels as follows:

- Organization
- Division

An enterprise can be divided into one or more Organizations, each of which may contain one or more Divisions. Figure 6-1 shows an example of two possible logical structures for a sample enterprise. Case A uses each business unit of the enterprise as Organizations and their departments as Divisions. Case B defines a single Organization and each of the business units are Divisions logically attached to the single Organization. Both are valid cases the choice depends on, for example, the license requirements of the enterprise.

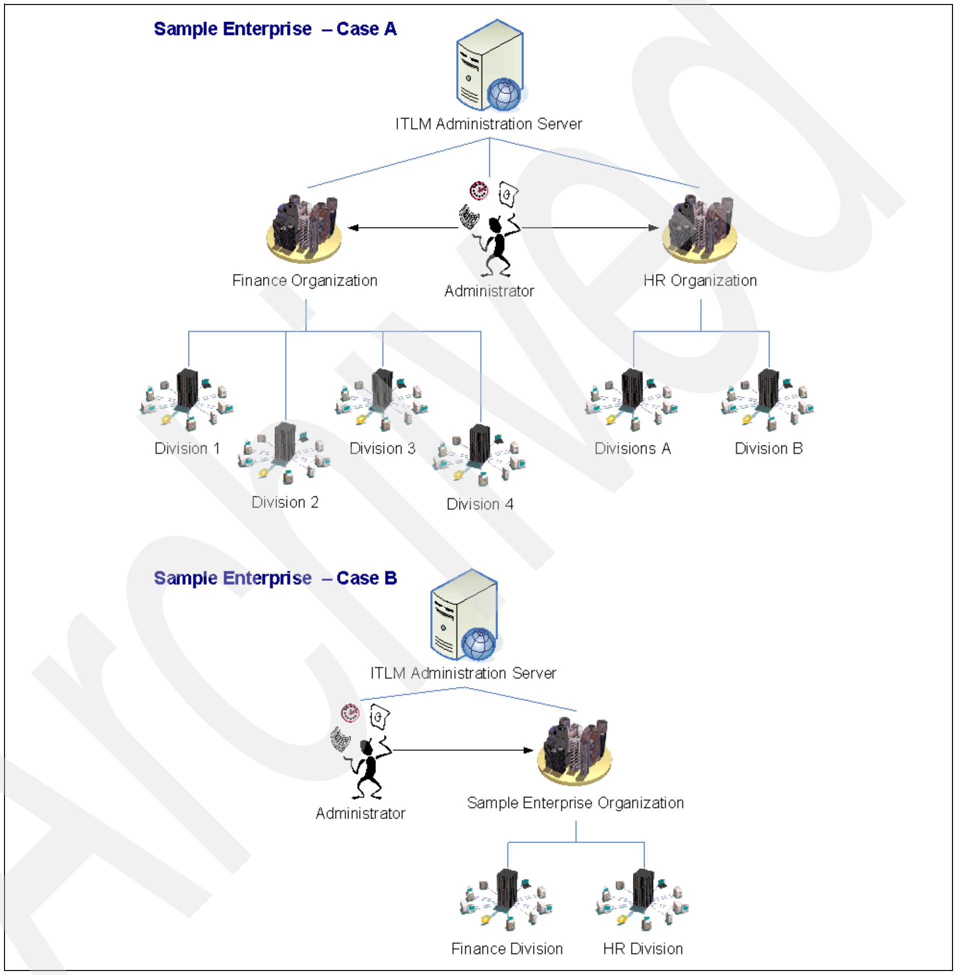


Figure 6-1 Organization and Divisions

The following sections provide design considerations on the IBM Tivoli License Manager logical hierarchy structure.

6.1.1 Organizations

An Organization is the highest level of the IBM Tivoli License Manager logical architecture. All IBM Tivoli License Manager logical components must belong to a single Organization, with the exception of the ITLM Administration server, which can serve multiple Organizations, and IBM Tivoli License Manager Administrators, which can manage more than one Organization.

Each Organization is the owner of its IBM Tivoli License Manager components, contracts, licenses, software usage and inventory data, reports, ITLM Runtime servers, and ITLM Agents. All of these components cannot be shared across Organizations.

The IBM Tivoli License Manager logical hierarchy divides the enterprise into one or more Organizations. The decision about how many Organizations you need to create depends on the logical units for which you need to control software licensing, contracts, and measure software usage.

Each Organization can reside on only one ITLM Administration server. All of the ITLM Runtime servers and ITLM Agents that are attached to that ITLM Administration server manage the licenses and report the software usage only for that Organization.

An Organization may be identified by an Organization code. Typically, the Organization code is set to a client account ID, in case of service providers, or an internal cost center.

Architectural decision sample: Due to the fact that the enterprise requires a global and consolidated view of its License Management structure, as well as the fact that licenses contract are negotiated at corporate level, a unique IBM Tivoli License Manager Organization will be used. All resources defined to the IBM Tivoli License Manager domain will belong to this Organization.

6.1.2 Divisions

A Division is the second level of the IBM Tivoli License Manager logical architecture and is part of the Organization's Topology. It is used to group ITLM Agents so that they can be logically organized. Divisions can be selected for some administration tasks, for example, to define different frequencies of software inventory scans, to create specific reports, and as a target of software licenses distributions.

At least one Division must be defined per Organization. Each ITLM Agent should be subscribed to only one Division.

Splitting the Organization structure into more than one Division gives more flexibility to manage the flow of inventory data. Even if each ITLM Agent only sends software inventory data that matches with a recognized software signature in the ITLM Master Catalog to avoid the upload of large amount of software inventory data from the same Division, having a large number of ITLM Agents members of the same Division may create a network bottle neck. Having multiple Divisions also facilitates the creation and analysis of License Management reports, administrative tasks, and provides better control of licenses definitions.

From an IBM Tivoli License Manager perspective, Divisions are the lowest structural unit into which an enterprise can be divided and often represent a specific level of the enterprise. Divisions can be created by taking into consideration, but not limited to, administrative layout structure, defined License Management requirements, geographical location, and amount of Nodes to be managed by IBM Tivoli License Manager, as well as networking topology. Divisions can also be logically split into different physical locations.

Considering the physical design of an IBM Tivoli License Manager solution, ITLM Agents are assigned to an ITLM Runtime server. For the Logical design of an IBM Tivoli License Manager solution, ITLM Agents are assigned to a Division.

Considering that Divisions are usually created reflecting the organizational structure of the enterprise and ITLM Runtime servers can serve multiple Divisions, the decision about how to logically distribute ITLM Agents among the IBM Tivoli License Manager solution comes down to performance. ITLM Agents should be assigned to local ITLM Runtime servers regardless of the Division in which they belong. We advise you to split Divisions with large numbers of ITLM Agents among ITLM Runtime servers.

Architectural decision sample: Divisions will be created representing the business unit structure of the enterprise, independent of the number of machines to be managed in each business unit. In case the business unit spans across multiple geographies, Divisions will be created to represent each geography following a pre-determined naming convention. In case new business units are created, corresponding Divisions will be created. All Divisions will be members of a single Organization defined for the enterprise and will be linked to a specific ITLM Runtime server. Using a proper naming convention, Divisions can spin off if the number of machines to be managed becomes too close to the maximum number of ITLM Agents supported by the ITLM Runtime server of that Division.

6.2 Naming policies

One of the most important steps during the Logical Design of the IBM Tivoli License Manager solution is to put in place a solid naming convention. It is also important that all personnel involved with the License Management project understand and adhere to the naming convention. If the naming convention is not followed, most likely, Administrators will lose track of the internal structure of the IBM Tivoli License Manager environment.

Immediate benefits of a well structured naming convention are, for example, the IBM Tivoli License Manager implementation leads itself to automation through scripting, easy to create, generate, and interpret reports provided by IBM Tivoli License Manager, and the environment becomes almost self-documenting with limited additional documentation required to understand the relationships of the different parts of the environment.

For large IBM Tivoli License Manager implementations, naming conventions have been used to virtually expand the two layer Organization - Division structure. Naming conventions for Divisions have been set so that virtual Divisions sub groups can be formed, facilitating administrative tasks, flow of inventory data, and greatly improving the scalability limits of the solution.

During the logical design phase of the project, naming policies must be defined and respected during the whole life of the solution independent of the IBM Tivoli License Manager component in question. Naming policies should be reviewed during each phase of the IBM Tivoli License Manager solution life cycle to ensure proper coherence.

Here we present a basic framework for the naming policies, as follows:

- Names must not contain spaces or special characters.
- Names must consist of alphanumeric characters, dash (-), or underscore (_).
- Names for the IBM Tivoli License Manager components are constructed by appending the component name, separated either by a dash (-), dot (.), or underscore (_).
- Underscores (_) should be used to replace blank spaces, dashes (-) to be used as type separators, and dots (.) to mark another level down in the name hierarchy.

Names must be defined in a way that give clarity and keep the name short. Administrators responsible for the setting up and maintenance of the IBM Tivoli License Manager environment must implement and follow the naming convention using the structure defined by the naming policies.

In addition to the component name, IBM Tivoli License Manager uses an unique identifier to further identify the component. This unique identifier is automatically generated by IBM Tivoli License Manager in case the IBM Tivoli License Manager Administrator defines the component using the IBM Tivoli License Manager administrative Web interface.

Components can also be defined by importing definitions stored in CSV or XML formatted files. In this case, the unique identifier must be specified in the definition files. A naming convention for the component unique identifier must also be put in place. A widely used naming convention for components unique identifier is as follows:

<ADMIN_ID>_<Counter>-<componentName>

Where <ADMIN_ID> is an internal ITLM Administration server identifier for administrative operations. The <ADMIN_ID> is automatically generated at ITLM Administration server installation time and stored in the ITLM Administration server database in the ADMIN_ID field of the CONTROL table.

Note: The <ADMIN_ID> as described above is different from the ITLM Administration server identification used to determinate the identity of the ITLM Administration server to IBM when submitting software usage information for those Clients using the Sub Capacity licensing model.

The following components are defined in a IBM Tivoli License Manager environment for which naming conventions must be created. The naming convention examples using the naming policies described above are also listed:

- “Organization” on page 193
- “Divisions” on page 193
- “IBM Tivoli License Manager Administrators” on page 193
- “Nodes” on page 194
- “Application users” on page 195
- “Procured Licenses and Contracts” on page 195

Organization

Organization naming should represent the name of the enterprise (or business unit) for which the Organization is being created. The following are examples using the naming policies described above:

A single enterprise wide Organization:

Name	SampleEnterprise-org
Organization Code	95127_01_SampleEnterprise-org

Multiple Organizations for the same enterprise:

Name	SampleEnterprise.HumanResources-org
Organization Code	95127_01_SampleEnterprise.HumanResources-org
Name	SampleEnterprise.Finance-org
Organization Code	95127_01_SampleEnterprise.Finance-org

Divisions

Division naming should represent a specific level of the enterprise (or business unit). The following are examples using the naming policies described above:

- Finance-div
- Finance.Credit-div
- HumanResources-div
- HumanResources.Benefits-div

In case Divisions are being defined using CSV or XML formatted files, the unique identifier must be provided. For example, 95127_01_Finance-div can be used as a unique identifier for Division Finance-div.

IBM Tivoli License Manager Administrators

As described in 5.2.3, “Administrator’s authentication options” on page 106, IBM Tivoli License Manager provides two methods of authenticating the user credentials provided by the IBM Tivoli License Manager Administrator at login time:

- The default authentication method using the ITLM Administration server database.
- User ID credentials of IBM Tivoli License Manager Administrators stored in Lightweight Directory Access Protocol (LDAP).

Regardless of the authentication method of choice, all user ID credentials must be defined in the ITLM Administration server database for authorization purposes.

In the case of using a LDAP infrastructure, names of IBM Tivoli License Manager Administrators defined in LDAP must match names defined in the ITLM Administration server database. In this case, the naming convention for IBM Tivoli License Manager Administrators must adopt the current naming convention for other types of Administrators in the enterprise.

Nodes

Nodes are defined automatically by IBM Tivoli License Manager whenever information is uploaded from the ITLM Runtime server to the ITLM Administration server with information gathered by the ITLM Agent. Nodes can also be defined manually using a CSV or XML formatted file, but *not* through the IBM Tivoli License Manager administrative Web interface.

Naming conventions for Nodes are related to two entities:

Name	A unique key within the enterprise. Once defined, it is not possible to change the Node Name information. It is this value that the ITLM Agent deployment process attempts to match when it looks for a Node entry in the ITLM Administration server database as the target Node for the deployment. If it does not find a match, a new entry is created.
Node Tag	Unique identification of the Node Tag (also known as computer label) where the Node is installed.

When Nodes are defined automatically, IBM Tivoli License Manager uses the following naming convention by default:

Name	The serial number of the machine
Node Tag	The hardware model name appended by the Name defined above (serial number of the machine).

The Node Name is of ultimate importance, as it is used during the ITLM Agent deployment process. It must match exactly to the information provided to the ITLM Agent installation process for an association between Node and ITLM Agent to occur. In case Nodes are defined using CSV or XML files, a general recommendation is to use the fully qualified host name as Node Names.

Node Tags can be changed at any time and do not affect the ITLM Agent deployment process. The structure to be used when the ITLM Administration server assigns Node Tags during automatic creation of Node records is dictated by the nodeTag parameter in the system.properties file of the ITLM Administration server. As described earlier, the default value is determined by the hardware model name, specified by the variable %MODEL, appended by the Name, and specified by the %NAME variable. The recommendation is to either reuse the identification of an existing Asset Management tools, or the fully qualified host name.

Application users

Application users are the users that can start software applications on Nodes. Details of application users are defined and maintained in the ITLM Administration server database to allow licenses to be restricted to specific users.

Naming conventions for application users are related to the logon name defined to the user at the Operating System level and must follow the same guidelines already adopted by the enterprise.

Application user definitions must be the same as the Operating System user name known by the ITLM Agent. For the Windows Operating Systems this is computer\user name or domain\user name, where computer is the host name of the computer where the ITLM Agent is installed and domain is the Windows Domain name. For UNIX Operating Systems, this is the user name defined for the user. On both cases, the name is case sensitive.

In case application users can be imported from existing user or identity management systems using CSV or XML formatted files, a unique identifier for the application user must be provided. A general recommendation is to use the <ADMIN_ID>_<Counter>-<ApplicationUserName>, where <ADMIN_ID> is the internal ITLM Administration server identifier described above.

Procured Licenses and Contracts

Procured Licenses are defined based on procurement information of the license according to the terms and conditions of the license contract. The general recommendation for license names is to adopt the naming convention defined by the Procurement Management team.

Contracts are created and associated with a license definition. Naming conventions for Contracts refers to the contract number. The contract number must be defined in a way that the association to the license definition can be easily identified. Again, the recommendation is to adopt the naming convention defined by the Procurement Management team.

For example, if a license name for a sample application is Sample001-lic, the Contract definition would have the contract number defined as Sample001-contract, making the association between Contract and License clear.

For both Licenses and Contracts definition using CSV or XML formatted files, a unique identifier must be provided. A general recommendation is to use the <ADMIN_ID>_<Counter>-<Name>, where <ADMIN_ID> is the internal ITLM Administration server identifier described above and Name specifies the License name or contract number.

6.3 IBM Tivoli License Manager Administrators

IBM Tivoli License Manager Administrators are responsible for managing various aspects of enterprise wide License Management. IBM Tivoli License Manager uses a role based approach to administrators that allows administrative functions to be performed according to the tasks these roles are allowed to perform.

These IBM Tivoli License Manager administrators are defined in the ITLM Administration server database and have access to administrative tasks using the IBM Tivoli License Manager administrative Web interface. These IBM Tivoli License Manager Administrators *do not* relate to tasks performed using both a ITLM Administration server and ITLM Runtime server command-line interface.

6.3.1 Administrators account structure

In order to define IBM Tivoli License Manager Administrators, it is necessary to understand the roles and the responsibilities that a given administrator will endorse. Within the IBM Tivoli License Manager environment administrator, accounts are defined as follows:

- An IBM Tivoli License Manager Administrators account is composed of an user ID and password combination as well as one or more profiles.
- A profile is a combination of a role defining what tasks the account can accomplish, plus an optional privacy policy determining restrictions on what information can be visible to the account.
- Each profile is valid within an single IBM Tivoli License Manager Organization.
- The same account user ID may have different profiles for different IBM Tivoli License Manager Organizations.

Figure 6-2 on page 197 depicts the structure of an IBM Tivoli License Manager Administrator account.

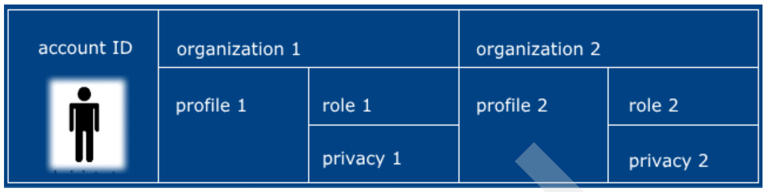


Figure 6-2 IBM Tivoli License Manager Administrator account structure

6.3.2 Administrators profiles

Defining profiles for IBM Tivoli License Manager Administrator represents the definition of an association of the IBM Tivoli License Manager Administrator to a role and a privacy policy within a particular IBM Tivoli License Manager Organization.

Roles

Roles must be associated to IBM Tivoli License Manager Administrators based on the responsibilities of individuals in the enterprise and their mission and objectives regarding the License Management solution.

The following are IBM Tivoli License Manager Administrator roles available with IBM Tivoli License Manager V2.2:

- Super Administrator

  The super administrator role is entitled to run any task for any IBM Tivoli License Manager Organization. There can be only one account with this role in any IBM Tivoli License Manager deployment. Such an account is created by default during the IBM Tivoli License Manager installation and cannot be deleted or have its profile modified. The account name is *tlmroot* and its password is set to *system*.

  The tlmroot account is the only account allowed to manage IBM Tivoli License Manager Organizations and other IBM Tivoli License Manager administrator accounts.

- Procurement Managers

  They have financial responsibility for all contracts and negotiations. They have decision making ownership about the procurement of software within the IBM Tivoli License Manager Organization. Therefore, this role is allowed to manage contracts, procure licenses, and view reports.

- License Administrators

  They are the primary users of the License Management solution. They configure software entitlements according to contracts and distribute licenses to users and machines. They also generate reports on software usage.

- Procurement and Licensing Managers

  They manage software licensing from contract management through license distribution and hold responsibility for both the proper contract information, and storing and applying the necessary software use restrictions entailed by them. They need an accurate understanding of the terms and conditions and how to translate them into the technical licensing scope. This role combines the roles of the Procurement Managers and License Administrators.

- Software Resources Managers

  They usually require information to support technology decisions and funding requests. Therefore, this role is responsible for reporting on software use.

- System Resources Managers

  The IBM Tivoli License Manager Administrator with System Resources Manager role is responsible for creating and managing the infrastructure in which IBM Tivoli License Manager will operate for a particular IBM Tivoli License Manager Organization. They are responsible for defining Divisions, Nodes, application users, and ensuring that ITLM Runtime servers and ITLM Agents are deployed, as well as defining the mapping of complex products.

- Administrators

  They are assigned the rights of all the other available roles, except for the Super Administrator. Therefore, they are allowed to run any task except to manage IBM Tivoli License Manager Organizations and IBM Tivoli License Manager Administrators.

IBM Tivoli License Manager provides a hierarchical model for IBM Tivoli License Manager Administrator account roles, as shown in Figure 6-3 on page 199.

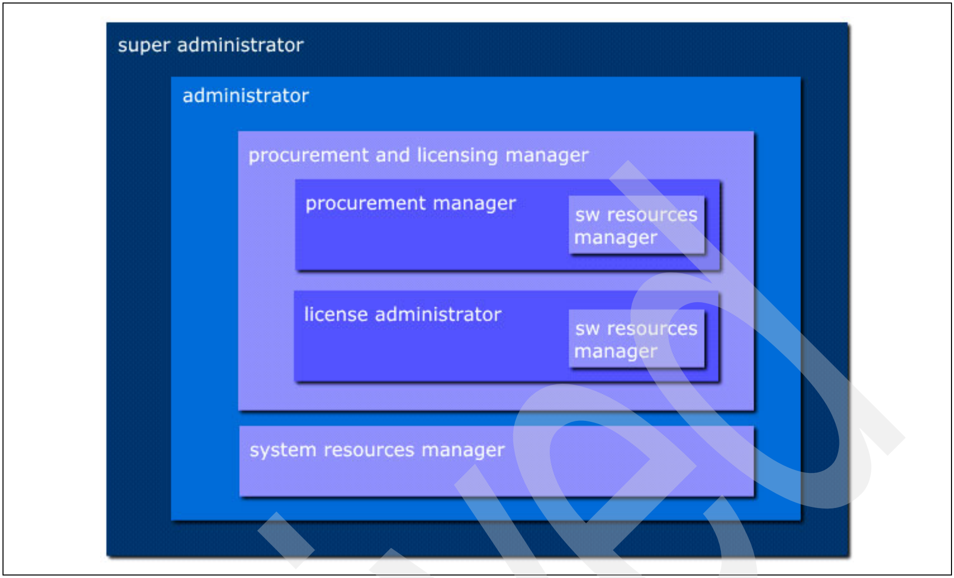


Figure 6-3 IBM Tivoli License Manager Administrator roles hierarchy

Table 6-1 provides a summary of the activities an IBM Tivoli License Manager Administrator can perform using the IBM Tivoli License Manager administrative Web Interface based on its assigned role.

Table 6-1 IBM Tivoli License Manager administrator roles & tasks relationship

IBM Tivoli License Manager Administrator Roles and activities	**Procurement Managers**	**License Administrators**	**Procurement and Licensing Managers**	**Software Resources Managers**	**System Resources Managers**	**Administrators**	**Super Administrator (tlmroot)**
Produce Reports	X	X	X	X		X	X
Manage Batch Reports	X	X	X	X		X	X
Manage Procurement	X		X			X	X
Assign Licenses		X	X			X	X
Define Product Properties		X	X			X	X
Schedule Software Scans					X	X	X
Manage Resources					X	X	X
Manage Complex Products					X	X	X
Manage Infrastructure					X	X	X
Manage Organizations							X
Manage Access							X
Define Custom Fields	X		X			X	X
Export IBM Use		X	X		X	X	X

Privacy policy

IBM Tivoli License Manager gathers information about software usage and software inventory on a computer, associating it with the user who logged on to the computer. The IBM Tivoli License Manager administrative Web interface enables the super administrator to manage how this information can be viewed, and by who. If this type of information gathering is contrary to local information privacy regulations, it is possible to configure IBM Tivoli License Manager not to collect this information. If this type of information can be collected but not accessible for certain type of IBM Tivoli License Manager administrators, it is possible to set privacy policies to the IBM Tivoli License Manager administrator account so that privacy policies can be applied.

If the privacy policy setting is active for the IBM Tivoli License Manager administrator account, the search criteria is changed to restrict user related information about the computer. Data for reports is restricted to information stored in the Products table and only information at Division level is displayed.

6.4 Nodes

An IBM Tivoli License Manager Node is a physical asset representation in an IBM Tivoli License Manager infrastructure. It can be any workstation or server of an organization. In order for an Node to be monitored by IBM Tivoli License Manager, at least one ITLM Agent needs to be deployed on it. Logical partitioning (LPAR) capable Nodes can host multiple ITLM Agents.

ITLM Agents must be subscribed physically to one ITLM Runtime server and logically to one ITLM Division. ITLM Agents are identified to the ITLM Administration server using a Node definition.

When an ITLM Agent registers itself to the ITLM Runtime server, a unique identifier must be provided. This identifier is referred to as Node Name. The Node Name information must be unique and very often based more on business information normally linked with financial information, asset tag, or a cost center than on technical characteristics.

Nodes are defined automatically by IBM Tivoli License Manager whenever information is uploaded from the ITLM Runtime server to the ITLM Administration server with information gathered by the ITLM Agent. Nodes can also be defined manually using CSV or XML formatted file, but *not* through the IBM Tivoli License Manager administrative Web interface. The Node Name is of ultimate importance when using a predefined but not automated Node definition approach. The Node Name must match exactly to the information provided to the ITLM Agent installation process for an association between the predefine Node and ITLM Agent occur.

6.4.1 Virtualization and partitioning support

In the case of logical partitioning (LPAR) capable Nodes hosting multiple ITLM Agents it is still possible to license selected software on only a subset of the total number of processors of the Node with the enhanced IBM Tivoli License Manager V2.2 virtualization and partitioning support.

IBM Tivoli License Manager V2.2 introduces the concept of a generic virtual machine (VM) Layer. A VM Layer represents a software layer that allows running multiple instances of Operating Systems, or even other VM Layers, on the same available hardware. Each VM Layer can run on a Node or on another VM Layer and are capable of running one or more VM Layers or one Operating System instance in it.

Each VM Layer has a set of processing resources for its purposes, known as total capacity, and can use subsets of its resources to implement capacity for nested VM Layers.

The VM Layer Abstraction is used to represent the following types of partitioning and virtualization elements:

- Node

  Represents the whole physical machine.
- Shared Pool

  Represents a group of processors shared among LPARs.
- LPAR

  Represents an autonomously operating portion of the Node.
- VM

  Represents the Operating System emulation layer.

Figure 6-4 on page 203 depicts generic partitioned machine topology examples. The path in the Topology that goes from the Node to an Operating System instance is know as the Operating System virtualization stack.

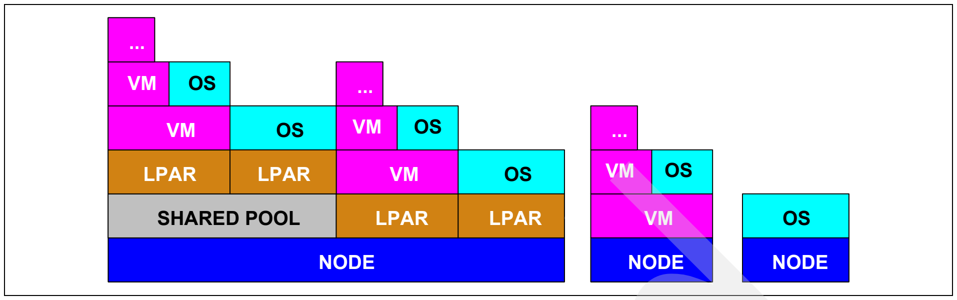


Figure 6-4 IBM Tivoli License Manager Operating System virtualization stack

6.5 Procurement management

In IBM Tivoli License Manager terms, procurement management relates to the creation of acquisition information related to software applications. It involves creating contracts, licenses definitions, and then assigning license definitions to a contract.

As described in 6.3, "IBM Tivoli License Manager Administrators" on page 196, contract and license information is created by the Procurement Manager and this information is then managed by the License Administrator. Both tasks can be performed by the IBM Tivoli License Manager administrator with the Procurement and Licensing Manager role.

IBM Tivoli License Manager's procurement management involves the following activities, as seen in Figure 6-5 on page 204:

- Create Software Contracts.
- Define Software Licenses.
- Associate licenses definitions to a contract.
- Assign software products to a license definition.
- Perform license assignment related tasks, such as defining the target type for the license (granting access to software licenses to specific Division, ITLM Agents, Nodes, and application users), as well as defining software licenses distribution quotas.
- Generating and analyzing license compliance reports.

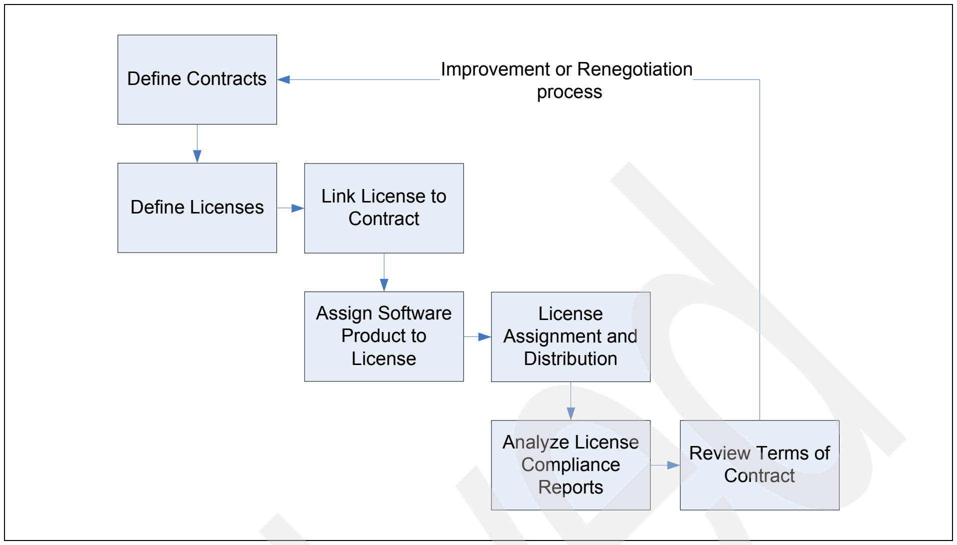

Define Contracts
Improvement or Renegotiation process
Define Licenses
Link License to Contract
Assign Software Product to License
License Assignment and Distribution
Analyze License Compliance Reports
Review Terms of Contract

Figure 6-5 IBM Tivoli License Manager procurement management tasks

IBM Tivoli License Manager V2.2 allows for contracts and licenses to be defined independently. However, to effectively manage license compliance, contracts must be assigned to a defined license, and licenses must be linked to a valid software product signature entry in the ITLM Master Catalog. As seen in Figure 6-6 on page 205, these associations are essential so that licenses are usable upon license distribution, and that software usage and license compliance reports are accurately analyzed.

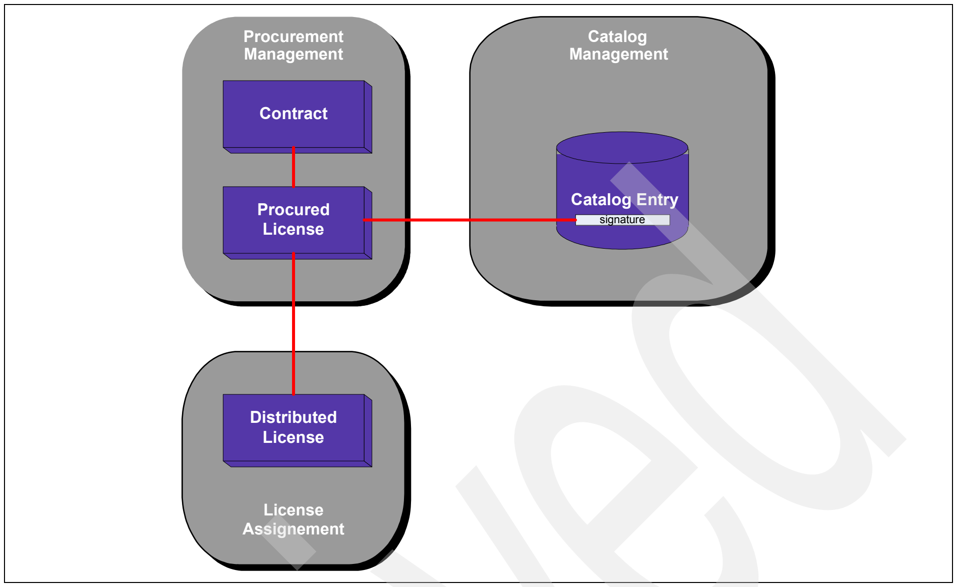


Figure 6-6 Contracts, licenses, and products relationship

The software product signature for a specific product or version of a product must be known and registered within the ITLM Master Catalog before it can be managed. When this signature is known and registered, it can be linked to the license within IBM Tivoli License Manager for management. This will be further described in 6.8, “Software Catalog management” on page 219.

The following sections go into detail about planning for contract and licenses definitions:

- 6.5.1, “Contracts” on page 205
- 6.5.2, “Procured licenses” on page 206
- 6.5.3, “License assignment and distribution” on page 208

6.5.1 Contracts

IBM Tivoli License Manager offers the ability to create a contract data repository that can be used as an alternative to trace procurement of software applications. IBM Tivoli License Manager can be used to create contracts and link them directly to the license definitions of a particular software application. A certain number of standard fields are available within IBM Tivoli License Manager. They correspond to the main contract business and legal information sections.

IBM Tivoli License Manager also provides a set of custom fields that could be used to store contract specific information. Refer to *IBM Tivoli License Manager Administration*, SC32-1430 for details on the available custom fields for contract definitions.

Defining contracts is not a mandatory action for enabling license compliance analysis. Compliance analysis is possible by enabling software monitoring and licenses definition distributions. However, defining contract information helps to have a complete view of procurement information related to License Management.

In case the enterprise uses a variety of tools for contract management, IBM Tivoli License Manager enables creation of a centralized contract data repository by allowing the import of contract definitions using XML formatted files.

Identify and improving existing software procurement process adopted by the enterprise is vital to achieving procurement management with IBM Tivoli License Manager.

6.5.2 Procured licenses

In IBM Tivoli License Manager, licenses are defined to facilitate the procurement of software application. Licenses are defined according to the terms and conditions of the license contract.

A certain number of standard fields for license definition are available within IBM Tivoli License Manager. They correspond to the main license related business and legal information sections. IBM Tivoli License Manager also provides a set of custom fields that could be used to store licenses specific information. Refer to *IBM Tivoli License Manager Administration*, SC32-1430 for details on the available custom fields for license definitions.

In IBM Tivoli License Manager, licenses can be of two categories:

- Use licenses
- Install licenses

Use licenses

These are licenses with which licenses of software application are measured by software usage. IBM Tivoli License Manager V2.2 defines the following types of use licenses, including those based on the IBM Program License Agreement (IPLA).

- Usage Concurrent Session

  This license type counts multiple instances of the software application running concurrently on the same ITLM Agent by the same application user as one used license.

- Usage Concurrent Nodelock

  This license type counts multiple instances of the software application running concurrently on the same ITLM Agent as one used license, and does not take application users into account.

- IPLA Full Capacity

  This license type is applied to IBM software using the IBM capping rule following the full capacity counting rule.

  The full capacity counting rule states that the processor quantity used by an IBM software running on a Node is the total number of processors on that Node.

- IPLA Sub-Capacity

  This license type is applied to IBM software using the IBM capping rule following the sub-capacity counting rule.

  The sub-capacity counting rule states that the processor quantity used by an IBM software running on a logical partition (LPAR) is the number of processors available to the LPAR. Refer to 6.4.1, “Virtualization and partitioning support” on page 202 for a description of IBM Tivoli License Manager support on partitioning.

Install licenses

These are licenses with which licenses of software application are measured by software installation. IBM Tivoli License Manager V2.2 defines the following types of install licenses, including those based on the IBM Program License Agreement (IPLA).

- Install Instance licenses

  This license type counts multiple installation instances of the software application on the same ITLM Agent as one used license.

- Install IPLA Full Capacity licenses

  This license type is applied to installed IBM software using the IBM capping rule following the full capacity counting rule.

  The full capacity counting rule states that the processor quantity used by an IBM software installed on a Node is the total number of processors on that Node.

- Install IPLA Sub-Capacity licenses

  This license type is applied to installed IBM software using the IBM capping rule following the sub-capacity counting rule.

  The sub-capacity counting rule states that the processor quantity used by an IBM software installed on a logical partition (LPAR) is the number of processors available to the LPAR. Refer to 6.4.1, “Virtualization and partitioning support” on page 202 for a description of IBM Tivoli License Manager support on partitioning.

6.5.3 License assignment and distribution

At license definition time, IBM Tivoli License Manager automatically assigns the complete quota licenses to the default target type based on the license type. The default target type for all license types in IBM Tivoli License Manager V2.2 is Organization.

Before licenses can be distributed, they need to be assigned to an ITLM Master Catalog entry, which represents a software product signature, in order to be usable upon license distribution. In addition, IBM Tivoli License Manager allows for the definition of license usage thresholds, which represents a percentage of the number of licenses that are being used before a warning is issued.

Once licenses are created and software products assigned to them, IBM Tivoli License Manager Administrators can change the default license assignment using the IBM Tivoli License Manager administrative Web interface. Licenses can be assigned to select targets and application users using a license distribution task.

Licenses distributions activities consist of two actions:

- Defining the target type of the distribution
- Defining the license distribution quota per target type

The following are valid target types for licenses distributions:

- Organization

  Licenses will be assigned to the entire organization. These are used when enterprises make corporate-wide licenses contracts with vendors.
- Division

  Licenses are distributed to selected Divisions.
- Node

  Licenses are distributed to selected Nodes.
- ITLM Agent

  Licenses are distributed to selected ITLM Agents.

Based on the target types described above, distributions can be narrowed down to application users defined in the ITLM Administration server database.

Defining the license distribution quota to targets includes three distinct steps:

1. Assigning products to licenses definitions

   This step is vital to certifying that the license defined corresponds to a particular software entry in the ITLM Master Catalog.
2. Ensuring that license definitions have been verified.

   This is done under the Assign Licenses option on the IBM Tivoli License Manager administrative Web interface. Specifying that a license definition has been verified indicates that the license is ready for use.
3. Assigning distribution quotas

   When performing distribution of procured licenses, the IBM Tivoli License Manager Administrator can distribute the license quota defined in a license among multiple targets. In this case, each target receives a subset of the total amount of licenses defined in the procured license definition.

Case scenario example

This section provides a license distribution example base on a fictitious enterprise. This enterprise has already implemented an IBM Tivoli License Manager solution and acquired software licenses of Use type for the ABCExample software application. Figure 6-7 shows the physical and logical designs of IBM Tivoli License Manager for the fictitious enterprise.

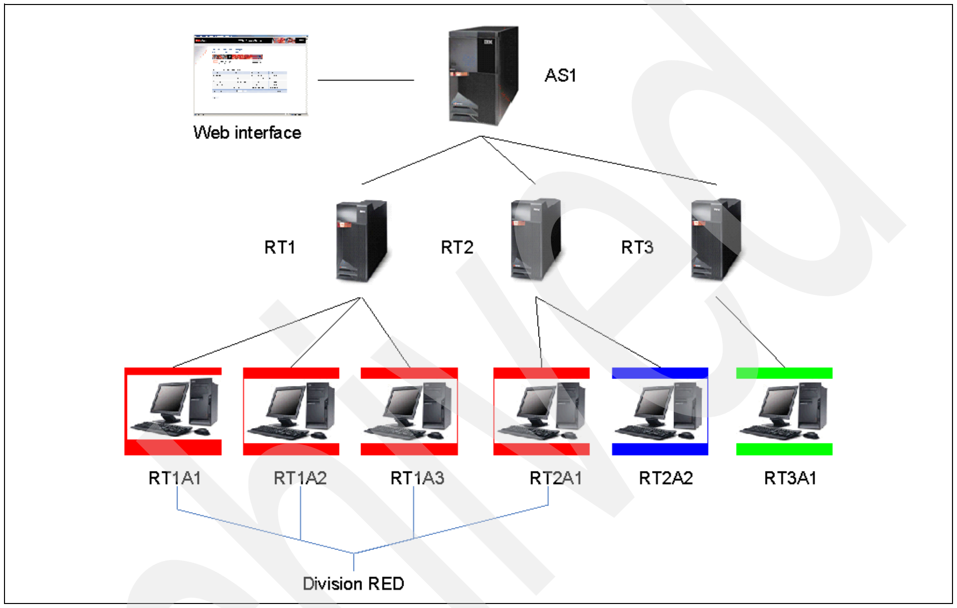


Figure 6-7 License Distribution scenario

ITLM Agents have been deployed and distributed among Divisions of a single Organization defined for the enterprise. Division RED has been defined as the target type for the distribution of ABCExample software licenses. The ITLM Agents part of the Division RED are connected to different ITLM Runtime servers. When licenses are assigned and distributed by the IBM Tivoli License Manager administrator, they are distributed according to an IBM Tivoli License Manager internal algorithm to each of the ITLM Runtime servers that have Division RED ITLM Agents connected to them. According to architecture presented in Figure 6-7, only ITLM Runtime servers RT1 and RT2 will receive the distributed ABCExample software licenses information and only ITLM Agents RT1A1, RT1A2, RT1A3, and RT2A1 will have usage of the ABCExample software application in compliance with the defined and distributed licenses.

6.6 Software usage monitoring

When you enable software usage monitoring, you allow the ITLM Agent to collect usage statistics for the product. IBM Tivoli License Manager by default does not collect any usage information about software.

As described in 6.5, “Procurement management” on page 203, contracts and licenses can be defined independently, but to effectively manage license compliance, defined licenses must be linked to a valid software signature entry in the ITLM Master Catalog. When a software signature entry is assigned to a license definition, IBM Tivoli License Manager automatically enables the usage monitoring for the software product associated to the ITLM Master Catalog entry.

Software usage information can also be enabled for software products with no license definitions, as long as a valid entry in the ITLM Master Catalog exists for the product, as seen in Figure 6-8.

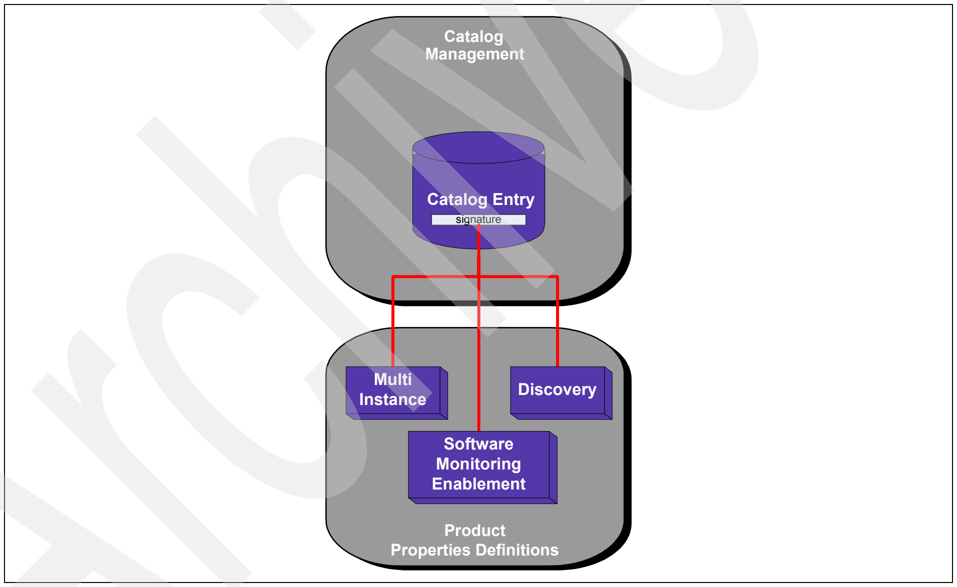


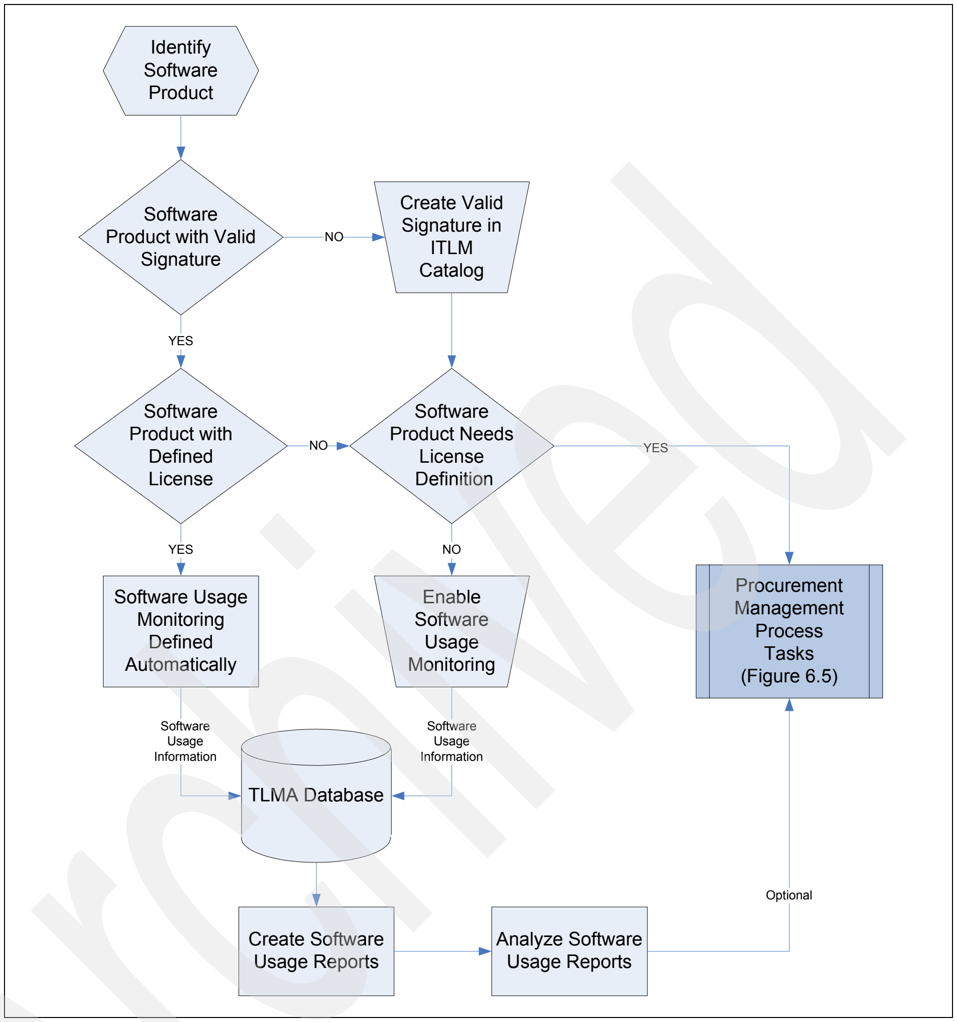
Figure 6-8 Software monitoring

Enabling software usage monitoring for a product can be done using the IBM Tivoli License Manager administrative Web interface, under the Product Properties Definitions tasks (Figure 6-8). The IBM Tivoli License Manager administrator must have one of the following roles: Administrator, License Administrator, or Procurement and Licensing Manager. For additional information, refer to *IBM Tivoli License Manager Administration*, SC32-1430.

By enabling software usage monitoring for a product, software usage information can be collected by the ITLM Agents. As described in 5.3, “Communication flow between components” on page 111, this information is then uploaded to ITLM Runtime servers and finally to an ITLM Administration server based on configuration parameters as described in 5.7, “IBM Tivoli License Manager servers settings” on page 171.

The software product signature for a specific product or version of a product must be known and registered within the ITLM Master Catalog before its usage can be managed by IBM Tivoli License Manager. When this signature is known and registered, it can be used for enabling software usage monitoring for the software products it corresponds. This will be further described in 6.8, “Software Catalog management” on page 219.

Software usage information stored in the ITLM Administration server database can then be used to generate software usage reports as well as in support of license compliance management tasks for those products with no licenses defined in IBM Tivoli License Manager, as shown in Figure 6-9 on page 213.



Identify Software Product
Software Product with Valid Signature
NO
Create Valid Signature in ITLM Catalog
YES
Software Product with Defined License
NO
Software Product Needs License Definition
YES
YES
NO
Software Usage Monitoring Defined Automatically
Enable Software Usage Monitoring
Procurement Management Process Tasks (Figure 6.5)
Software Usage Information
Software Usage Information
TLMA Database
Optional
Create Software Usage Reports
Analyze Software Usage Reports

Figure 6-9 IBM Tivoli License Manager software usage scenario

6.7 Software inventory information

Before performing software usage and procurement management tasks, IBM Tivoli License Manager needs to know what software with a matching software signature in the ITLM Master Catalog is installed on Nodes running ITLM Agents.

IBM Tivoli License Manager provides means for collecting software information from a distributed set of machines into a central repository. The central repository is the ITLM Administration server database. IBM Tivoli License Manager administrators define instructions describing the schedule and frequency of the inventory scans to be performed by the ITLM Agents.

The following topics are discussed in this section:

- 6.7.1, "Software scans considerations" on page 214
- 6.7.2, "Tivoli Common Inventory Technology" on page 215
- 6.7.3, "Software inventory information processing and upload period" on page 217

6.7.1 Software scans considerations

As the scans are being performed, results are transmitted and stored in the ITLM Runtime server database to which the ITLM Agent is connected. The ITLM Runtime server then periodically sends the information collected from all of the ITLM Agents it manages to the ITLM Administration server at scheduled times. The time and frequency that the ITLM Runtime servers upload the software inventory information to the ITLM Administration server is defined using a system.properties parameter, as described in 5.7, "IBM Tivoli License Manager servers settings" on page 171. The data flow of software inventory information is described in detail in 5.3, "Communication flow between components" on page 111.

Before the software inventory information is available for reporting, it must be available in the ITLM Administration server database. The time it takes for this information to be available depends on the number of ITLM Agents, the communication settings between ITLM Agents and ITLM Runtime servers, and the communication settings between IBM Tivoli License Manager servers. The total time is determined by the total inventory upload value for all ITLM Agents plus the communication time between ITLM Runtime servers and the ITLM Administration server.

In IBM Tivoli License Manager, software inventory scans are scheduled by Division. In order to correctly plan the software inventory scans, the maximum upload time for a Division must be known. If a software inventory scan for a Division is scheduled to occur more frequently than the maximum upload time for that Division, the information of one scheduled scan will not have finished being uploaded before the next begins. Thus, calculating and planning for the software inventory scan schedule is a crucial part of the logical design and prevents network usage bottlenecks.

As software inventory scans are performed by the ITLM Agent, it is also possible to run scans locally at the Node on which the ITLM Agent is running. This feature is particularly useful for troubleshooting tasks, or when a scheduling tool can be used to trigger the ITLM Agent to initiate the software inventory scan.

The initial software inventory scan is performed automatically and immediately after the ITLM Agent install process and subsequent scans are performed according to the inventory scan schedule. The following must be taken into account when defining the software inventory scans schedule:

- Software inventory scans should be scheduled for times when activities at the ITLM Agent's Node is low.
- Software inventory scans schedule for a Division must be set based on the maximum upload time of all Divisions.
- Time zones differences must be considered.

Only software corresponding to a software signature known by IBM Tivoli License Manager will be scanned by the ITLM Agent. Mounted file systems, temporary directories, and recycle bins are automatically discarded by the scan process.

6.7.2 Tivoli Common Inventory Technology

Tivoli Common Inventory Technology (CIT) is a component technology used to collect hardware, software, file system, and registry information from systems. Common Inventory Technology is another step towards uniting all IBM products under a single scanning technology. Common Inventory Technology includes a common scanner that provides enhanced hardware and software recognition functionality.

ITLM Agents make use of the Common Inventory Technology to perform software scans on Nodes. This section contains information about the Common Inventory technology that relates to IBM Tivoli License Manager V2.2.

Scanners

IBM Tivoli License Manager V2.2 makes use of the following Common Inventory Technology scanners:

- wscanfs

  Collects information about the file system, such as size and creation time of files on the file system.

- wscanhw

  Collects system, hardware, installed software, and file information. Installed software information is derived from collecting registry information and verifying the existence of files.

- wscansw

  Collects installed software information by evaluating a signature catalog file.

- wscanvpd

  Collects installed software information from installation registries.

Configuration files

Common Inventory Technology scanners use specific configuration files tailored for the application in which they are embedded. In IBM Tivoli License Manager V2.2, these configuration files are related to the ITLM Agents. The configuration files for the Common Inventory Technology scanners used by IBM Tivoli License Manager are located in the <ITLMAgent_install>\itlm\scanner directory, where <ITLMAgent_install> represents the ITLM Agent installation directory.

The following configuration files are used for ITLM Agent scans:

- <ITLMAgent_install>\itlm\scanner\wscanner_hw.xml

  This file configures the wscanhw scanner. The file contain configuration parameters indicating which type of hardware information will be collected by the scanner about the Node on which the ITLM Agent is running, such as Processor, Memory, Storage, and IP Address of the Node.

- <ITLMAgent_install>\itlm\scanner\wscanner_sw.xml

  This file configures the variables in the signature file and the plug-ins used when performing the scan. This file contains configuration parameters for both the wscanfs and wscansw scanners, defining parameters such as the maximum age for data to be retrieved from the cache, the time out for the query to return data, as well as directories and file systems to be included or excluded.

Software signatures file

The wscansw scanner starts a software scan using a specific signature catalog file containing signatures for software products to be retrieved. The software signature catalog file is a XML file arranged in a hierarchical structure dictated by the Common Inventory Technology.

The software signature catalog file is created by the ITLM Agent based on the information contained in the ITLM Master Catalog for products related to the Operating System on which the ITLM Agent is running. It is named signatures.xml and is located in the <ITLMAgent_install>\itlm\scanner directory, where <ITLMAgent_install> represents the ITLM Agent installation directory.

Output file

Once the software scan process is completed by the Common Inventory Technology wscansw scanner, a second software signature file is created. This file contains only software signatures found by the scanner that match a signature in the software signature catalog file (signatures.xml). This second file is named matched_signatures.xml and is created in the <ITLMAgent_install>\itlm\scanner directory.

6.7.3 Software inventory information processing and upload period

As described in 5.3, "Communication flow between components" on page 111, Software Inventory scans are scheduled on the ITLM Administration server and the schedule settings automatically flow to the ITLM Runtime servers at the time the ITLM Runtime servers request configuration updates. Likewise, the ITLM Runtime servers send inventory scan schedules to the ITLM Agents when the ITLM Agents request configuration updates.

When the ITLM Agent executes a scheduled software scan, the ITLM Agent first makes updates to the software signature catalog file, signatures.xml, based on the Operating System on which that ITLM Agent is running and known and valid software signatures stored in the ITLM Master Catalog. The ITLM Agent then uses the Common Inventory Technology wscansw scanner to perform the software inventory scan on the Node. The wscansw scanner then identifies software signatures found on the Node with signatures in the software signatures catalog file. The matching signatures are stored in the matched_signature.xml files.

The ITLM Agent processes the information stored in the matched_signatures.xml, together with other information, to generate an internal representation of the Agent Catalog, as well as several software inventory data files, and store this information in ITLM Agent cache directory. The cache directory is <ITLMAgent_install>\itlm\cache directory, where <ITLMAgent_install> represents the ITLM Agent installation directory.

Hardware information is also collected by the ITLM Agent; however, only information that is relevant to support procurement management is collected by the ITLM Agent. The ITLM Agent periodically performs updates of hardware information about the Node or partition on which it is running according to the sysConfUpdatePeriod parameter defined in the ITLM Runtime server's system.properties file.

The ITLM Agent stores hardware information in the tivhscan.mif file. This file is a Management Information Format (MIF) formatted file that contains specific information about the hardware being used on the Node or partition on which The ITLM Agent is running. The tivhscan.mif file resides in the <ITLMAgent_install>\itlm\scanner directory and consists of one or more groups containing attributes, which describe each hardware component collected by the ITLM Agent.

At predefined time intervals, the ITLM Agent sends software and hardware inventory information to its ITLM Runtime server according to the schedule of the next upload data service. The upload data service schedule is determined by the offlineUsagePeriod parameter defined in the ITLM Runtime server's system.properties file.

The software and hardware inventory information is created and maintained in the ITLM Agent cache only for the time needed to send the content to the ITLM Runtime server. The software inventory information is cleaned from the ITLM Agent cache once the ITLM Agent receives an acknowledgement from the ITLM Runtime server it has received the software inventory information. In case the ITLM Agent stops before the inventory information can be sent to the ITLM Runtime server, or is unable to communicate with its ITLM Runtime server, the software inventory information remains in the ITLM Agent cache until it can be sent again.

6.8 Software Catalog management

Software Catalog management is of vital importance to the overall IBM Tivoli License Manager solution as procurement management and software usage monitoring activities are dependent on a software signatures defined and maintained as entries in a Software Catalog. IBM Tivoli License Manager creates and maintains a Software Catalog named the ITLM Master Catalog.

Figure 6-10 provides a complete overview of the relationship between Software Catalog Management and procurement management as well as software usage monitoring of an IBM Tivoli License Manager solution.

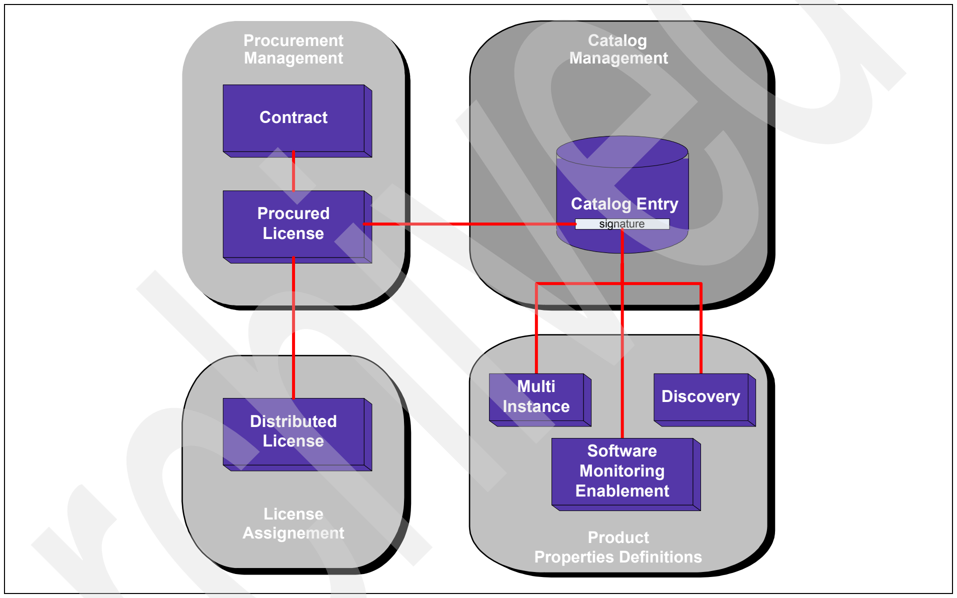


Figure 6-10 Catalog Management relationships

The following topics are discussed in this section:

- 6.8.1, “ITLM Master Catalog” on page 220
- 6.8.2, “ITLM Catalog Manager” on page 222
- 6.8.3, “Software products structure” on page 224
- 6.8.4, “Potential signatures management” on page 229

6.8.1 ITLM Master Catalog

As presented in Chapter 3, “IBM Tivoli License Manager general overview” on page 59, IBM Tivoli License Manager uses a Software Catalog, named ITLM Master Catalog, as a central repository of software products information.

The ITLM Master Catalog resides in the ITLM Administration server database. It stores data that enables IBM Tivoli License Manager to relate files and other identifiers (XSLM ID, registry entries, and so on) discovered by software scans performed by ITLM Agents to the corresponding software products. The ITLM Master Catalog contains product details that enable IBM Tivoli License Manager to determine which product releases are installed and in use on computers where the ITLM Agent is running. Entries in the ITLM Master Catalog are often referred to as software signatures.

The ITLM Master Catalog is populated initially at the ITLM Administration server installation time with information defined in the IBM Software Catalog. The IBM Software Catalog is therefore an extensive knowledge base of products and software signatures used by IBM Tivoli License Manager. There are two versions of the IBM Software Catalog:

- IBM Software Catalog for the IBM Tivoli License Manager for IBM Software product

  Contains software signatures for IBM Software only. It is meant to be used with the IBM Tivoli License Manager for IBM Software product only.

- IBM Software Catalog for the IBM Tivoli License Manager product

  Contains the entire software signatures set and is meant to be used with the IBM Tivoli License Manager product.

As new software products are being developed, signatures are added to the IBM Software Catalog. Updates to the IBM Software Catalog are made available for download from the IBM Tivoli License Manager Software Support Web Site:

`http://www.ibm.com/software/sysmgmt/products/support/IBMTivoliLicenseManager.html`

Note: The IBM Software Catalog for IBM Tivoli License Manager V2.2 has a different schema and is not backward compatible with the IBM Software Catalog for IBM Tivoli License Manager V2.1.

The entire ITLM Master Catalog resides in the ITLM Administration server database and, as described in 5.3, “Communication flow between components” on page 111, a subset of it is periodically downloaded to each ITLM Runtime server. This subset of the ITLM Master Catalog, called Runtime Catalog, only includes entries from the ITLM Master Catalog that relate to products that have been discovered on Nodes running ITLM Agents assigned to the ITLM Runtime server. The Runtime Catalog is refreshed periodically based on communication settings as described in 5.7, “IBM Tivoli License Manager servers settings” on page 171. A subset of the Runtime Catalog is also downloaded to each registered ITLM Agent. This subset, named Agent Catalog, contains software signatures information based on the Operating System platform on which the ITLM Agent runs.

Figure 6-11 shows how software signature information is distributed to the various components of the IBM Tivoli License Manager solution.

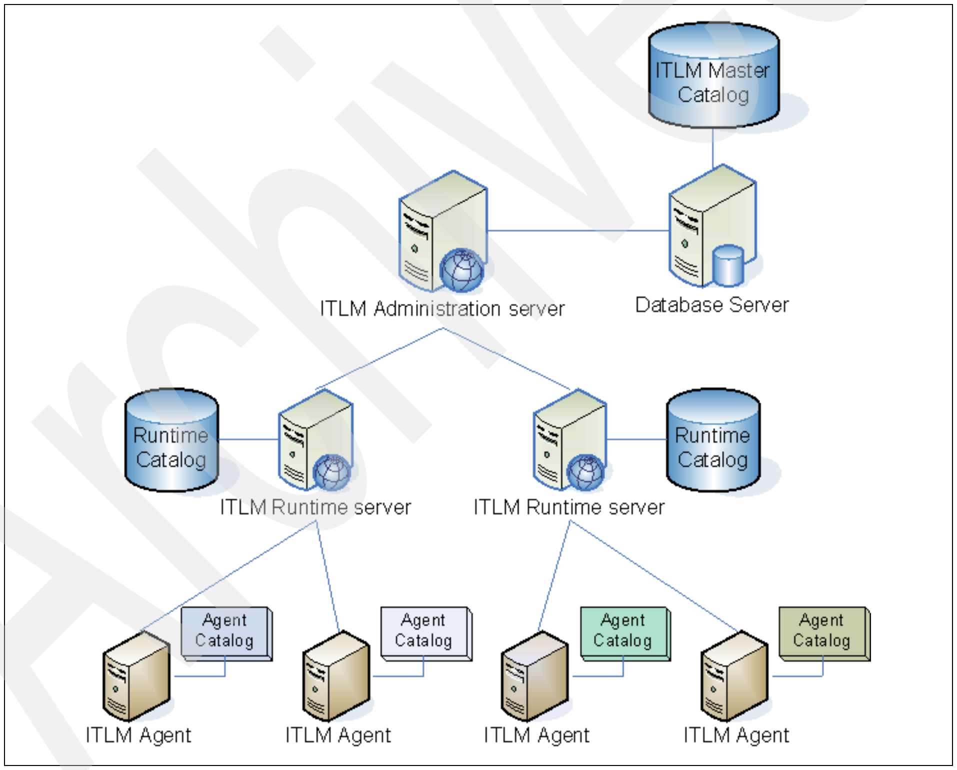


Figure 6-11 ITLM Master Catalog overview

6.8.2 ITLM Catalog Manager

IBM Tivoli License Manager provides a tool for managing the ITLM Master Catalog named IBM Tivoli Catalog Manager, or simply Catalog Manager. Using the Catalog Manager, IBM Software Catalog updates can be incorporated into the existing ITLM Master Catalog when they become available and are released by IBM.

Note: The ITLM Catalog Manager is not available in the *IBM Tivoli License Manager for IBM Software* Package.

In addition to the software signatures provided by the IBM Software Catalog, the Catalog Manager allows for expanding the ITLM Master Catalog with custom defined products and components using a hierarchical structure. This structure is described in detail in 6.8.3, “Software products structure” on page 224.

As the IBM Tivoli License Manager administrator responsible for the management of the software catalog updates the ITLM Master Catalog using the latest IBM Software Catalog, all custom defined products and components are preserved. In case any of the custom defined products has a conflicting signature in the IBM Software Catalog, the ITLM Catalog manager tool allows for the IBM Tivoli License Manager administrator to choose between its own definition or the ones in the IBM Software Catalog. Conflicts are resolved by renaming the signature by appending a *_IBM* string to the signature name.

The Catalog Manager tool also provides a means to manage the unknown software found on ITLM Agents and turns them into a recognized and manageable software signature in the ITLM Master Catalog, so that software usage monitoring can be enabled or even licenses may be defined and applied to them. The management of unknown software is described in detail in 6.8.4, “Potential signatures management” on page 229.

The Catalog Manager is a stand-alone Java application whose installation location is independent of any other IBM Tivoli License Manager component. As a general practice, the ITLM Catalog Manager tool and the ITLM Administration server are installed on separate servers (see Figure 6-12 on page 223). The common rule is to install the ITLM Catalog Manager tool on the ITLM Administrator’s workstation that is responsible for managing the ITLM Master Catalog, as long as the workstation runs a supported Operating System platform.

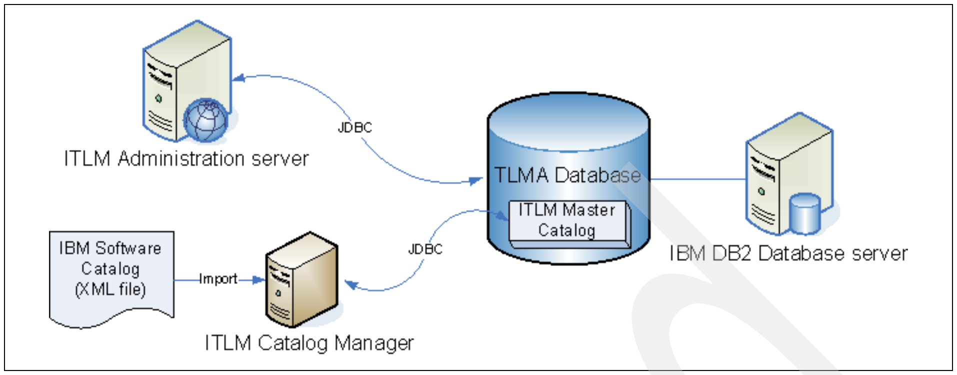


Figure 6-12 ITLM Catalog Manager system perspective

The requirement for the ITLM Catalog Manager tools are the same as for the ITLM Administration server. Refer to *IBM Tivoli License Manager Planning, Installation, and Configuration*, SC32-1431 for a complete list of supported platforms and prerequisites of the ITLM Catalog Manager.

Similarly to the ITLM Administration server application, IBM DB2 Client is no longer required, as the ITLM Catalog Manager tool accesses the ITLM Master Catalog stored in the ITLM Administration server database through a JDBC Driver of type 4, version 2.3.63. This JDBC Driver uses classes defined in the db2jcc.jar file and is shipped with the ITLM Master Catalog tool.

Access to the ITLM Administration server database requires user authentication with read/write access. A valid user ID and password credentials are requested by the ITLM Catalog manager at startup time. Most IBM Tivoli License Manager implementations use the ITLM Administration server instance owner user ID credentials. The instance owner user ID is created during ITLM Administration server installation, and the default user ID is *tlmsrv*.

Changes to the ITLM Master Catalog are applied directly to the *ITLM Master Catalog* and are made available to the IBM Tivoli License Manager administrator without the need to close the ITLM Catalog Manager or restart the ITLM Administration server.

6.8.3 Software products structure

Using the ITLM Catalog manager tool described above, it is possible to perform the management of the following entities in the ITLM Master Catalog, as shown in Figure 6-13:

- Product

  These represent software products managed by IBM Tivoli License Manager. Products can have many Components. If a component is shared among products, these products are called *complex products*. Software products are represented in the ITLM Master Catalog using a three-tier structure as follows: Product $\rightarrow$ Version $\rightarrow$ Release.

- Components

  These are the software components that make up a product. For example, ITLM Administration server, ITLM Runtime server, and the ITLM Agent are software components of IBM Tivoli License Manager. A software product can have many Components and each Component is identified by one or more Signatures.

- Signatures

  These are the entities used by IBM Tivoli License Manager to identify software products installed on Nodes on which the ITLM Agent runs. IBM Tivoli License Manager uses many different types of Signatures to accurately identify software products

Figure 6-13 shows an overview of the elements described above.

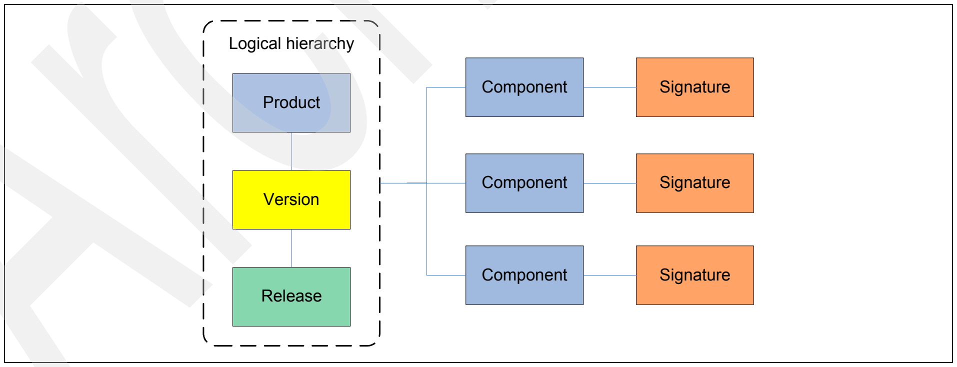


Figure 6-13 Product, Component, and Signature relationship

Software products are represented in the ITLM Master Catalog using a three-tier structure presented in the above figure: Product → Version → Release. Software products can be defined at Product and Version levels, but it is at the Release level that a physical software product is delineated in the ITLM Master Catalog.

The Product → Version → Release hierarchy identifies a software product for a specific Operating System Platform (Figure 6-14). In case software products are supported on multiple platforms, a different Product → Version → Release entry is defined in the ITLM Master Catalog per platform. In case of Java software products, the platform used is JVM. Also, the hierarchy allows for a Product to be associated with one or many Versions. Each Version can themselves have one or many Releases.

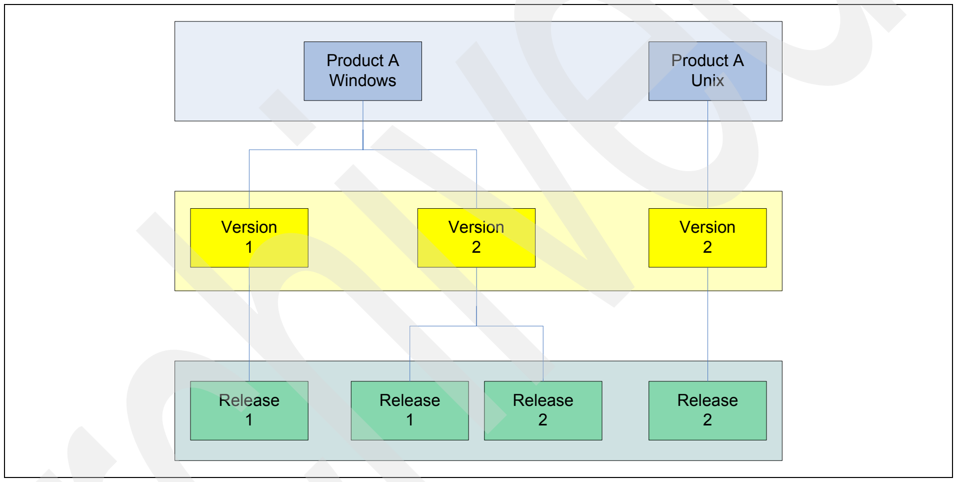


Figure 6-14 Software product hierarchy

As described in 6.8.1, "ITLM Master Catalog" on page 220, a copy of the ITLM Master Catalog is distributed to ITLM Runtime servers, which in turn send a platform tailored subset of the Runtime Catalog to the ITLM Agent. This platform tailored catalog contains Product → Version → Release information for the Operating System platform on which the ITLM Agent runs plus the JVM platform. For example, If the ITLM Agent runs on Windows Operating System, the Agent Catalog will contain Product → Version → Release entries for both Windows and JVM platforms.

Each Release of a software product can have one or multiple Components and each Component is identified by one or more Signatures (Figure 6-15). The association between a software product and a Signature is done at the Release level. This association enables the ITLM Agent to identify a Product that is installed or running on a monitored Node.

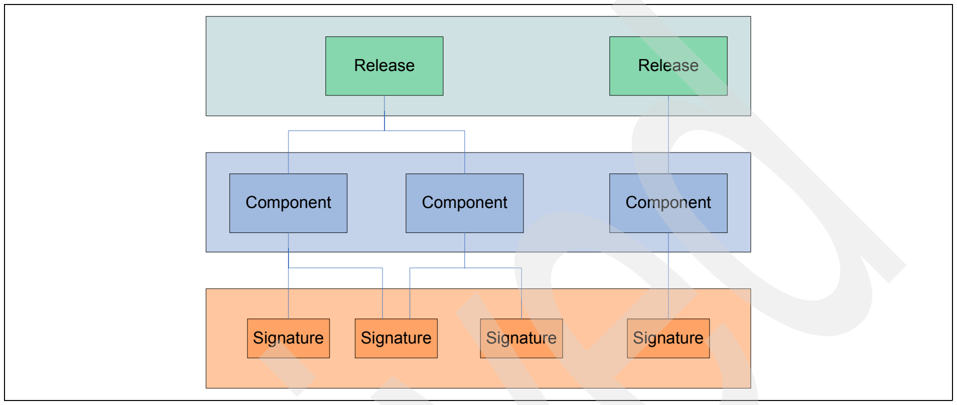


Figure 6-15 Software products, components, and signatures relationship

Signatures are entities used by the ITLM Agent to perform the following tasks:

- Recognition of installed software
- Monitoring of software usage

In IBM Tivoli License Manager, Signatures are identified by a type and a function. In terms of function, Signatures are used by ITLM Agents to either identify what is installed (recognition signatures) or running on (monitoring signatures) a monitoring Node.

A recognition signature can be linked to just one Component and monitoring signatures can be linked to multiple Releases.

Depending on the type of Signature, both recognition and monitoring functions can be performed. Table 6-2 on page 227 provides the type of Signatures used in IBM Tivoli License Manager V2.2, their respective function, and information required when creating them using the ITLM Catalog Manager tool.

Table 6-2 IBM Tivoli License Manager V2.2 Signature types and functions

Type	Description	Function	Information Required
File	Main executable file of a product, or files always installed with a specific product level.	Both	- Platform - File name - File size - Description - Version - Function
Windows Registry Key	A key recorded in the Windows registry when a software product is installed.	Recognition	- Key path - Key value - Data type
Installation Registry Key	A key recorded in the installation registry when a software product is installed with a InstallShield MultiPlatform (ISMP) wizard.	Recognition	- Platform - Product ID - Description - Modification - Manufacturer - Source (Operating System or ISMP registry)
J2EE	An application running on a JVM	Monitoring	- Name - Type (Application or Application server)
X-Open Software License Management (XSLM) ID	XSLM IDs are identifiers that IBM instrumented products pass to an API when requesting, checking, or releasing licenses.	Monitoring	n/a
Extended Signature	A more flexible type of recognition signature.	Recognition	n/a

Note: The XSLM ID and Extended Signature type described in the above table *cannot* be manually created in the ITLM Master Catalog using the ITLM Catalog Manager tool. These are only defined in the ITLM Master Catalog through importing the IBM Software Catalog.

Signatures of type File, Windows Registry Key, Installation Registry Key, and J2EE considered simple signatures as they are based on a simple string matching mechanism. The Extended Signatures type is considered complex, as their definitions are based on operands and operators. Extended Signatures operands include AND, OR, NOT, and so on, while Operators include file system expressions, such as FindFilePath, native registry expressions, such as AixProductInfo, windows file expressions such as WinFileVersionCompare, windows registry expressions such as RegKeyVersionCompare, and so on.

Figure 6-16 shows the Product → Version → Release hierarchy, and relationships between Components and Signatures using the IBM Tivoli License Manager as an example.

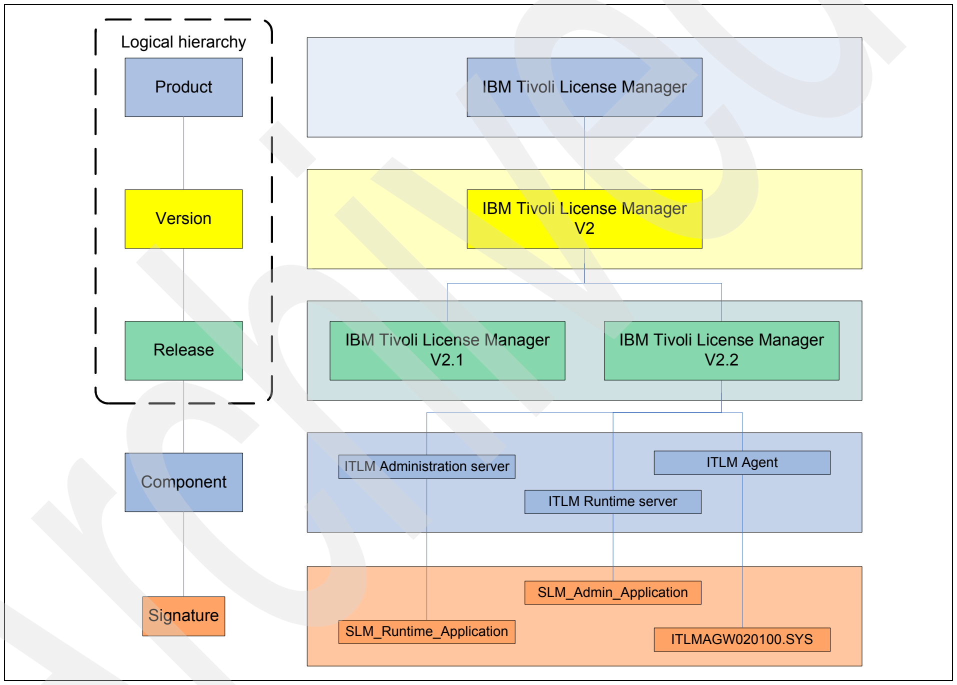


Figure 6-16 Software products definition example

For additional information about software Products, Components, and Signatures, refer to *IBM Tivoli License Manager Catalog Management*, SC32-1434.

6.8.4 Potential signatures management

IBM Tivoli License Manager also collects information about unknown software by performing software inventory scans on the monitored Nodes. As ITLM Agents perform software inventory scans, they compare the results of these scans with the software signatures found in the Agent Catalog. If there is no signature match for the newly found software product, the software product is considered unknown software.

These unknown software products are installed software discovered by ITLM Agents, which cannot be monitored by IBM Tivoli License Manager because there is no corresponding software signature for the products in the ITLM Master Catalog. These unknown software entries collected by the ITLM Agent with no correspondent signature in the ITLM Master Catalog are named *potential signature* in IBM Tivoli License Manager.

As described in 5.3, “Communication flow between components” on page 111, The ITLM Agent sends potential signatures’ information to the ITLM Runtime server at scheduled intervals, based on the configuration in the ITLM Runtime server’s system.properties file. The ITLM Runtime server compiles information from all ITLM Agents and consolidates them into the UNKNOWN table in the ITLM Runtime server database. At predefined intervals, the ITLM Runtime server sends the consolidated content of the UNKNOWN table contents to the ITLM Administration server according to the filtering options defined in the ITLM Runtime server’s unknownFiles.properties. The ITLM Administration server consolidates information from all ITLM Runtime servers and creates a master potential signatures collection into the UNKNOWN table of the ITLM Administration server database.

For each unknown software product found by the ITLM Agents, a potential signature entry is created in the ITLM Administration server database and includes all the identifying information for the unknown software, such as the name of the executable, size, and product description.

Using the ITLM Catalog Manager tool, it is possible to identify installed products whose signatures are missing from the ITLM Master Catalog. It is possible then to use the information about each potential signature to create a valid software signature for the software product components so that their usage can be monitored and controlled.

6.9 Reports

IBM Tivoli License Manager V2.2 reporting provides facilities and information to support the procurement management process.

IBM Tivoli License Manager reports help answer questions like the following:

- Which software products are installed on computers at a specific time?
- Which software products have been used within a specified time frame?
- What is the usage trend for a software product within a specified time frame?
- What is the license use trend for a software product within a specified time frame?
- What are the most used software products in the enterprise?
- What is the license compliance and availability status for a software product?
- What is the unlicensed status of software products in terms of usage and installation?

IBM Tivoli License Manager reporting relies heavily on the information collected by the ITLM Agents, communication settings, configuration parameters settings related to the type and frequency of the exchange of information between the components (specially software inventory and monitoring data exchange), and privacy policies in effect. Another factor that influences the content of IBM Tivoli License Manager reports is the content of the ITLM Master Catalog, accuracy of licenses definitions, and software monitoring settings for software products.

As seen in Table 6-1 on page 200, all IBM Tivoli License Manager administrators, except the ones assigned to the System Resource Manager role, are able to generate reports using the IBM Tivoli License Manager administrative Web interface. The information collected by IBM Tivoli License Manager at the time of report generation is dependant on the privacy policy settings defined for the IBM Tivoli License Manager administrator generating the report.

IBM Tivoli License Manager administrators can generate and browse them online or produce them asynchronously via a batch process. When IBM Tivoli License Manager administrators generate a report using the batch process, the request is queued. Then a low-priority thread consumes the queue, and processes the report. IBM Tivoli License Manager administrators can then use the batch report management facility to verify when the report has been completed. The batch report processing method makes the report available in XML formatted files. This method is specially effective for complex, time, and computing resources consuming reports. It is not possible to schedule the generation of reports using the batch process. The produced report can then be downloaded and viewed using traditional XML readers or other Reporting tools.

The following ITLM Administration server system.properties parameters affect report generation:

queryTimeout	The maximum time available for retrieving report data for an online report.
queryMaxRows	The maximum number of rows that can be included in a report.

Refer to Appendix A, “IBM Tivoli License Manager V2.2 servers system configuration file” on page 239 for a description and details on ITLM Administration server system.configuration parameters.

Historical reports of software inventory, software usage, and license compliance information are available from a set of preconfigured reports, as discussed in the following sections.

6.9.1 Software inventory report

Software inventory information is obtained by generating the Installs Snapshot report.

Installs Snapshot report

The Installs Snapshot report provides a detailed view of software products installed on computers running the ITLM Agent at a specified date and time.

The following steps and information are required to produce the Installs Snapshot report:

- One, a group, or all software products needs to be selected.
- The software product hierarchy (Product, Version, and Release) must be selected to define how much of the hierarchical tree for software products should be displayed in the report.
- The ITLM Agents to be included in the report. One, a group, or all ITLM Agents can be selected. The selection can be done also at the Division level, where one, a group, or all Divisions can be selected.
- The period of time that the report should cover.
- The level of detail in which the report will be presented. This can be a full or summary report.
- How the information in the report must be organized. Information can be grouped by software product or by ITLM Agent.

If the generated report has the detail level set to full, it is possible to drill down and obtain further installation information. For example, if the report is organized by ITLM Agent, it is possible to view detailed information about one or more ITLM Agents. This will provide a graphical representation of the software products installed on the selected ITLM Agents. It is possible to expand the presented structure to view properties for the products and components installed on the ITLM Agent.

6.9.2 Software usage reports

Software products usage information is obtained in IBM Tivoli License Manager V2.2 by generating the following reports:

- Product Use Trend report
- Product Use Level report

Product Use Trend report

The Product Use Trend report provides a graphical representation of trends in the use of selected software products during a specific period of time.

The following steps and information are required to produce the Product Use Trend report:

- Only one software product needs to be selected.
- The software product hierarchy (Product $\rightarrow$ Version $\rightarrow$ Release) must be selected to define how much of the hierarchical tree for software products should be displayed in the report.
- The Divisions to be included in the report. One or all Divisions can be selected. This report provides software usage trends at the Division level only. It is not possible to drill down to ITLM Agent level.
- The period of time that the report should cover.

After the report has been generated, it is possible to drill down and obtain further software usage trend information for the selected software product. It is also possible to change the graphical format of the report and to export its content into a XML formatted file.

Product Use Level report

The Product Use Level report provides a representation of the level of software products usage during a specific period of time.

The following steps and information are required to produce the Product Use Level report:

- One, a group, or all software products need to be selected.
- The software product hierarchy (Product $\rightarrow$ Version $\rightarrow$ Release) must be selected to define how much of the hierarchical tree for software products should be displayed in the report.
- The Divisions to be included in the report. One or all Divisions can be selected. This report provides software usage trends at the Division level only. It is not possible to drill down to the ITLM Agent level.
- As this report provides information to analyze software product usage levels, the analysis can be based either on a high water mark (HWM) or average use basis. In addition, a threshold must be provided in order to properly set the level of the analysis.
- The period of time that the report should cover.
- The report can be presented by software product name or by the basis of the analysis (HWM or average use).

An important mention for this report is that only products that have either license definitions or monitoring enabled will be displayed. IBM Tivoli License Manager by default does not monitor software product usage. Refer to 6.6, “Software usage monitoring” on page 211 for details on enabling software usage monitoring.

6.9.3 Software license compliance reports

Software license compliance information is obtained in IBM Tivoli License Manager V2.2 by generating the following reports:

- License Use Trend report
- Unlicensed Use report
- License Compliance report

License Use Trend report

The License Use Trend report provides a graphical representation of trends in the use of selected licenses during a specific period of time.

The following steps and information are required to produce the License Use Trend report:

- Only one defined license needs to be selected. The license does not need to be associated with a contract for selection.
- The period of time that the report should cover.

After the report has been generated, it is possible to drill down and obtain further license usage information for the selected license. The information displayed is the daily use value of the unit defined for the license type of the selected license. It is also possible to change the graphical format of the report and to export its content into a XML formatted file.

Unlicensed Use report

The Unlicensed Use report provides a representation and details of products being used or that are installed without licenses.

The following steps and information are required to produce the Unlicensed Use report:

- One, a group, or all software products needs to be selected. As there is no search function enabled, it is possible to limit the report by software product name matching using part of the software product name preceded, followed, or enclosed by the wildcard characters (%).
- The software product hierarchy (Product → Version → Release) must be selected to define how much of the hierarchical tree for software products should be displayed in the report. Only the software products part of the software inventory information collected by ITLM Agents will be part of the report search criteria.
- The period of time that the report should cover.
- The information displayed by the report is based on a high water mark (HWM) time range that states the amount of time, up to the end date specified, that the report should cover.

The report presents a single view in which the unlicensed high water mark (Unlicensed HWM) column shows the peak number of unlicensed use, and the unlicensed installs (Unlicensed Installs) column shows the peak number of unlicensed software products installations. The report also shows an unlicensed use quota (Unlicensed Use Quota) column showing a percent indicator of the severity of unlicensed use by combining the information of the columns above.

License Compliance report

The License Compliance report provides a representation and details of compliance for defined licenses during a specific period of time.

The following steps and information are required to produce the License Compliance report:

- One, a group, or all licenses needs to be selected. As there is no search function enabled, it is possible to limit the report by license name or license reference matching using part of the license name or license reference preceded, followed, or enclosed by the wildcard characters (%).
- The period of time in which the delivery date for the license was set. These are normally the dates the license is usable as per the license definition. A date later that the last usable date will not be accepted.
- The license type needs to be selected. The selection may be one on all license types.
- The scope of the report in terms of content must be defined. The scope is defined to presenting information about either defined licenses or distributed licenses only.
- The period of time that the report should cover.
- The information displayed by the report is based on a high water mark (HWM) time range that states the amount of time, up to the end date specified, that the report should cover.

The report presents a view in which the following attributes of the selected licenses or license distributions are displayed: License reference, license name, license type, quantity, the license unit type, amount of used units, and a percentage of available units.

The report also allows for drilling down to view details of software product use associated with a license definition.

Part 3

Appendixes

A

IBM Tivoli License Manager V2.2 servers system configuration file

This appendix provides information about definitions and parameters defined in the system.properties of IBM Tivoli License Manager servers.

ITLM Runtime server system.properties settings

This section provides a description of parameter settings in the system.properties file of the ITLM Runtime server.

The ITLM Runtime server system.properties file is located in the <ITLM_Install>/runtime/SLM_Runtime_Application.ear/slm_runtime.war/WEB-INF/conf directory and is divided into five distinct sections:

- Main Configuration File
- Runtime Server Settings
- Agent Settings
- Runtime and Agent Settings
- E-mail Configuration Settings

Main Configuration File section

The following list shows parameters defined in the main Configuration File section of the system.properties file of the ITLM Runtime server:

tlmVersion	Describes the version of the ITLM Runtime server.
runtimeToAdminSecurityLevel	Security level used for the communication between Runtimes and Administration server.
fipsEnabled	Enables the use of fips-approved providers for SSL/TLS communication and for cryptographic functions.

Runtime Server Settings section

The following list shows parameters defined in the Runtime Server section of the system.properties file of the ITLM Runtime server:

runtimeBaseTime	The hour of the day (from 0 to 23) used as a base time from which to calculate when to perform tasks that need to be performed at the same time or times each day.
runtimePluginPeriod	The interval between attempts to plug the ITLM Runtime server to its ITLM Administration server. It is used only if a plug-in has failed.
adminDownloadPeriod	The length of time that the ITLM Runtime server waits between requesting downloads of updated license, catalog, and topology information from the ITLM Administration server.

adminUploadPeriod	The amount of time that the ITLM Runtime server waits between uploading software inventory and ITLM Agent information to the ITLM Administration server.
maxAgentInactivity	The maximum amount of time the ITLM Runtime server waits for an ITLM Agent communication before it is considered inactive. This period must not be lower than the downloadParametersPeriod parameter defined in the Agent Settings section below.
storeUser	This parameter determines whether privacy policy for Application Users will be in effect when collecting software usage information.

Agent Settings section

The following list shows parameters defined in the Agent Settings section of the system.properties file of the ITLM Runtime server:

pingPeriod	The amount of time between communications that the ITLM Agent checks with its ITLM Runtime server.
downloadParametersPeriod	The interval between downloads of ITLM Agent configuration parameters from the ITLM Runtime server. At download time, the ITLM Runtime server also downloads software Catalog information to the ITLM Agent.
offlineUsagePeriod	The interval between uploads to the ITLM Runtime server of offline software use data. This parameter is also used for the ITLM Agent to upload software inventory information to the ITLM Runtime server.
processListPeriod	The interval between the compilation of consecutive versions of the ITLM Agent software products process list. This is performed to identify software applications start and stop events.
sysConfUpdatePeriod	The amount of time the ITLM Agent waits until it updates the basic hardware information about the Node or partition in which is running.
wasAgentCheckPeriod	The amount of time the ITLM Agent waits to ensure that the IBM WebSphere Application Server Agent is running.

uploadMinTime	The amount of time the ITLM Agent waits to upload the content of the potential signatures file to the ITLM Runtime server.
updateAgentEnabled	Determines whether the ITLM Agent self-updating feature is enabled.
updateAgentPeriod	In case the ITLM Agent self-updating feature is enabled, this parameter determines the amount of time between checks of updates to the ITLM Agent code.

Runtime and Agent Settings section

The following list shows parameters defined in the Runtime and Agent Settings section of the system.properties file of the ITLM Runtime server:

agentToRuntimeSecurityLevel	Determines the level of security to be used for communications between ITLM Agents and ITLM Runtime server. Possible settings are: Communication is not secure; Communication is secure with server authentication; Communication is the secure with both server and client authentication.

E-mail Configuration Settings

The following list shows parameters defined in the E-mail Configuration Settings section of the system.properties file of the ITLM Administration server:

smtpServer	Host name or IP address of a valid SMTP server
mailSender	E-mail sender address
mailRecipient	Optional, additional e-mail recipient

ITLM Administration server system.properties settings

This section provides a description of parameter settings in the system.properties file of the ITLM Administration server.

The ITLM Administration server system.properties file is located in the <ITLM_Install>/admin/SLM_Admin_Application.ear/slm_admin.war/WEB-INF/conf directory and is divided into four distinct sections:

- Main Configuration File
- Web UI Settings
- Administration Server Settings
- E-mail Configuration Settings

Main Configuration File section

The following list shows the parameters defined in the main Configuration File section of the system.properties file of the ITLM Administration server:

tlmVersion	Describes the version of the ITLM Administration server.
runtimeToAdminSecurityLevel	Security level used for the communication between Runtimes and Administration servers.
fipsEnabled	Enables the use of fips-approved providers for SSL/TLS communication and for cryptographic functions.

Web UI Settings section

The following list shows the parameters defined in the Web UI Settings section of the system.properties file of the ITLM Administration server:

maxSelectionListLength	Defines the maximum number of items that can be displayed in a list box on the Web interface. Probably the longest list box in use on the GUI is the list of vendors, used, for example, on the Installs History Report. Increasing this value can cause decreased performance of the Web interface.
privacyPolicyInfoAllowed	Determines whether the privacy policy statement panel must be accessible from an icon on the ITLM Administration server Web interface. This applies to users attempting to register their computers with a ITLM Runtime server.

sessionTimeout	Determines the maximum inactive time on the ITLM Administration server Web interface before a user is automatically logged off.
minimumPasswordLength	Determines the minimum required length of the password of an IBM Tivoli License Manager administrator.
secureData	Determines whether confidential transactions will be sent using the HTTP or HTTPS protocol.
securePort	Determines the server port to be used by the browser for secure communications when performing secure transactions.

Administration Server Settings section

The following list shows parameters defined in the Administration Server section of the system.properties file of the ITLM Administration server:

adminBaseTime	The hour of the day (from 0 to 23) used as a base time from which to calculate when to perform tasks that need to be performed at the same time or times each day.
cleanUsagePeriod	The interval between cleanups of the unaggregated software use database tables.
productInventoryBuilderPeriod	The interval of time between consecutive builds of the inventory on the ITLM Administration server. At this interval of time, the ITLM Administration server reconciles the installed software information collected by the ITLM Agent with the product information held on the ITLM Administration server.
maxUsageAge	The number of days that software usage data will be retained in the unaggregated software use database tables.
aggregateUsagePeriod	The interval between aggregations of the unaggregated software use database tables. The aggregation process aggregates qualifying usage information by component, division, and license, and stores it in the corresponding history tables.
maxAggregateUsageAge	The age of the usage data (in days) before it will be included in the aggregations of the unaggregated software use database tables. This setting is used to

	ensure that all the relevant data for an aggregation has arrived at the ITLM Administration server, taking into account the frequency with which it is uploaded from the agent to the ITLM Runtime server and the frequency with which it is uploaded from the ITLM Runtime server to the ITLM Administration server.
maxServerInactivity	Determines the length of time that the ITLM Administration server waits without receiving a communication from a ITLM Runtime server before it declares the server inactive.
queryTimeout	The maximum time available for retrieving report data for an online report.
queryMaxRows	The maximum number of rows that can be included in a report.
replacementPeriod	The interval that passes before changed product structures that are the result of changes imported with a new IBM Software catalog replace the existing product structures.
nodeTag	The structure to be used when the ITLM Administration server assigns Node Tags during automatic creation of Node records.

E-mail Configuration Settings

The following list shows the parameters defined in the E-mail Configuration Settings section of the system.properties file of the ITLM Administration server.

smtpServer	Host name or IP address of a valid SMTP server
mailSender	E-mail sender address
mailRecipient	Optional, additional e-mail recipient

B

ITLM Administration server database - ADM Schema

This appendix provides a list of database tables, descriptions, and columns of the ITLM Administration server database under the schema ADM.

This table can be used as reference for creating custom reports to support software usage, license compliance, and procurement management.

Table B-1 ITLM Administration server database - Schema ADM

Table Name	Description	Columns
ADMIN_CUST_REL	Maps the ADMINISTRATOR table to the CUSTOMER table (many-to-many).	ADMINISTRATOR_ID
		CUSTOMER_ID
		PROFILE_ID
		HIDE_HOST_INV
		HIDE_HOST
		HIDE_USER
		HIDE_GROUP
		LAST_MODIFIED
ADMINISTRATOR	Stores details of administration server administrators.	ID
		LOGON_NAME
		FIRST_NAME
		MIDDLE_NAME
		LAST_NAME
		PASSWORD
		EMAIL_ADDRESS
		EVENT_NOTIFICATION
		PHONE
		FAX
		EXT_ID
		LAST_MODIFIED
AGENT	Stores details of installed Tivoli License Manager agents.	ID
		HOSTNAME
		IP_ADDRESS
		VERSION
		OS_NAME

Table Name	Description	Columns
		OS_VERSION
		PLUGIN_TIME
		SCAN_TIME
		ACTIVE
		SERVER_ID
		NODE_ID
		DIVISION_ID
		CUSTOMER_ID
		CREATION_TIME
		SECURITY_LEVEL
		INV_SYNCED
		FORCE_PLUG_IN
		LAST_MODIFIED
AGENT_DELETED	Deleted agent.	ID
		HOSTNAME
		IP_ADDRESS
		VERSION
		OS_NAME
		OS_VERSION
		PLUGIN_TIME
		SCAN_TIME
		SERVER_ID
		NODE_ID
		DIVISION_ID
		CUSTOMER_ID
		CREATION_TIME
		SECURITY_LEVEL

Table Name	Description	Columns
		DELETED_TIME
AGENT_EVENT	Stores details of events at agents.	AGENT_ID
		EVENT_ID
		TIME
AGENT_INV	Provides historical software inventory of each agent.	ID
		AGENT_ID
		CUSTOMER_ID
		COMPONENT_ID
		SCAN_TIME
		SCOPE
		SCOPE_ID
		LINKS
		MI_REASON
		LAST_MODIFIED
AGT_VM_REL	Stores the relationship (session) between the last VM layer and the agent.	AGENT_ID
		VM_ID
		START_TIME
		END_TIME
BATCH_REPORT	Stores requests for batch reports.	ID
		CUSTOMER_ID
		TYPE
		REQUEST_TIME
		GENERATION_TIME
		USER_ID
		STATUS
		PARAMETERS

Table Name	Description	Columns
		REPORT_DATA
CONTRACT	Stores details of contracts.	CONTRACT_NUM
		TYPE
		STATUS
		SUPPLY_VENDOR
		PURCHASE_VENDOR
		OWNER
		START_DATE
		END_DATE
		COST
		COST_CURR
		TERMS
		NOTES
		MODIFIED_BY
		EXT_ID
		CUSTOMER_ID
		LAST_MODIFIED
CONTROL	Stores configuration information in key-value format.	NAME
		VALUE
COUNTRY	Stores country information.	ISO_CODE3
		NAME
		ISO_CODE2
		ISO_CODE4
		CURRENCY_NAME
		CURRENCY_CODE

Table Name	Description	Columns
CUST_OPT_FIELD	Maps the CUSTOMER table to the OPT_FIELD_DEF table (many-to-many). It provides the list of available custom fields for each organization.	OPT_FIELD_ID
		CUSTOMER_ID
CUSTOMER	Stores information about each organization.	ID
		NAME
		ACCOUNT_ID
		COUNTRY_CODE
		DESCRIPTION
		CHECK_PERIOD
		REQUEST_SCOPE
		LAST_MODIFIED
DIVISION	Stores details of divisions that represent the organization primary structural level.	ID
		NAME
		INV_START_DATE
		INV_RATE_TYPE
		INV_RATE_VALUE
		DESCRIPTION
		SELFUPDATE
		EXT_ID
		CUSTOMER_ID
		CREATION_TIME
		LAST_MODIFIED
DIVISION_DELETED	Deleted division.	ID
		NAME
		INV_START_DATE

Table Name	Description	Columns
		INV_RATE_TYPE
		INV_RATE_VALUE
		DESCRIPTION
		SELFUPDATE
		EXT_ID
		CUSTOMER_ID
		CREATION_TIME
		DELETED_TIME
ENDUSER	Stores details of users of the applications that are monitored by Tivoli License Manager.	ID
		LOGON_NAME
		FIRST_NAME
		MIDDLE_NAME
		LAST_NAME
		EMPLOYEE_NO
		EMAIL_ADDRESS
		LOCATION
		PHONE
		FAX
		EXT_ID
		CUSTOMER_ID
		LAST_MODIFIED
ENTITLEMENT	Details the usage rules for product licenses for which an organization has set up an entitlement.	ID
		PRODUCT_ID
		CUSTOMER_ID
		ACTIVE

Table Name	Description	Columns
		INV_STATUS
		MI_STATUS
		LAST_MODIFIED
IBM_REPORT	Contains the generated IBM reports and possibly the client comment associated to them.	ID
		GENERATION_TIME
		REPORT
		START_DATE
		END_DATE
		CUSTOMER_ID
		IS_EXPORTED
		CUSTOMER_COMMENT
INV_H_DIV	Stores inventory scan data aggregated by division.	DIVISION_ID
		PRODUCT_ID
		DATE_USAGE
		ID
		HWM
		UNLICENSED_HWM
INV_H_LIC	Stores inventory scan data aggregated by license.	LICENSE_ID
		DATE_USAGE
		ID
		HWM
INV_H_LIC_PROD	Stores inventory scan data aggregated by license and component.	LICENSE_ID
		PRODUCT_ID
		DATE_USAGE

Table Name	Description	Columns
		ID
		HWM
INV_H_PLIC	Stores inventory scan data aggregated by procured license.	PLICENSE_ID
		DATE_USAGE
		ID
		HWM
INV_H_PLIC_PROD	Stores inventory scan data aggregated by procured license and component.	PLICENSE_ID
		PRODUCT_ID
		DATE_USAGE
		ID
		HWM
INV_H_PROD	Stores inventory scan data aggregated by component.	CUSTOMER_ID
		PRODUCT_ID
		DATE_USAGE
		ID
		HWM
		UNLICENSED_HWM
LIC_TARGET_HREL	History of LIC_TARGET_REL table.	LICENSE_ID
		TARGET_ID
		START_TIME
		END_TIME
LIC_TARGET_REL	Maps license distributions to an organization or to one or more targets (divisions, nodes, or agents).	LICENSE_ID
		TARGET_ID
		LAST_MODIFIED

Table Name	Description	Columns
LIC_USER_HREL	History of LIC_USER_REL table.	LICENSE_ID
		ENDUSER_ID
		START_TIME
		END_TIME
LIC_USER_REL	Maps the relationships between licenses and end users (many-to-many).	LICENSE_ID
		ENDUSER_ID
		LAST_MODIFIED
LICENSE	Stores information about license pools.	ID
		PLICENSE_ID
		SERIAL_ID
		QUANTITY
		TARGET_TYPE
		ALL_USERS
		LAST_MODIFIED
LICENSE_H	History of LICENSE table.	ID
		PLICENSE_ID
		SERIAL_ID
		QUANTITY
		TARGET_TYPE
		ALL_USERS
		START_TIME
		START_ACTION
		END_TIME
		END_ACTION
LINK	Stores information about the user deployment of installed components.	ID
		SCOPE

Table Name	Description	Columns
		SCOPE_ID
		COMPONENT_ID
		AGENT_ID
		TYPE
		BRANCH_ID
		CUSTOMER_ID
		PRODUCT_ID
		BRANCH_DEL
		LAST_MODIFIED
MEASURE	Stores aggregated capacity data.	ID
		METRIC_ID
		TARGET_ID
		TARGET_TYPE
		QUANTITY
		START_TIME
		END_TIME
MEASURE_RAW	Stores raw capacity data.	ID
		METRIC_ID
		LAYER_HASH
		AGENT_ID
		QUANTITY
		DATE_SCAN
METRIC	Stores the available metrics.	ID
		NAME
		CAPACITY_TYPE
NODE	Stores details of systems on which agents are installed.	ID

Table Name	Description	Columns
		TAG
		PLATFORM
		NAME
		HASH
		HARDWARE_MODEL
		HARDWARE_MANUFACT U
		HARDWARE_TYPE
		LOCATION
		DESCRIPTION
		PLUGIN_TIME
		CUSTOMER_ID
		CREATION_TIME
		LAST_MODIFIED
NODE_DELETED	Table of deleted nodes.	ID
		TAG
		PLATFORM
		NAME
		HASH
		HARDWARE_MODEL
		HARDWARE_MANUFACT U
		HARDWARE_TYPE
		LOCATION
		DESCRIPTION
		PLUGIN_TIME
		CUSTOMER_ID
		CREATION_TIME

Table Name	Description	Columns
		DELETED_TIME
OID	Stores the most recent (highest) system-generated number for various tables throughout the database.	TABLE_NAME
		LAST_ID
		BLOCK_SIZE
OPT_FIELD	Stores the values of optional fields.	OPT_FIELD_ID
		TABLE_RECORD_ID
		VALUE
OPT_FIELD_DEF	Defines optional fields. Each line defines one optional field for one table.	ID
		TABLE_NAME
		NAME
		TYPE
PACKET_CONTROL	This table is used to avoid processing a specific packet coming from a specific Runtime twice.	CLIENT_ID
		SERVICE_ID
		PACKET_ID
PLATFORM_MAPPING		AGENT_PLATFORM
		COMPONENT_PLATFORM
PLIC_PROD_HREL	History of PLIC_COMP_REL table.	PLICENSE_ID
		PRODUCT_ID
		START_TIME
		END_TIME
PLIC_PROD_REL	Maps the relationships between procured licenses and components (many-to-many).	PLICENSE_ID
		PRODUCT_ID

Table Name	Description	Columns
PLICENSE	Stores details of procured software licenses.	ID
		EE_ID
		REF_CODE
		TREE_LEVEL
		SOFTWARE_NAME
		OWNER
		LICENSE_TYPE
		QUANTITY
		PURCHASE_TYPE
		COST
		COST_CURR
		DELIVERY_DATE
		START_DATE
		EXPIRATION_DATE
		CONTRACT_REF
		ORDER_REF
		LICENSE_KEY
		TC
		NOTES
		THRESHOLD
		VERIFIED
		EXT_ID
		CUSTOMER_ID
		MODIFIED_BY
		PID
		CCID

Table Name	Description	Columns
		PROTECTED_MASK
		LAST_MODIFIED
PLICENSE_H	History of PLICENSE table.	ID
		REF_CODE
		TREE_LEVEL
		SOFTWARE_NAME
		LICENSE_TYPE
		EE_ID
		QUANTITY
		DELIVERY_DATE
		START_DATE
		EXPIRATION_DATE
		THRESHOLD
		CUSTOMER_ID
		PID
		CCID
		START_TIME
		START_ACTION
		END_TIME
		END_ACTION
PROD_HINV	Inventory products table.	ID
		SCOPE
		SCOPE_ID
		COMPONENT_ID
		PRODUCT_ID
		AGENT_ID
		BRANCH_ID

Table Name	Description	Columns
		START_TIME
		END_TIME
PROD_INV		ID
		SCOPE
		SCOPE_ID
		COMPONENT_ID
		AGENT_ID
		BRANCH_ID
		PRODUCT_ID
		AGENT_INV_ID
		LINK_ID
		TYPE
		START_TIME
		END_TIME
PROFILE	Describes administrator profiles.	ID
		NAME
		DESCRIPTION
PROFILE_ACTION	Actions not allowed for administrator profiles.	PROFILE_ID
		ACTION_KEY
		ACTION_VALUE
SERVER	Stores informations about the Tivoli License Manager runtime servers.	ID
		NAME
		ADDRESS
		PASSWORD
		PASSWORD_TYPE
		VERSION

Table Name	Description	Columns
		PORT_NUMBER
		SSL_PORT_NUMBER
		FORCE_SERVICE
		DOWNLOAD_PERIOD
		UPLOAD_PERIOD
		UPDATE_PERIOD
		WARNING_TIME
		LOCATION
		CUSTOMER_ID
		ADMIN_RECOVERY
		CREATION_TIME
		AGT_SECURITY_LEVEL
		LAST_MODIFIED
SERVER_DELETED		ID
		NAME
		ADDRESS
		PASSWORD
		PASSWORD_TYPE
		VERSION
		PORT_NUMBER
		SSL_PORT_NUMBER
		FORCE_SERVICE
		DOWNLOAD_PERIOD
		UPLOAD_PERIOD
		UPDATE_PERIOD
		WARNING_TIME
		LOCATION

Table Name	Description	Columns
		CUSTOMER_ID
		ADMIN_RECOVERY
		CREATION_TIME
		DELETED_TIME
SERVER_SERVICE_TIM		
SERVICE	Provides a history of the service requests made.	SERVICE_ID
		SERVER_ID
		INSERTED
		FORCE_CHECK_TIME
		DESCRIPTION
TEMP_AGENT_INV		
		AGENT_ID
		CUSTOMER_ID
		COMPONENT_ID
		SCAN_TIME
		SCOPE
		SCOPE_ID
TEMP_BRANCH_REPLAC		
TEMP_ENTITLEMENT_M		
TEMP_INV_LICENSED_		
TEMP_INV_PRODUCT_A		
TEMP_INV_UNLICENSE		
TEMP_PROD_INV		SCOPE_ID
		COMPONENT_ID
		AGENT_ID
		PRODUCT_ID

Table Name	Description	Columns
		TYPE
		IS_I5OS
TEMP_USAGE_LICENSE		
TEMP_USAGE_PRODUCT		
TEMP_USAGE_SW_PROD		COMPONENT_ID
		PRODUCT_ID
		AGENT_ID
		START_TIME
		END_TIME
		SCOPE_ID
		USERNAME
		TYPE
		AGT_TRANSACTION_ID
TEMP_USAGE_UNLICEN		
USAGE_COMP	Stores product usage information.	ID
		COMPONENT_ID
		SCOPE_ID
		AGENT_ID
		START_TIME
		END_TIME
		USERNAME
		AGT_TRANSACTION_ID
USAGE_H_DIV	Stores usage data aggregated by component and division.	DIVISION_ID
		PRODUCT_ID
		DATE_USAGE
		ID

Table Name	Description	Columns
		HWM
		HWM_TIME
		AWM
		UNLICENSED_HWM
USAGE_H_LIC	Stores usage data aggregated by license.	LICENSE_ID
		DATE_USAGE
		ID
		HWM
		HWM_TIME
		AWM
USAGE_H_LIC_PROD	Stores usage data aggregated by license and component.	LICENSE_ID
		PRODUCT_ID
		DATE_USAGE
		ID
		HWM
		HWM_TIME
USAGE_H_PLIC	Stores usage data aggregated by procured license.	PLICENSE_ID
		DATE_USAGE
		ID
		HWM
		HWM_TIME
		AWM
USAGE_H_PLIC_PROD	Stores usage data aggregated by procured license and product.	PLICENSE_ID
		PRODUCT_ID
		DATE_USAGE

Table Name	Description	Columns
		ID
		HWM
		HWM_TIME
USAGE_H_PROD	Stores usage data aggregated by component.	CUSTOMER_ID
		PRODUCT_ID
		DATE_USAGE
		ID
		HWM
		HWM_TIME
		AWM
		UNLICENSED_HWM
VM	Stores the VM layers where the agents runs.	ID
		LAYER_NAME
		LAYER_HASH
		NODE_ID
		VM_PARENT_ID
		TYPE
		DEPTH

Abbreviations and acronyms

AIX	Advanced Interactive Executive (IBM UNIX)
API	Application Program Interface
BSI	British Standards Institution
CCMDB	Change and Configuration Management Database
CCTA	Central Computing and Telecommunications Agency
CEO	Chief Executive Officer
CFO	Chief Financial Officer
CIO	Chief Information Officer
CIT	Common Inventory Technology
CLERP	Corporate Law Economic Reform Program
CMDB	Configuration Management Database
CORBA	Common Object Request Broker Architecture
CPU	Central Processing Unit
CSV	Comma Separated Values
DB	Database
DMS	Database Managed Space
DPAR	Data-Parallel Abstraction
EIS	Enterprise Information Systems
EJB	Enterprise JavaBeans™
ERP	Enterprise Resource Planning
ESI	Edge Side Include
ESX	Engagement Simulation Exercise
FCM	Fast Communications Manager
FIPS	Federal Information Processing Standards
GUI	Graphical User Interface
HP/UX	Hewlett Packard UNIX
HTTP	HyperText Transfer Protocol
HTTPS	HTTP running under the Secure Socket Layer
HW	Hardware
HWM	High Water Mark
I/O	Input/Output
IBM	International Business Machines Corporation
ICT	Information and Communications Technologies
IIOP	Internet InterORB Protocol
IP	Internet Protocol
IPLA	IBM Program License Agreement
ISMP	InstallShield MultiPlatform
IT	Information Technology
ITAM	IT Asset Management
ITIL	IT Infrastructure Library
ITLM	IBM Tivoli License Manager
ITSM	IT Service Management
ITSO	International Technical Support Organization
ITUP	IBM Tivoli Unified Process
JCA	J2EE Connector Architecture
JDBC	Java DataBase Connectivity
JMS	Java Messaging Service
JNDI	Java Naming and Directory Interface™
JSEE	Java Secure Socket Extension
JSP	JavaServer™ Pages™

JVM	Java Virtual Machine
KB	Knowledge Base
LDAP	Lightweight Directory Access Protocol
LINUX	UNIX-like open source operating system
LOB	Line Of Business
LPAR	Logical Partitioning
LU	Logical Unit
MAC	Media Access Control
MIF	Management Information Format
MOF	Managed Object Format
MPP	Massively Parallel Processor
OGC	Office of Government Commerce
ORB	Object Request Broker
PMI	Performance Monitoring Infrastructure
QA	Quality Assurance
RAID	Redundant Array of Independent Disks
RAM	Random Access Memory
RDBMS	Relational Database Management System
RMI	Remote Method Invocation
ROI	Return Of Investment
RUP	IBM Rational Unified Process
SAM	Software Asset Management
SLA	Service Level Agreement
SLM	Service Level Management
SMP	Shared Multiprocessor
SMS	System Managed Space
SMTP	Simple Mail Transfer Protocol
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SOX	Sarbanes-Oxley
SQL	Structured Query Language
SSL	Secure Sockets Layer
SSL/TLS	Transport Layer Security over SSL
SUSE	Software und System Entwicklung GmbH (LINUX distribution)
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TLS	Transport Layer Security
UDB	Universal Data Base
UDF	User Defined Functions
UI	User Interface
URL	Universal Resource Locator
VM	Virtual Machine
VMM	Virtual Memory Manager
XML	eXtensible Markup Language
XP	Extended Performance
XSLM	X-Open Software License Management

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information about ordering these publications, see “How to get IBM Redbooks” on page 273. Note that some of the documents referenced here may be available in softcopy only.

- *Approach to Problem Determination in WebSphere Application Server V6*, REDP-4073
- *DB2 UDB V8 and WebSphere V5 Performance Tuning and Operations Guide*, SG24-7068
- *DB2 UDB V8.2 on the Windows Environment*, SG24-7102
- *DB2 UDB/WebSphere Performance Tuning Guide*, SG24-6417
- *WebSphere Application Server V6: Installation Problem Determination*, REDP-4068
- *WebSphere Application Server V6 Planning and Design WebSphere Handbook Series*, SG24-6446
- *WebSphere Application Server V6 Problem Determination for Distributed Platforms*, SG24-6798
- *WebSphere Application Server V6 Scalability and Performance Handbook*, SG24-6392
- *WebSphere Application Server V6 Security Handbook*, SG24-6316
- *WebSphere Application Server V6: System Management Problem Determination*, REDP-4067
- *WebSphere Product Family Overview and Architecture,* SG24-6963

Other publications

These publications are also relevant as further information sources:

- *IBM Tivoli License Manager V2.2 Administration*, SC32-1430
- *IBM Tivoli License Manager V2.2 Catalog Management*, SC32-1434
- *IBM Tivoli License Manager V2.2 Commands*, SC32-1501
- *IBM Tivoli License Manager V2.2 Data Dictionary*, SC32-1432
- *IBM Tivoli License Manager V2.2 Planning, Installation, and Configuration*, SC32-1431
- *IBM Tivoli License Manager V2.2 Overview*, SC32-1503
- *IBM Tivoli License Manager V2.2 Problem Determination*, SC32-9102
- *IBM Tivoli License Manager V2.2 Security Management*, SC32-1502
- Office of Government Commerce Staff, *Software Asset Management*, The Staionery Office, 2003, ISBN 0113309430

Online resources

These Web sites and URLs are also relevant as further information sources:

- IBM Tivoli License Manager V2.2 Education

  http://www-306.ibm.com/software/tivoli/education/edu_prd.html#L
- IBM Tivoli License Manager V2.2 Information Center

  http://publib.boulder.ibm.com/tividd/td/IBMTivoliLicenseManager2.2.html
- IBM Tivoli License Manager V2.2 Product Support Web site

  http://www-306.ibm.com/software/sysmgmt/products/support/IBMTivoliLicenseManager.html
- IBM Tivoli License Manager V2.2 Product Web site

  http://www.ibm.com/software/tivoli/products/license-mgr/

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

IBM

Redbooks

Implementation Best Practices for IBM Tivoli License

(0.5" spine)
0.475"<->0.875"
250 <-> 459 pages

Implementation Best Practices for IBM Tivoli License Manager

Systematic approach to implementation

Thorough solution design tasks information

Based on real large scale implementations

The primary objective of this IBM Redbook is to provide best practices for the implementation of a License Management solution based on IBM Tivoli License Manager V2.2. This redbook is divided into two main parts:

In Part 1, "Business aspects and project engagement considerations" on page 1, we discuss the reasons why you should deploy a License Management solution. We also position License Management in the overall picture of the IT Service Management and describe the IBM solution in this area. We also provide best practices on how to approach and engage such a project. The high level tasks of such a project are also introduced. We emphasize the fact that you must not only realize what the key elements are that need to be managed to make your project a success, but also how you should organize your IT operations to get the highest benefit from your solution.

In Part 2, "Solution design considerations" on page 57, we provide detailed information for the architecture and design of a IBM Tivoli License Manager V2.2 solution implementation. Both IT Architects and IT Specialist benefit from the information provided in this part, as it covers a great deal, from an introduction to IBM Tivoli License Manager V2.2, its components, and architecture, and solution design fundamental tasks, to detailed physical and logical design and configuration of all components of an IBM Tivoli License Manager V2.2 solution.

SG24-7222-00 ISBN 0738497428