Visual LANSA Administrator's Guide

- Planning Your Site
- Getting Started with Administration
- Visual LANSA Logon
- Visual LANSA Initialization
- Remote Systems
- Change Management
- System Information
- Environment Settings
- Compiler Settings
- Troubleshooting
- Appendix A. LANSA's Customer Services

Edition Date May 16, 2014 © LANSA

1. Planning Your Site

If you have not yet installed the Visual LANSA software, you should take time to properly plan and understand how the Visual LANSA software will be used at your site. The Visual LANSA development environment may be just one component of a complete LANSA System installation that involves many different LANSA Products.

For an easy to follow planning guide, go to Install Planner.

LANSA is a family of multi-platform application development tools. This section defines each of the software components that can be installed.

Because LANSA supports multiple platforms, the software can be configured to use development environments on IBM i and Windows.

Web development and Web deployment models may also be multi-tier using both IBM i and Windows.

Web development and Web deployment models may be installed in virtualized and Citrix IT infrastructure. Search in LANSA Support to view the latest:

- Citrix install instructions
- LANSA's Virtualization support policy
- LANSA's Supported Versions.

It is important to be clear about the software components and where they are required. They are described in:

• 1.1 LANSA Software Components

The various models for development and deployment are described in:

- 1.2 LANSA Development Models
- 1.3 Choosing a Development Model
- 1.4 Recommended Development Models
- 1.6 LANSA for the Web Deployment Models
- 1.7 LANSA Integrator Development/Deployment Components

Basically, LANSA Systems can be classified as:

- 1.2.4 Independent Windows Server and 1.2.1 IBM i Slave Systems
- 1.5 Promotion & Deployment, which are remote systems, for example Linux, that receive objects from a Visual LANSA development system.

If you are using a master and slave system, it is important that you understand the task of 1.2.10 Synchronizing Master and Slave Systems.

A LANSA System is defined as a group of LANSA Software Components that share a common LANSA System Definition. A single system may contain many different products. For example, a single LANSA System could be a development environment using LANSA for iSeries on an IBM i server with LANSA for the Web on a Windows server and many Slave Visual LANSA Systems on developer PCs. A LANSA for iSeries System running a different LANSA version would be a separate LANSA System because it uses a different LANSA System Definition (e.g. System Definitions for that version).

1.1 LANSA Software Components

The LANSA Software Components are the LANSA program objects that are installed on a platform. These are the programs that LANSA uses to support both the development environment and the application execution environment.

Each LANSA product has its own set of software components and may also share its software components with other LANSA products.

For example, LANSA for the Web requires the LANSA for iSeries or Visual LANSA software components to be installed.

As LANSA supports an n-tier architecture, all LANSA Software Components do not have to be installed on the same machine or platform. For example, the LANSA for the Web's Web Server Components may be installed on a different machine (Web Server) than the Visual LANSA Software Components (Data/Application Server).

Also see

1.1.1 LANSA System Definitions

- 1.1.2 LANSA Repository Data
- ↑1. Planning Your Site

1.1.1 LANSA System Definitions

The LANSA System Definition Data is the LANSA system-specific data used by LANSA software to define and support the development environment. The LANSA System Definition Data includes the following types of information:

- RDML Command Definitions
- System Message Files
- Application Templates
- Built-In Functions
- System Definition Characteristics (system owners, user authorities, machine date formats, etc.)
- Task Lists
- Enrolled Users
- Enrolled PCs.

A LANSA System Definition is primarily identified by its version number. Parts of the System Definition Data can be customized to create a specific development environment.

↑ 1.1 LANSA Software Components

1.1.2 LANSA Repository Data

The LANSA Repository data is the application-specific data entered by a developer using the LANSA Software. The developer builds a repository for their business application. The LANSA Repository includes the following types of objects:

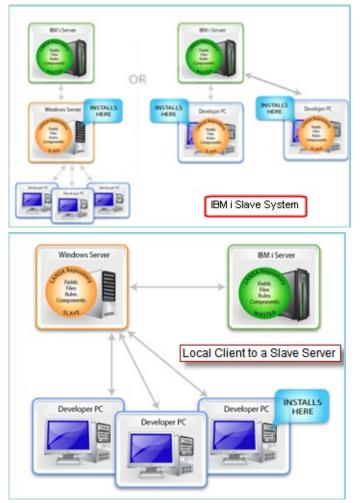
- Fields
- Files
- Processes
- Functions
- System variables
- HTML pages
- Web Components
- Forms
- Reusable parts, and so on....

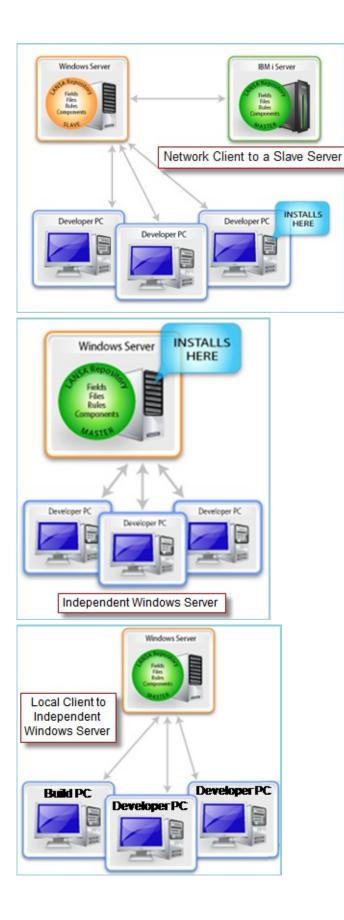
For example, in the Personnel System Demonstration, the PSLMST file and the EMPNO field are data in the LANSA Repository.

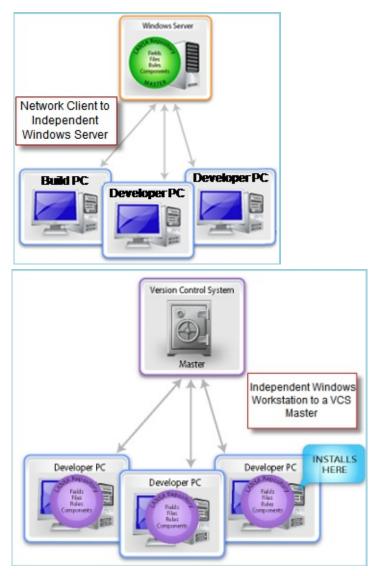
↑ 1.1 LANSA Software Components

1.2 LANSA Development Models

The following are the development models available with Visual LANSA. Click on one of the options to see more details:





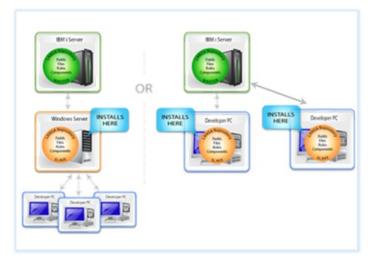


Also See

1.2.10 Synchronizing Master and Slave Systems

1.2.11 LANSA for the Web Development Models

1.2.1 IBM i Slave Systems



An IBM i Slave is a Visual LANSA System on a Windows PC connected to a Master LANSA System running on an IBM i.

Master LANSA System

A Master LANSA System is identified by its ability to maintain its LANSA System Definition Data. Master LANSA Systems are independent. The Master System Definition Data is loaded when a system is installed or upgraded. LANSA Repository data can be imported from another LANSA System. (A Master LANSA System does not use a host monitor.)

If you use LANSA for iSeries on an IBM i it is always a Master System. Each LANSA for iSeries System may have a Slave Visual LANSA System.

A Visual LANSA System on Windows can be a Master System but it does not have any Slave Systems. It can have Clients, which can use the Repository data for development. Refer to 1.2.4 Independent Windows Server for more details.

Slave LANSA System

A Slave LANSA System is not able to maintain the LANSA System Definition Data. It receives its system definition from a Master LANSA System. It is dependent upon the Master LANSA System. The System Definitions must be updated on the Slave system whenever a change is made to the Master System.

A Slave LANSA System is loosely coupled to a Master System. It works separately from the Master System, but needs to obtain permission from the Master to make modifications to LANSA objects. It also obtains the latest version of an object from the Master System. And when the changes have been made they are checked back into the Master System. Only one Windows PC at a time has authority to make changes. Once a Windows PC has obtained authority, no other Windows PC will be permitted to change it.

A Slave LANSA System exchanges Repository data with the Master System using a host monitor to check-in and check-out Repository data from the Master LANSA System. For more details about the host monitor, refer to Host Monitor in the *Administrator's Guide*.)

A Visual LANSA System can be a slave to only a LANSA for iSeries Master System. It cannot be a slave to another Visual LANSA System.

You may have multiple Slave Visual LANSA systems on different Windows PCs accessing the Master LANSA Repository on a single IBM i.

Developing with Master and Slave Systems

When developing LANSA applications with Master and Slave Systems, there can be only one Master LANSA Repository. The Master LANSA Repository typically resides on the IBM i with the Master LANSA for iSeries System.

The Visual LANSA Slave system also has a local or Slave LANSA Repository. The Slave Visual LANSA Repository contains copies of the objects from the Master Repository that are in the process of being developed or maintained using Visual LANSA. Visual LANSA uses a database management system on the workstation to store the local LANSA Repository.

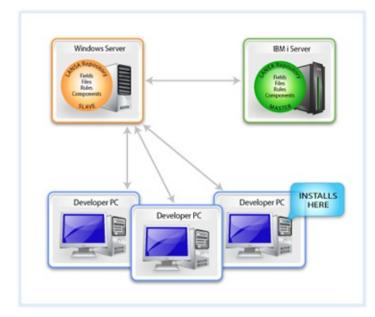
Two types of information are stored in the LANSA Repositories:

- LANSA system definitions
- LANSA application definitions.

It is very important to keep the LANSA system definitions and LANSA application definitions synchronized. LANSA provides a number of facilities to assist in this task. Refer to 1.2.10 Synchronizing Master and Slave Systems.

One of the IBM i Slaves will need to be designated as the Build Machine. See 1.2.8 Windows Build Machine

1.2.2 Local Client to a Slave Server



A **Local Client to a Slave Server** has a full LANSA system installed on the Windows PC except for the Repository which is accessed on the Windows Server (1.2.1 IBM i Slave Systems). Developers must be connected to the Slave Server.

To install this, in the installation process you choose as the *Setup Type*: **Typical or Custom Visual LANSA development environment** and select **Client to a Slave Database Server** (master is on an IBM i).

If any LANSA upgrades are needed, they must be made to each Local Client.

One of the Local Client machines, or the Windows Server will need to be designated as the Build Machine. See 1.2.8 Windows Build Machine

Windows Server IBM i Server Bess Box Server Great Server Great Server Great Server Great Server Great Building Great Developer PC Developer PC Developer PC Developer PC Developer PC Developer PC Developer D Developer PC Developer PC Developer PC Developer PC Developer PC

1.2.3 Network Client to a Slave Server

A **Network Client to a Slave Server** is a thin client that has only shortcuts installed. This means that all Repository and LANSA processes are accessed from the Windows Server. Because the network client obtains all files from the Windows Server, processing is slower than with a Local Client. With this client type, however, when any LANSA upgrades are made to the Windows Server, they are automatically also made to the Network Client.

A small number of files are installed locally for Visual LANSA to function correctly in this environment. Installation of the Microsoft C++ compiler is optional.

To install this, in the installation process select the *Setup Type*: **Shortcuts to run Visual LANSA install on another workstation**.

One of the Network Client machines, or the Windows Server, will need to be designated as the Build Machine. See 1.2.8 Windows Build Machine

1.2.4 Independent Windows Server

A Visual LANSA System is described as Independent when it is not connected to an IBM i Master System. It maintains its own LANSA System Definition Data and is the sole arbiter of object permissions. It may have one or more 1.2.1 IBM i Slave Systems linked to it. This allows the Repository data to be shared by more than one developer, but there is only one Repository.



The Deployment Tool is used to move or export your LANSA applications developed on an Independent System.

One of the Client machines, or the Windows Server will need to be designated as the Build Machine. See 1.2.8 Windows Build Machine

Build PC Deve loper PC

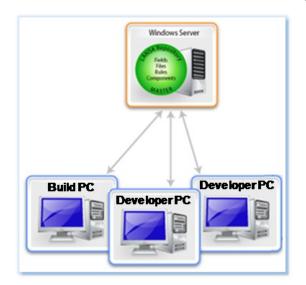
1.2.5 Local Client to an Independent Windows Server

A **Local Client to an Independent Windows Server** has a full LANSA system installed except for the Repository which is accessed on the Windows Server.

To install this, in the installation process you choose as the *Setup Type*: **Typical or Custom Visual LANSA development environment** and select Client to an Independent Database Server (Master is on a Windows workstation).

If any LANSA upgrades are needed, they must be made to each Local Client.

One of the Local Client machines, or the Windows Server will need to be designated as the Build Machine. See 1.2.8 Windows Build Machine



1.2.6 Network Client to an Independent Windows Server

A **Network Client an Independent Windows Server** is a thin client that has only shortcuts installed. This means that all Repository and LANSA processes are accessed from the Windows Server. Because the network client obtains all files from the Windows Server, processing is slower than with a Local Client. With this client type, however, when any LANSA upgrades are made to the Windows Server, they are automatically also made to the Network Client.

A small number of files are installed locally for Visual LANSA to function correctly in this environment. Installation of the Microsoft C++ compiler is optional.

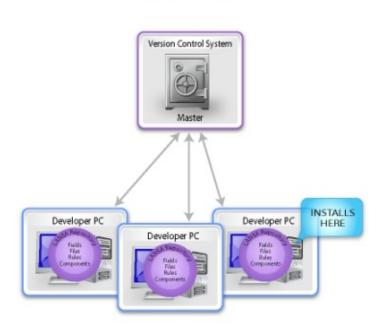
To install this, in the installation process select the *Setup Type*: **Shortcuts to run Visual LANSA install on another workstation**.

One of the Network Client machines, or the Windows Server will need to be designated as the Build Machine. See 1.2.8 Windows Build Machine

1.2.7 Independent Windows Workstation to a VCS Master

A Visual LANSA System used with as VCS Master is always an Independent Visual LANSA System. It is a single user LANSA development environment. It has a local database and repository. It is connected to other Independent Visual LANSA Systems via a VCS Master.

A VCS Master is a Version Control System of your choice. It is only required to be able to store and retrieve text files.



SELECTED SCENARIO

This environment may be compared to the Visual Studio development paradigm. Visual LANSA is editing text files that have been checked out from the VCS or is creating them to be put in the VCS. The Repository is still used to contain the LANSA objects, but they are also now reflected into a directory which the VCS also has a view of.

The Deployment Tool is used to move or export your LANSA applications developed on an Independent System.

One of the Independent Windows Workstations will need to be designated as the Build Machine. See 1.2.8 Windows Build Machine

1.2.8 Windows Build Machine

An important aspect of a development model is isolating the construction of shipped objects to a single machine. This is often called a Build Machine. Packaging the application on a single machine allows the environment to be controlled. Changes to the operating system, C++ compiler and LANSA version may all change what is deployed. Thus LANSA recommends that a Build Machine is maintained for each version of the application in the field.

There are two features of Visual LANSA that make this imperative. A version number may now be assigned to a LANSA object and the construction of an MSI produces GUIDs and a version number of the application. The GUIDs in particular are vital for linking together an upgrade of an application to a previous version. If the GUID is different the application is effectively different even though all objects and version information are identical. This is a Windows Installer restriction.

Another extremely useful feature of a build machine is that it can be setup to automatically build your application nightly and then run automated tests on the resulting application.

Its possible to have multiple systems being built on one Build Machine. LANSA recommends against this so that later versions of your application can be built on the latest supported environment - operating system and compiler, and not effect older deployed applications. You will also be able to purchase more powerful machines and be using newer hardware that's more likely to continue running for the life of your deployed application.

A useful method to still use a single machine and ensure against ageing hardware is to use a Virtual Build Machine. The virtual machine is then much more easily moved to new hardware and also allows multiple Build Machines to be running on a single piece of hardware. Given the low level of use of Build Machines this may result in little difference in performance of the build of your application.

Also See

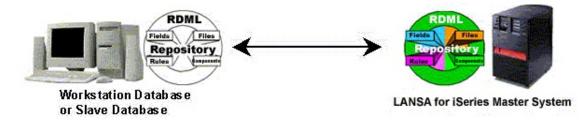
Compiler Settings

1.2.9 Master and Slave System Configurations

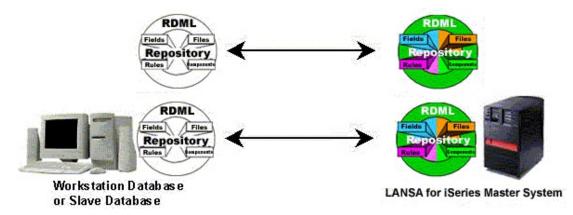
Visual LANSA uses a database management system to store the local or Slave LANSA Repository. This section defines the supported database configurations for developing with the Master and Slave systems.

Supported Configurations

A separate Visual LANSA database **must be used** for each LANSA system on the IBM i. The database may be a workstation (single user) or a server (multiuser) database. Just one Visual LANSA database should be used for one LANSA Repository on the IBM i.



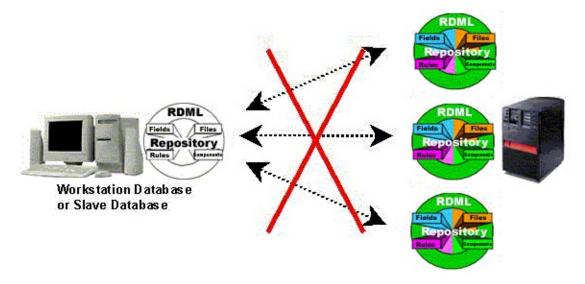
In the above configuration, a workstation or server with one Visual LANSA Repository is used to store information from one LANSA for iSeries system. More than one Slave System can store information from the same LANSA system, but there will be a separate local Visual LANSA Repository on each workstation for each LANSA system.



In the above configuration, a workstation or server database with two or more Visual LANSA Repositories is used to store information from different LANSA for iSeries systems. The LANSA systems may reside on one IBM i or on different IBM i servers.

Prohibited Configuration

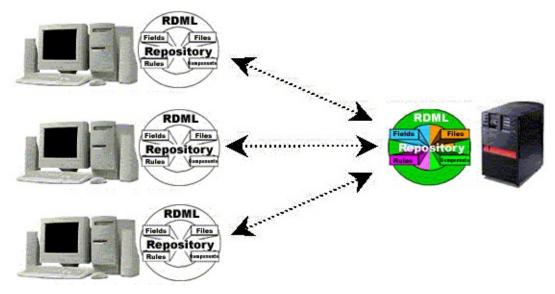
The following Master and Slave Repository configuration will cause problems and is therefore not supported:



A single Visual LANSA database **must not be used** to store information from more than one LANSA system on the IBM i. It does not matter whether the LANSA systems reside on one IBM i or on different IBM i. You must use multiple Visual LANSA Systems so that each system is separate.

1.2.10 Synchronizing Master and Slave Systems

When developing applications using Master and Slave LANSA Systems, you must keep both the LANSA System Definitions and the LANSA Application Definitions synchronized.



Synchronizing System Definitions

The system definitions of the local and master Repositories are set up using two automated procedures:

• To populate each local Repository for the first time and to periodically update the definitions, you will use the automated System Initialization and Partition Initialization facilities. These facilities are described Connect Master and Slave Systems in the *Visual LANSA Administrators Guide*.

Synchronizing Application Definitions

Application definitions are transferred between your Master and Slave using one of the following:

- Visual LANSA Check In/Check Out facilities, which are described in Host Monitor in the *Visual LANSA Administrators Guide*.
- LANSA for iSeries Housekeeping Export facility and the Visual LANSA Import facility, which are described in Transferring Fields, Files, Processes and Functions.
- LANSA for iSeries Repository Synchronization facility, which is described in *LANSA PC Development* in the *LANSA for iSeries User Guide*.

1.2 LANSA Development Models

LANSA for the Web development and operation can be done with a single-tier set up or a multi-tier configuration as described in this section.

1.2.11 LANSA for the Web Development Models

LANSA for the Web development and operation can be done with a single-tier set up or a multi-tier configuration as described in this section.

To develop LANSA for the Web applications, you will need:

- A Data/Application Server, which can be a Windows or an IBM i server. The Data/Application Server is where your LANSA Repository resides and where the LANSA application runs.
- A Web Server, which can be a Windows or an IBM i server. The Web Server can be the same IBM i as your Data/Application Server, i.e. you can use single IBM i for both the Web Server and Data/Application Server.
- A Windows PC connected to the IBM i server to run the Web Administrator.
- A Windows PC with a supported browser to test your applications.

Your application development environment may be different to your intended deployment configuration.

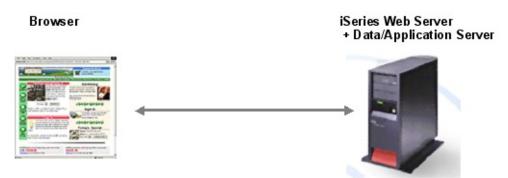
A typical multi-tier Windows-based development environment might appear as follows:



If you are running a single tier, Windows-based development environment where both the Web Server and the Data/Application Server are installed on a single PC your configuration will be similar to the picture following. This type of install is the standard, automatic install, provided by the LANSA Installation.



A typical Single-tier IBM i-based development environment might appear as follows:



Considerations for Web Development Environments

Following are some considerations for using LANSA for the Web with your LANSA development environment:

- LANSA supports the concept of logical partitions to separate your different environments in your LANSA system. For example, you might have partitions for development (DEV), testing (TST), training (TRN), production (PRD), etc.
- You do not have to use LANSA for the Web in all of your LANSA partitions.
- A LANSA for the Web application developed in a partition executes independently of applications in other partitions. The application partition is identified when the user executes the LANSA for the Web application. Refer to Uniform Resource Locator (URL) Syntax in the LANSA for the Web Guide.
- You must have both LANSA for iSeries and LANSA for the Web development licenses on the Data/Application Server in order to develop Web applications on the IBM i.

You must have both Visual LANSA and LANSA for the Web development licenses on the Data/Application Server in order to develop Web application on Windows.

LANSA for the Web Administrator

The LANSA for the Web Administrator should be installed only on the System Administrator's PC. This is both for development and deployment systems.

Web Administrator is used to:

- Configure the IIS Plug-in, Servlet, Apache Module
- Configure the Web Runtime (Windows, iSeries, Linux)
- Configure the iSeries Web Server
- Perform maintenance tasks on Web Runtime.

These are tasks that usually only a System Administrator will perform.



When using LANSA for the Web on IBM i, you will only need to install, as a separate process, the LANSA for the Web Administrator. This is described in Task: Install Other Features .

The LANSA for the Web Administrator can execute in a local mode for IIS Plug-in configuration on a Windows Web Server, or it can execute in a client/server mode to the Data/Application Server. In client/server mode, it requires a connection to the Data/Application Server. A TCP/IP connection using a listener program on the server is recommended for this.

↑ 1.2.11 LANSA for the Web Development Models

Multilingual Considerations

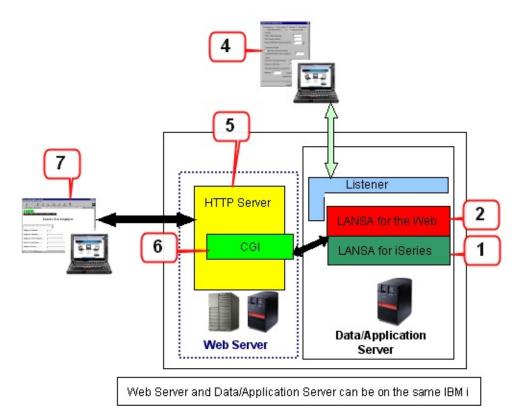
Following are some multilingual considerations if you are using the LANSA for the Web Administrator:

- If you are not using a completely US English version of LANSA, then you must be aware of code page translation issues on the IBM i.
- To enable LANSA to display messages and translate data correctly from the host system, the following two files must be converted to the appropriate language: LCOECHAR.DAT (character translation) and LCOEMESG.DAT (error message text).
- For more details, refer to Multilingual Support in the LANSA Open Guide.

1.2.11 LANSA for the Web Development Models

IBM i Software Components

When installing and configuring LANSA for the Web on IBM i, it is very important to understand the fundamentals of its architecture.



The diagram above identifies several of the key components used by LANSA for the Web, which are:

1. LANSA for iSeries

This is the core development environment. It provides the Repository and RDML technology used for the application database and programs.

2. LANSA for the Web

LANSA for the Web extends the LANSA for iSeries development environment. It provides the HTML components as well as the transaction support for executing applications over the Internet.

3. Item removed from diagram

4. LANSA for the Web Administrator

The Windows-based LANSA for the Web Administrator is used to configure the

Web transaction environment and is only required as an administrator so is not required on more than the Administrator's PC.

5. Web Server

The Web Server can be on a separate platform (Windows or IBM i) or all software can be on a single IBM i. LANSA for the Web can be used with a number of Web Servers including IBM's HTTP Server for i5/OS, or Microsoft IIS for Windows.

6. CGI

You can use CGI with the IBM HTTP Server.

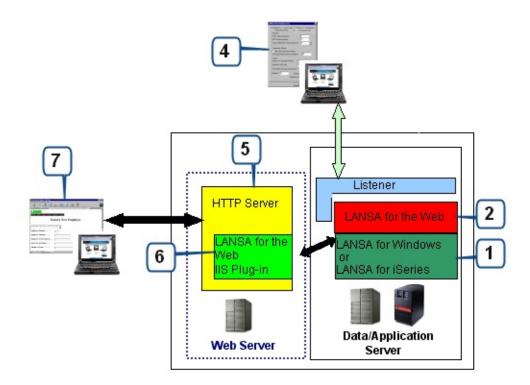
7. Client/Browser

The Client's browser sends requests to the Web Server and then it displays the resulting Web page. Internet Explorer is recommended.

 \Uparrow 1.2.11 LANSA for the Web Development Models

Windows Software Components

When installing and configuring LANSA for the Web on Windows, it is very important to understand the fundamentals of its architecture.



The diagram above identifies several of the key components used by LANSA for the Web. These components are:

1. LANSA for iSeries or Visual LANSA

The LANSA for iSeries is the development environment if an IBM i Data/Application Server is used. The Visual LANSA is the development environment if a Windows Data/Application Server is used. LANSA provides the Repository and RDML technology used for application database and programs.

2. LANSA for the Web

LANSA for the Web extends the LANSA development environment. It provides the HTML components as well as the transaction support for executing applications over the Internet.

3. Discontinued

4. LANSA for the Web Administrator

The Windows-based LANSA for the Web Administrator is used to configure the Web transaction environment and is only required as an administrator so is not required on more than the Administrator's PC.

5. Web Server

LANSA for the Web is used with Microsoft IIS and IIS Plug-in technology.

6. IIS Plug-in

LANSA for the Web on Windows uses IIS Plug-in technology.

You can use LANSA for the Web IIS Plug-in using ISAPI technology.

7. Client/Browser

The Client's browser sends requests to the Web Server and then it displays the resulting Web page. Internet Explorer is recommended.

1.2.11 LANSA for the Web Development Models

1.3 Choosing a Development Model

For information about choosing development models, refer to the following:

- 1.3.1 IBM i Master for Development?
- 1.3.2 Local Client or Network Client?
- 1.3.3 Windows Master for Development?
- 1.2 LANSA Development Models

1.3.1 IBM i Master for Development?



Pros

- IBM i centric development. It's the model of choice for development shops with a major focus on IBM i.
- Existing IBM i change management systems may be used.
- Can develop without a connection to the Master System.
- It's the model you've been using.
- The LANSA development team has used this model extensively for developing LANSA in LANSA. (Not sharing the database with Repository Synchronization)
- All developers can be kept up to date using Repository Synchronization.
- Using Individual PC databases and Repository Synchronization together provides for some control over receiving other developer's changes as they are only received when connecting to Host Monitor.

Cons

- The Master System must be available when installing the IBM i Slave and whenever system data needs to be updated. (System Initialization and Partition Initialization)
- The Master System must be available to obtain permission to modify an object (Check Out) and to make those changes available to other developers (Check In)
- When the database is shared or Repository Synchronization are used, other

developer's changes are updated into a developer's environment according to the schedule of the other developer. They are not obtained on demand. That is, the developer is not sandboxed.

- If the database is installed on each Developer PC then this requires more disk resource.
- Each Developer needs to install and update their Visual LANSA software.

Note: The redundancy afforded by having the Master System reduces the importance of backing up individual PC databases. If a PC database is lost, then only the changes up to the last check in will be lost. If developer's check in frequently then little is exposed.

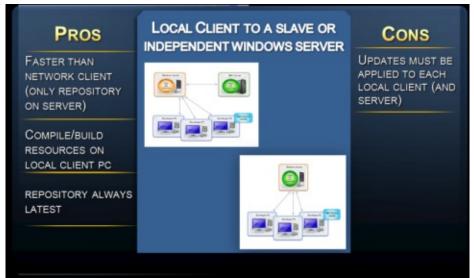
1.3 Choosing a Development Model

1.3.2 Local Client or Network Client?

A major constraint of deciding to use a Client model over individual developer databases (IBM i Slave), apart from the pros and cons listed, is the hardware specification of the client PCs and the server PC. If the server is relatively low specification, Network Client performance may be severely affected. On the other hand, if a powerful server is available the Network Client model becomes feasible.

A further difference with IBM i Slave is that the repository is shared so other developer's changes are immediately impacting the work environment - either positively as the updates are needed or negatively because they are defective. There is more control with an IBM i Slave, and even more with an Independent Windows Workstation to a VCS Master.

Local Client



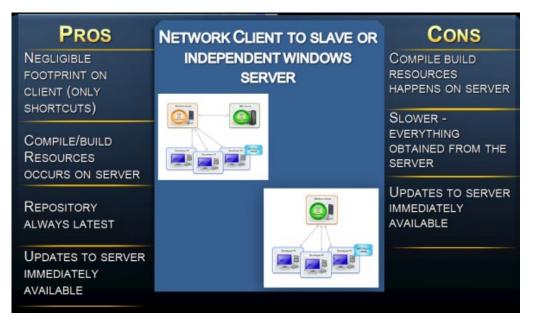
Pros

- Typically faster than a Network Client as the local PC resources are used for everything except the database.
- Compiling and building is usually faster
- Compiled objects are not shared so they may be compiled at any time

Cons

• Each Developer needs to install and update their Visual LANSA software, as well as the Windows Server.

Network Client



Pros

- A negligible footprint on the client. Not much more than a few shortcuts are added to the client.
- Only the server needs to be updated with new LANSA releases.
- The powerful server resources are used to compile and build.
- Option to use the compiler on the client.
- Repository is always up to date with other developer's changes.
- Execution environment is always up to date with other Developer changes.

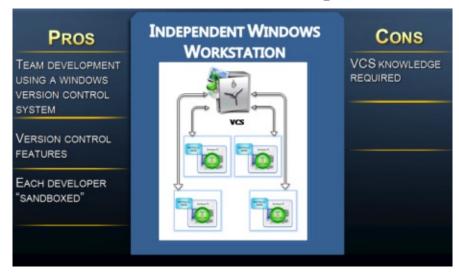
Cons

- Compiled objects are shared so if another developer is using the object then it cannot be compiled and if its not being used then another developer's untested work may introduce defects into your code delaying development.
- Insufficient server resources reduce productivity.

Note: Client Systems are sharing the database. This reduces the effectiveness of the redundancy afforded by having the Master System. If a PC database is lost, then the changes of ALL developers using that database up to the last check in will be lost. So using a Client model increases the need for a good PC database backup strategy to be implemented.

1.3 Choosing a Development Model

1.3.3 Windows Master for Development?



Pros

- Windows centric team development. It's the model of choice for development shops with a major focus on Windows.
- Existing Windows change management systems and version control systems may be used.
- Can develop without a connection to the VCS Master System.
- The LANSA development team is using this model for developing LANSA in LANSA.
- Developers can choose when to update their Repository and what part of it, down to an individual object. The developer is in complete control of what LANSA Object updates they receive. Nothing effects their development environment without their express and precise consent. That is, they are sandboxed from all other developers.
- Security and Task Tracking are disabled and are replaced by the VCS method of controlling access to LANSA objects. For example, a VCS may require an object to be checked out and that object may have limited permissions on it.
- System data is maintained by Visual LANSA and can also be received from the VCS Master via alterations that have been made by another Developer and checked in to the VCS Master.
- The sophistication of the development environment is expanded to the extent that the VCS Master provides those features. Those features may include,

but are not limited to, branching, merging, source comparison, patches, labeling, and bug tracking integration.

Cons

- A thorough understanding of the chosen version control system is required and it must also be administered and maintained.
- The VCS Master System must be available when installing the IBM i Slave and whenever system data needs to be updated. (System Initialization and Partition Initialization)
- The Master System must be available to obtain permission to modify an object (Check Out) and to make those changes available to other developers (Check In)
- Each Developer PC requires more disk resource than with any other model.
- Each Developer needs to install and update their Visual LANSA software

1.3 Choosing a Development Model

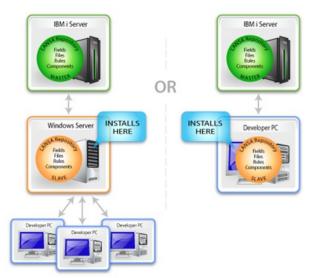
1.4 Recommended Development Models

For information about recommended development models, refer to the following:

- 1.4.1 Recommended IBM i Development Model
- 1.4.2 Recommended Windows Development Model
- ↑1.2 LANSA Development Models

1.4.1 Recommended IBM i Development Model

The recommended IBM i development model is a Visual LANSA Slave to an IBM i Master System.

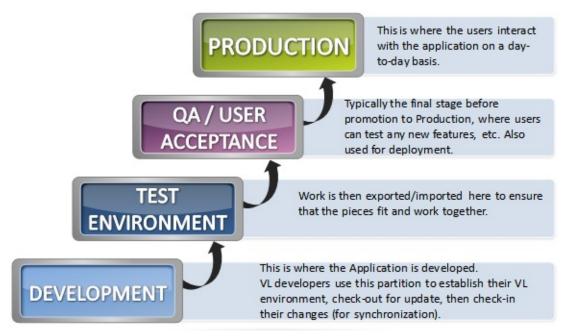


The reasons for this development model are:

- The primary target platform is IBM i.
- Development is primarily IBM i in-house.
- A change management system or a version control system third party tool is used on IBM i.
- This model can also be used for Windows development.
- These are the requirements for this development model:
- Multiple partitions for staging For details, see System Partition Definitions in the *LANSA* for iSeries User *Guide*.
- Task tracking and security For details, refer to Task tracking in the *LANSA for iSeries User Guide*.
- Repository Synchronization For details, see Repository Synchronization in the LANSA for iSeries User *Guide*.
- Interfaces to third party tools for change management or version control.
- Separate build machine for Windows deployments, if deploying to Windows too.

↑1.2 LANSA Development Models

Setup - Partitions



The Development partition is where the application is developed and where Task Tracking is used to track changes, lock objects to tasks, and synchronize changes as they occur. Developers check out objects to work on, make their changes, then check-in those changes. Unlocking of objects from tasks may occur on check-in, depending on the Task Tracking approach. When changes are complete, objects can be moved (export/import) to the Test partition for more stringent integration testing. Export List can be moved, as well.

The Test Environment is where Integration testing is performed. No changes are made here to the code here; all new/changed objects originate from Development. When testing is complete, objects are moved (export/import) to QA/User Acceptance environment, using and moving export list that originated in Development.

Quality Assurance / User Acceptance – Final testing/staging before promotion to Production; should be identical to Production with the exception of objects/feature/fix being evaluated; when users sign off on the changes, objects are promoted to Production (export/import).

Production – Final destination for any changed objects – this is where the endusers perform their day-to-day work.

Also See

System Partition Definitions in the iSeries User Guide.

1.4.1 Recommended IBM i Development Model

Setup - Task Tracking IBM i

All Task IDs are created and maintained in the LANSA for iSeries Master System, and all partition and other task tracking settings are set up in the LANSA for iSeries Master System.

Settings at the System level include Special Task IDs to support different Task Tracking approaches, and when objects are unlocked from a task.

Partition-level settings specify how developers interact with Task Tracking.

Export lists can be created to include all objects locked to a task, and the export list itself can be included in the export.

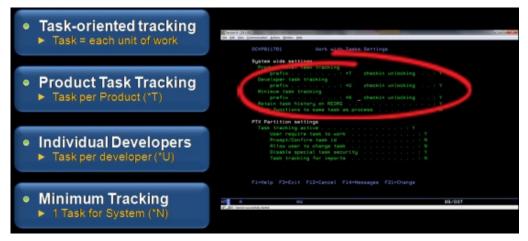
Deliver To can be used to transport objects to a Deployment System (IBM i or Linux), and can include an export list created on an IBM i).

Also See

Working with Tasks in the iSeries User Guide

↑1.4.1 Recommended IBM i Development Model

Setup - Task Tracking Visual LANSA



Task-oriented tracking

A task ID is created for each new unit of work, which might represent a new feature or a "fix". As work is completed, the task is "closed". All objects worked under a task are then exported for testing; once promoted to Production, the task is "finished" and all objects are unlocked from the task.

Product task tracking

Unique task IDs are created for each "product", or group of objects; multiple developers may use the same task ID. Use special task ID names (*T) with Repository Synchronization and Check-in Unlocking.

Individual developers

A task ID is assigned to each developer, who is the only one authorized to use that task. Use special task ID names (*U) with Repository Synchronization and Check-in Unlocking. User tasks are never "closed".

Minimum tracking

Only 1 task is created; all are authorized and use that task ID. Use special task ID name (*N; e.g. *NONE) with Repository Synchronization and Check-in Unlocking.

Also See

Approaches for using Task Tracking in the *Visual LANSA Administration Guide* 1.4.1 Recommended IBM i Development Model

Setup - Repository Synchronization

Host Repository Synchronization / Repository Groups / Work Groups

A Repository Group is a list of Visual LANSA PCs with a repository to be synchronized. Optional Work Groups can be used be used to group Visual LANSA PCs that share the same group of objects and need to be synchronized as a separate group. If no Work Groups are defined, all propagations go to all repositories.

Also See

Repository Synchronization Concepts in the *Visual LANSA Administration Guide*

↑1.4.1 Recommended IBM i Development Model

Lock by PC Name

This is a new feature in Version 13. When an object is checked out it is locked to the PC. No other PC can modify it. When it is checked out read-only, it is not locked to the PC. So it is important to check out read-only unless you intend to modify it.

There is an unlock option in the Visual LANSA IDE if you need to allow another developer to access it.

It can also be unlocked on the IBM i. See

Also See

Unlock Objects in Task Tracking

Change Task for an Object in the LANSA for iSeries User Guide

↑1.4.1 Recommended IBM i Development Model

Setup - Change Management/Version Control Strategy

There are many facets to change management within LANSA. There may be Third Party change management systems that suit your needs. See Third Party Packages in the *Visual LANSA Administration Guide*

Ensure that you understand all the following concepts and how they fit in with your development needs.

Impact Analysis – As objects that require change are identified, Impact Analysis can be used to assess the effect on related objects. See Impact Analysis Concepts in the *Visual LANSA Administration Guide*

Tasks – Tasks are created per the Task Tracking approach chosen. See Task Tracking Concepts in the *Visual LANSA Administration Guide*

Object History & Movement – As work is performed and objects are moved in and out of Development and Testing, a history of the changes made and an object's movements should be retained.

Synchronization – If multiple developers are working with the same group of objects, a strategy to ensure that they are working with current versions of objects, including changes, must be established. See Repository Synchronization Concepts in the *Visual LANSA Administration Guide*

Testing – A clear and concise procedure must be established for testing at each stage, and movement to the next phase of testing. This will include a comprehensive test plan which will change to accommodate new features and scenarios. This should include error reporting and correction.

Migration between environments – As changes are made, testing performed and work is completed, procedures to move between development, testing, acceptance and production environments are established.

Deployment – Once all work is completed and testing has been signed-off on, how will the final product be delivered? See Deployment Concepts in the *Visual LANSA Administration Guide*

Also See

Change Management in the Visual LANSA Administration Guide

1.4.1 Recommended IBM i Development Model

1.4.2 Recommended Windows Development Model

The recommended Windows development model is an Independent Windows Workstation to a VCS Master.

VCS VCS
INSTALLS HERE Developer RC HERE Developer RC HERE

The reasons for this development model are:

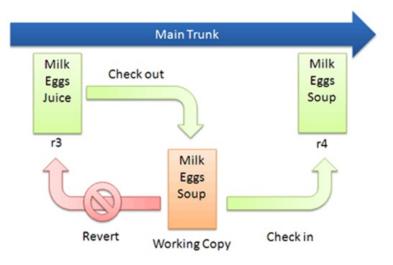
- The primary target platform is Windows.
- Development is primarily Windows in-house.
- Already using a Version Control System and, potentially, a change management system on Windows.
- These are the requirements for this development model:
- Version Control System of your choice
- No task tracking or security
- Separate build machine for Windows deployments.
- All developers use Independent Windows Workstation to a VCS Master

The Version Control System is a fundamental part of this model. The decision about which VCS suits your development shop will affect every facet of the development life cycle. VCS are many and varied. As such, it is almost essential that existing administrative and management experience in the use of a VCS already exists in-house. See the following sections for assistance in where to start with VCS:

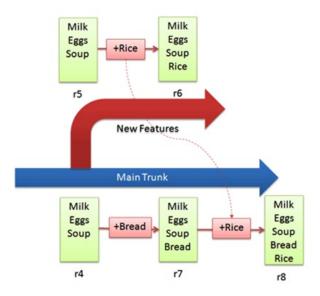
1.2 LANSA Development Models

What is a Version Control System?

- A VCS helps provide discipline for team development
- It tracks and provides control over changes to source code
- Lets you track all changes to files over time and by whom.
- Provides an easy mechanism to revert to previous working versions to back out defects not just at a file level, but a whole system too.
- Manages the locking of source code, merging, reversion, source comparison and much more...



In this diagram above we have a simple file (revision 3) that contains the text "Milk Eggs Juice". The developer checks it out and modifies the working copy so it now contains "Milk Eggs Soup". This change may be checked in, in which case it becomes revision 4 in the VCS. Or, the change is reverted to revision 3 so that the working copy is now "Milk Eggs Juice". This is a simple concept but provides for a powerful paradigm that allows the developer much more freedom to try out changes knowing that its firstly easy to revert and secondly if the change is committed to the VCS that the exact changes made are recorded and can be reverted at any time in the future.



In this scenario, the main trunk is branched at revision 4 to create a parallel development (revision 5) for adding new features to the product whilst the Main Trunk continues with bug fixes being applied. The developer gets straight into it and adds in the new feature "Rice" which produces revision 6. This change waits until sometime in the future when a new release is required and the change will be merged back into the Main Trunk.

Meanwhile a bug is reported from the field so "Bread" is added and this change is issued as a bug fix. Clearly the customer does not receive the change for the new feature. Its been isolated from the released code.

The time comes to produce the next release so the new features are merged back to the Main Trunk adding "Rice" to the file and resulting in "Milk Eggs Soup Bread Rice". This is a semi-automated step. The system can make most changes accurately but where the same line has changed it will report conflicts and require that the conflict is resolved before the change can be committed into the VCS to create revision 8. This happens here because line 4 has been changed in both the Main Trunk and Branch. In this case the developer decided that both changes needed to stay with the new feature change added after the Main Trunk change.

Also See

Working with Tasks in the iSeries User Guide

1.4.2 Recommended Windows Development Model

Why do I need a Version Control System?

Refer to these articles: Visual Guide to Version Control Visual Guide to Distributed Version Control Good excuses NOT to use version control Comparisons of some VCS tools

Example Costs

These costs are indicative only. Contact the supplier for exact costs and for purchasing the licenses.

Perforce - Free for up to 20 users and 20 workspaces, or unlimited users with up to 1,000 files, or for OSS or educational use; else \$900 per seat in perpetuity, with volume discounts, or \$360 per seat on a subscription model

Team Foundation Server - Free for up to 5 users in the TFS 11 express edition or for open source projects through codeplex.com; else non-free, licensed through MSDN subscription or direct buy

Vault - \$300 per user

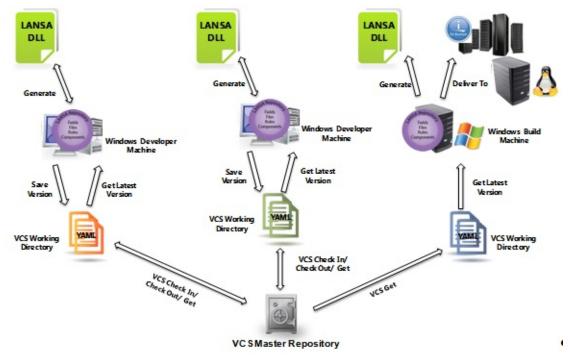
Subversion – Free

Git – Free

Mercurial – Free

Kiln built on Mercurial (Maximum \$25 per user per month hosted. Including bug management - \$30)

↑ 1.4.2 Recommended Windows Development Model



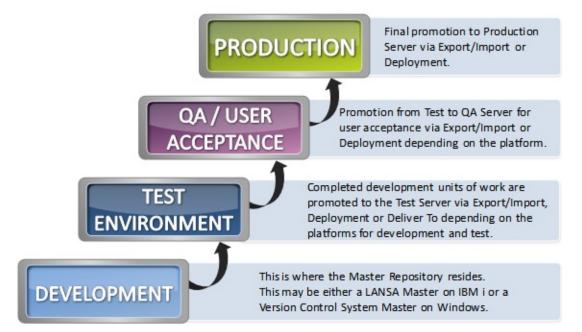
Visual LANSA Development using a Version Control System

This environment may be compared to the Visual Studio development paradigm. Visual LANSA is editing text files that have been checked out from the VCS or is creating them to be put in the VCS. The Repository is still used to contain the LANSA objects, but they are now reflected into a directory which the VCS also has a view of.

Also See

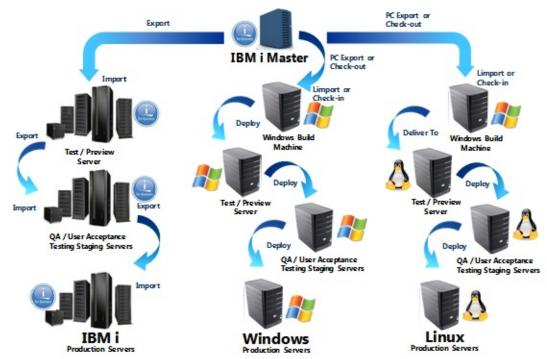
1.4.2 Recommended Windows Development Model

1.5 Promotion & Deployment



For further information on promotion and deployment, refer to the following:

- 1.5.1 IBM i Master Deployment Options
- 1.5.2 Windows Master Deployment Options
- 1.2 LANSA Development Models



1.5.1 IBM i Master Deployment Options

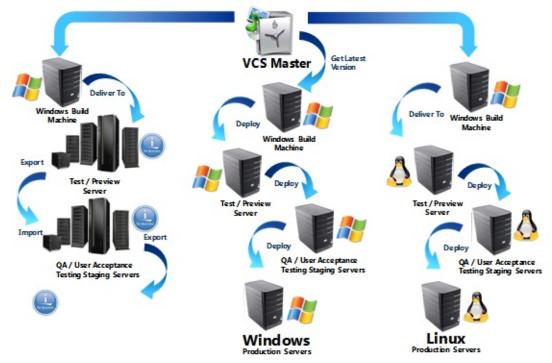
There is an IBM i Master. Deploy to the Test Environment on IBM i, Windows and Linux as described in this list:

- To IBM i: Use IBM i Export from the IBM i Master.
- To Windows: Get the latest objects from the IBM i Master to the Windows build machine then use Deployment Tool.
- To Linux: Get the latest objects from the IBM i Master to the Windows build machine then use Deliver To

Also see

1.2.8 Windows Build Machine

1.2 LANSA Development Models



1.5.2 Windows Master Deployment Options

There is a VCS Master and a single Windows Build Machine. Deploy to the Test Environment on IBM i, Windows and Linux as described in this list:

- To IBM i: Get the latest objects from the VCS Master to the Windows build machine. Use Deliver To to move the objects to the IBM i. Deliver To can create an export list on the IBM i test environment. Then Export/Import from there to move to Production.
- To Windows: Get the latest objects from the VCS Master to the Windows build machine then use Deployment Tool.
- To Linux: Get the latest objects from the VCS Master to the Windows build machine then use Deliver To

Note: If a VCS Master is not being used for Windows development, then the picture starts with the Windows Build Machine.

Also see

- 1.2.8 Windows Build Machine
- 1.2 LANSA Development Models

1.6 LANSA for the Web Deployment Models

A deployment model will consist of a combination of two or more of the following components:

- A Data/Application Server, which can be a Windows or an IBM i server. This is where your LANSA Repository resides and where the LANSA application runs.
- A Web Server, which can be a Windows or an IBM i server.
- A PC connected to the server to run the **LANSA for** the Web Administrator.
- A Browser PC. This PC could be anywhere, depending on the application you are running.

You will need to install LANSA on one or more platforms to create the desired configuration.

If you are using a Windows Multi-Tier deployment model, you will typically use separate Windows machines for your Web Server and your Data/Application Server:



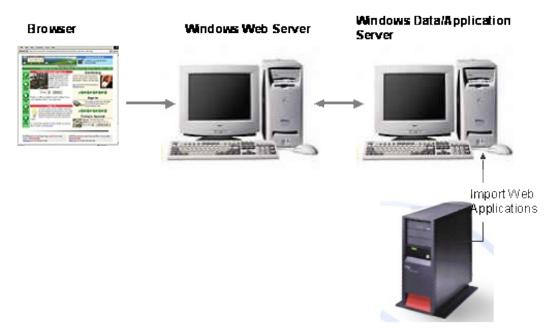
Refer to 1.6.1 Considerations for the Windows Multi-Tier Deployment Model for more details.

If you are using a Mixed Multi-Tier deployment model, you will typically use a Windows Web Server and an IBM i Data/Application Server. (Refer to the Installing LANSA on IBM i *Guide* to set up your Data/Application Server.)



Refer to 1.6.2 Considerations for the Mixed Multi-Tier Deployment Model for more details.

You can use a Windows Multi-Tier model, but develop your application on an IBM i as shown in the following example. In this case, you will import the Web application to transfer your existing LANSA application from the IBM i development environment to the Visual LANSA development environment.



If you are using a Single-tier IBM i deployment model, consisting of a single IBM i acting as both the Web Server and Data/Application Server, you will use the *Installing LANSA on IBM i Guide*.



If you are using a Multi-Tier IBM i deployment model, which uses an IBM i Web Server and an IBM i Data/Application Server, you will use the *Installing LANSA on IBM i Guide*.



1. Planning Your Site

1.6.1 Considerations for the Windows Multi-Tier Deployment Model

In a Windows Multi-Tier deployment, you will be using a Windows Web Server with a Windows Application Data Server.

Following are some considerations for a Windows Multi-Tier deployment model:

- The Visual LANSA development environment is installed on the Data/Application Server. The LANSA Repository (including the generated HTML) and your application data are installed on this machine.
- The LANSA for the Web PC-based Administrator will only need to connect to the Data/Application Server.
- A Visual LANSA system is not required on the Windows Web Server. Only the LANSA for the Web IIS Plug-in needs to be installed on the Web Server.
- The Windows Web Server can be Microsoft IIS with the IIS Plug-in.
- \Uparrow 1.6 LANSA for the Web Deployment Models

1.6.2 Considerations for the Mixed Multi-Tier Deployment Model

In a Mixed Multi-Tier deployment, you will be using a Windows Web Server with an IBM i Application Data Server.

Following are some considerations for a Mixed Multi-Tier deployment model:

- The LANSA for the Web development environment is installed on the Data/Application Server along with LANSA for iSeries. The LANSA Repository (including the generated HTML) and your application data are installed on the Data/Application Server.
- The LANSA for the Web PC-based Administrator will only need to connect to the IBM i Data/Application Server.
- The Windows Web Server can be Microsoft IIS with the IIS Plug-in.
- A Visual LANSA system is not required on the Window Web Server. Only the IIS Plug-in needs to be installed on the Web Server.
- A Listener must be configured on the IBM i Data/Application Server for the communications link from the Windows Web Server.
- Refer to the LANSA for the Web Administrators Guide for set up instructions for Multi-Tier Deployment.
- 1.6 LANSA for the Web Deployment Models

1.6.3 Multi-Homing with LANSA for the Web

Multi-homing allows you to use a single Web Server to handle multiple domain names or IP addresses. LANSA for the Web supports multi-homing.

If you are installing LANSA for the Web for the first time, you may wish to start with a single site. Once this system is operational, you can reconfigure the system to support multiple sites from your LANSA system.

Following are some considerations for multi-homing with LANSA for the Web:

- You can configure a different LANSA system for each of the domain names (i.e. you can have more than one LANSA system installed), or you can configure a number of domain names to be directed to the single LANSA system. For example, you could configure each partition to have a separate IP address.
- The LANSA for the Web Administrator is used to configure the multihoming support. Be very careful when configuring multi-homing with the LANSA for the Web Administrator. In most cases, you must enter the system names or IP addresses instead of using the *DEFAULT value for system names.

1.6 LANSA for the Web Deployment Models

1.7 LANSA Integrator Development/Deployment Components

The LANSA Integrator software is installed by selecting individual components. You may need to install one or more software components to create the desired configuration.

The LANSA Integrator software includes the following software components:

1.7.1 Studio

1.7.2 JSM Server

1.7.3 JSM Proxy

1.7.4 RFI

1.7.5 User Agent

You may install them all together or separately as described in LANSA Integrator Install and Set up.

1. Planning Your Site

1.7.1 Studio

Integrator Studio simplifies the management with the JSM Server by providing a single point of access for the Integrator tools and the files created by those tools.



Integrator Studio provides an easy to use graphical interface that greatly simplifies the configuration and setup of the JSM Server environment.

1.7.2 JSM Server

The JSM Server may be installed on the same machine as the LANSA Data/Application Server.

As the JSM Server supports HTTP, FTP, SMTP and other services, it may be installed on the same machine as the Web Server.



The JSM Server Installation will install the JSMDirect CGI program, the Java Service programs, and other JSM objects on the server.

JSMDirect is the HTTP service extension to JSM. JSMDirect allows a LANSA or 3GL program to participate in Web services and B2B transactions. Microsoft IIS and Windows Script Hosting files must be installed.

The JSM Server requires a JRE installed on the machine.

1.7.3 JSM Proxy

JSMProxy is an HTTP proxy extension to JSMDirect. It allows a proxy server to redirect the request. Using JSMProxy is very useful as it allows you to define a single entry point and then redirect to a number of different remote servers. Microsoft IIS and Windows Script Hosting files must be installed.



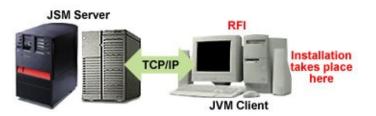
The JSM Proxy Installation will install the JSMProxy CGI program.

JSM Proxy does not require a JRE installed on the machine.

JSM Proxy is typically installed on a different server than the JSM server. The purpose of a proxy is to redirect requests between servers.

1.7.4 RFI

Remote Function Invocation (RFI) is a means of sending and receiving serialized Java objects between a client JVM and the remote JSM server.



The RFI Example Viewer is a simple GUI application that illustrates how to use the RFIService client class and the JSM service RFIDataSourceService. The Java programmer modifies the RFIExample source code and compiles and runs the RFI Example Viewer application to execute the RFIExample class to perform a Remote Function Innovation (RFI) to the remote JSM server.

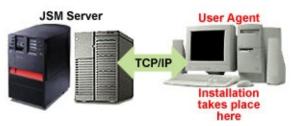
RFI requires a Java SDK (Software Developer Kit), which includes the JRE, installed on the PC.

RFI may be installed on the developer's PC.

Note: This software can be installed with Integrator or it can be installed separately. Refer to Task: Install Other Features.

1.7.5 User Agent

The User Agent is a Java-based tool, which allows file upload to a JSM HTTP Service using JSMDirect and sending of MQ-Series messages. The user agent allows Windows clients to enter data in spreadsheets (or in text files, comma or tab delimited formats or as more complex XML files), and send the validated files to remote hosts.



The User Agent requires a JRE installed on the PC.

The User Agent may be installed on an end-user's PC or on the developer's PC. **Note:** Note: This software can be installed with Integrator or it can be installed separately. Refer to the Installing LANSA on Windows Guide.

2. Getting Started with Administration

In this section is an overview of working with LANSA.

- 1. If Visual LANSA has not been installed, you should review 2.1 Installing LANSA.
- 2. If you have an Independent Visual LANSA system installed, review 2.3 Independent System Administration for a list of tasks.

OR

2. If you have a Slave Visual LANSA system installed, review 2.4 Slave System Administration for a list of tasks.

If you are unsure of the type of Visual LANSA system installed, refer to Choosing a Development Model and then review the appropriate task lists.

- 3. All Administrators should review 2.5 Development Environment Administration.
- 4. If you are using multiple configurations, review 2.1.4 Visual LANSA Administration.

2.1 Installing LANSA

The following sections provide a brief overview of:

2.1.1 Installing Visual LANSA

2.1.2 Upgrading Visual LANSA

2.1.3 Applying EPCs

2.1.4 Visual LANSA Administration

It is recommended that you open and review the Installing LANSA on Windows Guide and Planning Your Site.

Also See

2.5 Development Environment Administration

What is LANSA? in the *Documentation Roadmap*.

 $\ensuremath{\Uparrow}$ 2. Getting Started with Administration

2.1.1 Installing Visual LANSA

Once you have completed Planning Your Site and you have identified the required Visual LANSA System Types, you can begin the installation of the LANSA software on the required machine(s).

The steps for installing the Visual LANSA software are described in the *Installing LANSA on Windows Guide*.

If you require more than one LANSA System on a single machine, refer to 2.1.4 Visual LANSA Administration in this guide.

Also See

Licensing on the LANSA web site.

- 2.1.2 Upgrading Visual LANSA
- ↑ 2.1 Installing LANSA

2.1.2 Upgrading Visual LANSA

If you already have a version of the Visual LANSA Software installed, you may need to upgrade this software when a new version of LANSA is released.

The steps for upgrading the Visual LANSA software are described in the Installing LANSA on Windows Guide. Be sure to upgrade all the related systems, beginning with the IBM i Master, then the Windows Server then Local or Network Clients, as applicable.

Upgrading from Version 12 to Version 13: The introduction of PC Locks requires that all objects are checked into the Master IBM i System before upgrading.

Also See

- 2.1.3 Applying EPCs
- ↑ 2.1 Installing LANSA

2.1.3 Applying EPCs

Periodically, LANSA provides Expedited Program Changes (EPCs) to distribute minor updates and new features.

Each EPC will consist of a document, the updated software and usually a load routine. The EPC load routine may vary depending on the type of software objects to be installed by the EPC. The documentation will include instructions for the installation and removal of an EPC.

Read all documentation before installing an EPC.

Some EPCs require a prerequisite EPC to be loaded. When you are loading an EPC, the load process will check for the existence of previously required EPCs.

To see a list of EPCs already applied to Visual LANSA, use the Help menu in the LANSA Editor and select *Visual LANSA About* option.

↑ 2.1 Installing LANSA

2.1.4 Visual LANSA Administration

The administration of Visual LANSA focuses on the specific tasks for configuring and controlling the Visual LANSA environment and software. These are tasks that are performed by LANSA system administrators or developers after the software has been installed. Some administration tasks, such as partition and user setup, are required prior to the developers using the development environment.

Visual LANSA Administration tasks include the following types of activities:

- System definition maintenance
- Partition maintenance
- User and security maintenance
- Repository synchronization settings
- Task tracking set up and maintenance
- Object locking maintenance

Depending upon your development model and site standards, administration tasks may be completed by a single LANSA System Administrator or by individual LANSA developers. In most cases, the administration tasks will be shared based on the significance and scope of the changes being made. It is strongly recommended that site standards be defined for the administration of your LANSA Systems.

The setting of specific Visual LANSA Editor features is considered to be part of the user interface management and is the responsibility of individual developers. For details of these editor settings, refer to the *Visual LANSA User Guide*.

Also See

- 2.3 Independent System Administration
- 2.4 Slave System Administration
- 2.5 Development Environment Administration
- ↑ 2.1 Installing LANSA

2.2 Independent System Administration with a VCS Master

A Visual LANSA Independent System is identified by the fact that it may maintain its own System Definition data. The Visual LANSA Independent System does not require other systems to maintain any part of it and it has no other systems that strictly depend on it.

A Visual LANSA Independent System may be connected to other Visual LANSA Independent Systems in a peer arrangement using a VCS Master. In this arrangement each Visual LANSA Independent System must maintain its own System Definitions or import them from one of its peers. For consistency it is recommended to nominate one Visual LANSA Independent System as the Administration system and to export the System Definitions from there to its peers.

On the other hand, Partition Definitions are controlled by the VCS Master. Any developer may modify them, provided they have authority from the VCS. Other developers may then get the latest version from the VCS to update their Partition Definition.

The Visual LANSA Independent System to a VCS Master does not have an IBM i Master soHost Monitor and Repository Synchronization are not required.

The typical Administration tasks that are performed as part of an Independent installation include:

- 2.3.1 Maintaining System Definitions on Independent Systems on Independent Systems
- 2.3.2 Maintaining Partition Definitions on Independent Systems on Independent Systems

If you have installed a Slave Visual LANSA System, refer to 2.4 Slave System Administration.

Also See

2.5 Development Environment Administration

VCS Master Setup

 \Uparrow 2. Getting Started with Administration

2.3 Independent System Administration

The Visual LANSA Independent System has no other LANSA development systems that depend upon it, i.e. it has no slaves systems and it is not part of a distributed development model. Hence, there is just one LANSA Repository so a Host Monitor and Repository Synchronization are not required. (Note: It does not matter if the Repository is stored on a single workstation or on a server being used by many developers.)

The typical Administration tasks that are performed as part of an Independent installation include:

- 2.3.1 Maintaining System Definitions on Independent Systems on Independent Systems
- 2.3.2 Maintaining Partition Definitions on Independent Systems on Independent Systems
- 2.3.3 Maintaining Users and Tasks on Independent Systems on Independent Systems

If you have installed a Slave Visual LANSA System, refer to 2.4 Slave System Administration.

Also See

2.5 Development Environment Administration

2.3.1 Maintaining System Definitions on Independent Systems

A Visual LANSA Independent System can maintain its own System Definition data. When a Visual LANSA System is first installed or upgraded to a new version of the LANSA software, you are required to perform a System Initialization. The System Initialization installs a current set of LANSA System Software definitions based on a default set of shipped settings.

Typical Administration tasks include:

- When Visual LANSA System is installed, as part of the install, a System Initialization is performed.
- To change the installed System Definition, select System Information from the *Repository* tab or select it from the *File* menu. This option includes access to the operational settings which enable you to maintain the LANSA system characteristics that control how the development environment works.

You must have appropriate partition or system level authority to perform these tasks.

Also See

System Information

1 2.3 Independent System Administration

2.3.2 Maintaining Partition Definitions on Independent Systems

A Visual LANSA Independent System can create and maintain its own partitions within a LANSA System. When an Independent System is initially installed, a default DEM partition is typically installed along with default user and task settings. New partitions can then be added to the LANSA System.

When you first logon to a new Visual LANSA installation, you must use the PCXUSER/PCXUSER profile/password as this is the system owner and the only default user for the system. Once you logon as PCXUSER, you can create users, partitions, etc.

Note that users are not relevant to an Independent System to a VCS Master.

Typical Administration tasks include:

- To create or change a partition using the options on the Partition Maintenance tab. This includes enabling an existing RDML partition for RDMLX, using the RDMLX Partition Settings. For advice on changing a partition to RDMLX, review RDML and RDMLX Partitions Concepts.
- To initialize a partition using Partition Initialization.

You must have appropriate partition or system level authority to perform these tasks.

1 2.3 Independent System Administration

2.3.3 Maintaining Users and Tasks on Independent Systems

This section is not relevant when using an Independent System to a VCS Master.

A Visual LANSA Independent System can maintain its own users and tasks IDs. When you first logon to a new Visual LANSA installation, you must use the PCXUSER/PCXUSER profile/password as this system owner and the only default user for the system. Once you logon as PCXUSER, you can create users, partitions, etc.

Typical Administration tasks include:

- Creating a Visual LANSA user, using the *User* option from the *New* toolbar button or the the *New* submenu item from the *File* menu.
- Setting user access and User Privileges to Visual LANSA development for each user. Refer to User and Security Maintenance for more information.
- Creating or updating Task IDs. For details, refer to Using Task Tracking in LANSA and Task Maintenance.
- Updating object access and user authorities. Refer to User and Security Maintenance for information.

You must have appropriate partition or system level authority to perform these tasks.

Also See

System Initialization

↑ 2.3 Independent System Administration

2.4 Slave System Administration

A Slave LANSA System is not able to maintain the LANSA System Definition. It receives its system definition from a Master LANSA System. Hence, it is dependent upon the LANSA for iSeries Master System and the System Definitions must be updated on each Visual LANSA slave whenever a change has been made to the Master System. A Slave LANSA System shares Application Repository Data with the Master System.

It is very important that you understand all of the Host Monitor Concepts. The typical Administration tasks that are performed as part of a Slave installation include:

- 2.4.1 Maintaining System Definitions on Slave Systems on Slave Systems
- 2.4.2 Maintaining Partition Definitions on Slave Systems on Slave Systems
- 2.4.3 Maintaining Users and Tasks on Slave Systems
- 2.4.4 Configure LANSA Communications Software

You should also be familiar with the Host Monitor.

If you have installed an Independent Visual LANSA System, refer to 2.3 Independent System Administration.

Also See

2.5 Development Environment Administration

 \uparrow 2. Getting Started with Administration

2.4.1 Maintaining System Definitions on Slave Systems

When a Slave Visual LANSA System is first installed or upgraded to a new version of the LANSA software, you are required to perform a System Initialization. The System Initialization updates the required system software definitions in the Slave Visual LANSA System so that it matches the LANSA for iSeries Master System.

Whenever a change is made to the Master System, you must perform an update to the System Definitions in each of the Slave Systems to keep information synchronized. System Definition can be propagated automatically using Repository Synchronization.

Changes that require a system re-initialization would include the following:

- Modification to system definitions
- Additions or modifications to Standard Application Templates
- Additions or modifications to Standard Built-in Function Definitions
- Additions or modifications to Standard System Message File
- Additions or modifications to Standard System Variables
- Additions or modifications to Enrolled PC Names
- Additions or modifications to Enrolled PC Users
- Additions or modifications to Current Task List.

You can also use LANSA import/export features to transfer the system definitions. For more information, refer to Exporting System Definitions in the *iSeries User Guide*.

2.4.2 Maintaining Partition Definitions on Slave Systems

Slave Systems cannot change partition definitions. Once a change has been made on the Master System, the definition must be updated in each Slave System.

Typical Administration tasks include:

- To install a new partition in the Slave System or update an existing partition, refer to System Initialization and Partition Definitions
- To initialize a partition in Visual LANSA, use Partition Initialization.

The Partition Initialization can include the following:

- Partition Initialization to create or update the Partition Definition.
- Visual LANSA Framework (optional) installation in the Partition.
- LANSA for the Web (optional) enabled and installed in the Partition.
- Personnel System Demonstration (optional) objects installed in the Partition. Repository Synchronization can also be used to maintain partition definitions in Visual LANSA.

2.4.3 Maintaining Users and Tasks on Slave Systems

Slave Systems cannot maintain the user and task details. Once a change has been made on the Master System, these details must be refreshed in each Slave System.

Typical Administration tasks include:

- To update user details Enrolled PC Users and Enrolled PC Names.
- To update task details Current Task List.

Repository Synchronization can also be used to maintain task information in Visual LANSA.

You can also use LANSA import/export features to transfer the user and task information. For more information, refer to Exporting System Definitions in the *iSeries User Guide*.

2.4.4 Configure LANSA Communications Software

When using a Visual LANSA Slave System, you must have appropriately set up the communication between the Master LANSA System and the Slave LANSA System. This communications set up is typically performed as part of the initial installation process. There may be times, after the software has been installed, when you must change these communications. For example, if the IP address of the server containing the Master LANSA System is changed, then the communications settings must be updated on the machine(s) containing the Slave System(s).

The LANSA communications software is summarized in the Communications Setup Guide.

You must be familiar with the LANSA Communications Administrator software on the machine that contains the Slave System.

A listener job must also be set up on the machine that contains the Master System.

2.5 Development Environment Administration

In addition to the LANSA System maintenance, there are administration tasks relating to maintaining the development environment:

- 2.5.1 Controlling How Developers Work
- 2.5.2 Controlling What Developers Can Build
- 2.5.3 Import/Export Operations
- 2.5.4 Development Environment Settings
- 2.5.5 Setting Systems Information

Also See

2.3 Independent System Administration

- 2.4 Slave System Administration
- \Uparrow 2. Getting Started with Administration

2.5.1 Controlling How Developers Work

As the LANSA Administrator, you are responsible for controlling how developers work in the Visual LANSA environment. As part of the Visual LANSA Master System and Partition definitions, you can control:

- User and Security Maintenance related to development options
- Task Tracking Set Up
- Repository Synchronization (if using Slave Systems)
- 2.5.4 Development Environment Settings

Administrators should be aware of these settings and create a plan for managing the development environment.

2.5.2 Controlling What Developers Can Build

As the LANSA Administrator, you will be able to control what developer can build in the Visual LANSA environment. As part of the Visual LANSA Master System and Partition definitions, you can control:

- RDMLX Partition Settings
- Available Field Types in a Partition
- 2.5.5 Setting Systems Information

Administrators should be aware of these settings and create a plan for managing the development environment.

2.5.3 Import/Export Operations

LANSA supports the exporting and importing of application definitions and data between different partitions or different LANSA Systems.

For information about importing information into Visual LANSA, refer to Export and Import.

Also See

Exporting and Import in the iSeries User Guide

2.5.4 Development Environment Settings

Visual LANSA allows you to control the behavior of the editor, as well as development environment settings, that are used when building applications by using Environment Settings in the editor.

This includes:

- LANSA Editor Settings
- Object Locking
- Workstation Locks
- Design View Language
- Development Language
- Language Options
- Message File Maintenance
- ↑ 2.5 Development Environment Administration

2.5.5 Setting Systems Information

There is System Level information that impacts your development environment including:

- Default date format
- Decimal format
- Default screen settings
- Default print settings
- Operational settings (DC@OSVEROP) for import/export, task tracking, I/O module characteristics, file overrides, etc.
- and more.

Administrators should be aware of these settings in order to assist developers. For an Independent Visual LANSA System, the LANSA System Definition is maintained locally using the options described in System Definitions.

For a Slave Visual LANSA System, the LANSA System Definition is maintained on the Master System and you must perform a System Initialization to update your own Visual LANSA system. Repository Synchronization can also be used to keep information up to date in Visual LANSA.

3. Visual LANSA Logon

For information about Visual LANSA Logon, refer to the following:

3.1 Visual LANSA Logon without a VCS Master

3.2 Visual LANSA Logon with a VCS Master

3.3 Logon Parameters

4.2 System Initialization

4.3 Partition Initialization

4.4 Single Sign-On (SSO)

PRIVATE:MADCAP:FILENAME:l4wADM02_0240

3.1 Visual LANSA Logon without a VCS Master

The Visual LANSA Logon is displayed when the *Visual LANSA Development Environment* is started.

Visual LANS	A Logon	×
User ID:	pcxuser	
Password:	*********	
Use Windo	ws <u>c</u> redentials	
Partition	Current Language	Task ID
DEM	English	PCXTASK
	Français Japanese	SET_E8
ОК	ystem Init	<u>M</u> essages Cancel Help

You must specify the following:

- 3.3.1 User ID
- 3.3.2 Password
- 3.3.3 Use Windows Credentials
- 3.3.4 Partition
- 3.3.5 Current Language
- 3.3.6 Task ID

Perform a 4.2 System Initialization whenever new details are to be updated on the PC. For a new Visual LANSA system, the *System Initialisation* automatically appears at the end of the installation process.

If you are using a new partition defined on a Master System, 4.3 Partition Initialization will automatically appear.

You must enter a User ID and Password before the 4.3 Partition Initialization option will be enabled.

```
4.4 Single Sign-On (SSO)3.3.8 Database Login
```

3.3.7 Messages

3.2 Visual LANSA Logon with a VCS Master

The Visual LANSA Logon is used to start a Visual LANSA session and optionally initialize the LANSA environment.

To start a Visual LANSA session, choose the Partition and text language to display in the IDE and click OK.

Visual LANSA Lo	ogon	×
Partition	Current Language English Français	<u>System Initialization</u> Partition Initialization
OK Cance	Help	

- To access the **System Initialization** options, press the *System Initialization*... button before you press the *OK* button.
- To access the **Partition Initialization** options, , press the *Partition Initialization*... button before you press the *OK* button.

For information, refer to the following:

3.3.4 Partition

- 3.3.5 Current Language
- 4.2 System Initialization

4.3 Partition Initialization

Security and Task ID are not required for Visual LANSA when using a VCS Master. The authority to objects is obtained from the VCS itself. That is, you will need to logon to the VCS Master external to LANSA and the VCS Master will control which objects you may access and to what degree.

The VCS may also have a mechanism for assigning work like a bug report, or a job or a task.

↑ Visual LANSA Administrator Guide's cover page.

3.3 Logon Parameters

For information about Visual LANSA Logon Parameters, refer to the following

3.3.1 User ID

- 3.3.2 Password
- 3.3.3 Use Windows Credentials
- 3.3.4 Partition
- 3.3.5 Current Language
- 3.3.6 Task ID
- ↑ Visual LANSA Administrator Guide's cover page.

3.3.1 User ID

Specify the User ID defined to access LANSA. The User ID determines your authority level within LANSA. You are given three attempts to log on to Visual LANSA. User ID is not case sensitive.

Please note that this User ID and Password is NOT the User ID and Password you use to logon to your particular Database Manager. Refer to 3.3.8 Database Login.

Visual LANSA User ID and Passwords are set up in the Master Repository.

If you are using a Slave System with an IBM i Master Repository, you may need to update the User ID details in the Visual LANSA System Definition using 4.2 System Initialization and selecting the 4.2.14 Enrolled PC Users option. This option will retrieve the most current list of all User ID authorized to use LANSA.

If you are using a Visual LANSA Master or Independent System, you can maintain user profiles by opening the appropriate user from the *Users* listed in the *System Information* list in the *Repository* tab or creating a new User by pressing the *New* button on the toolbar.

If you are using Single Sign-On and select the 3.3.3 Use Windows Credentials option, you don't need to enter a User ID and Password.

3.3.2 Password

Specify the corresponding password for the 3.3.1 User ID used for this session of LANSA. A password can be up to 128 characters in length and is case sensitive.

The password will not be displayed as you type it in.

Visual LANSA User IDs and Passwords are set up in the Master Repository.

Also See

4.2.14 Enrolled PC UsersUser and Security Maintenance↑ 3.3 Logon Parameters

3.3.3 Use Windows Credentials

When this option is selected, a User ID or Password need not be entered – they will be disabled. The User Id and Password that was used during initial Windows logon will be used instead.

For a slave system with an IBM i Master Repository, whenever new users are added or if accreditation details (for Single Sign-On) are added or changed, the Windows user details will need to be re-updated using System Initialisation using the 4.2.14 Enrolled PC Users option in *System Initialization*.

For a Visual LANSA Master or independent system, the user profiles are maintained from the *Repository* tab in the LANSA Editor.

For more information about using the Windows Credentials, refer to 4.4 Single Sign-On (SSO).

Also see

Edit User Definitions

3.3.4 Partition

Specify the partition to open. Your 3.3.1 User ID must be authorized to use this partition.

The Partition list shows all the current Visual LANSA partitions. If the required partition is not in the list, you will need to import the partition information into the Visual LANSA Repository from the Master Repository.

Selecting a different partition in the list may change the list of languages displayed.

You can only work with one partition at a time. If you want to work with multiple partitions, you must start multiple sessions of LANSA.

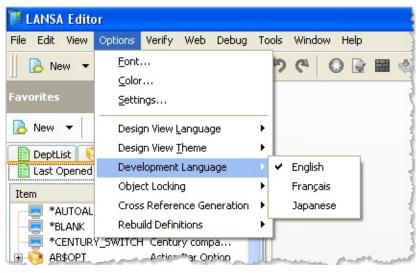
Also See

4.2.16 Partition Definitions4.2 System Initialization↑ 3.3 Logon Parameters

3.3.5 Current Language

If the selected partition is a multilingual partition, you must select a Language to work with. This will be the default language for the current session of LANSA.

Specify the current language to be used by LANSA in the selected 3.3.4 Partition. The list of languages displayed is specific to the selected partition.



Note: The current language selected for the partition and the development language as specified in the Development Language menu are not the same.

The development language determines the language of the LANSA development environment used the next time you log on to Visual LANSA. The language specified for the partition determines the default language for language-specific objects in your LANSA applications such as labels and the language used when the application is executed. The Design View Language is set to the current language but can be changed as required.

Also See

- 4.2.16 Partition Definitions4.2 System Initialization
- 1 3.3 Logon Parameters

3.3.6 Task ID

Specify the Task ID to be assigned to this session. Your 3.3.1 User ID must be authorized to use this Task ID.

If Task Tracking is used, you must select a Task ID. This will be the default Task ID for the current session of LANSA.

A Task ID ensures that only users who are authorized can modify the objects.

The Task ID list shows all the open or active Task IDs in the Visual LANSA System Definition.

If the required Task ID is not in the list, you will need to update the System Definition information by using the 4.2.15 Current Task List option on the 4.2 System Initialization dialog.

Also See

User and Security Maintenance

3.3.7 Messages

All the messages for the Visual LANSA Logon are accumulated. Selecting this option will display any current messages in a separate window.

3.3.8 Database Login

You can change the Database User ID that is currently in use. Select the Change DB Logon button on the Logon dialog to enter the required User Id and Password for the particular database that you are using. Refer to the documentation for the ODBC Driver that you are using for further information.

Connect to LANSA Database	
Login Details User ID: Bassword:	OK Cancel

The Change DB Logon... button is only available when an ODBC Data Source is being used. For example on DB2/2 this button is hidden.

Database User ID and Password

Specify the User ID and Password to be used for the logon to the database.

For the default database installed with Visual LANSA, the User ID is **DBA** and the Password is **SQL** (notice that these values are uppercase characters).

4. Visual LANSA Initialization

For information, refer to the following:

- 4.1 VCS Master Setup
- 4.2 System Initialization
- 4.3 Partition Initialization
- 4.4 Single Sign-On (SSO)
- ↑ Visual LANSA Administrator Guide's cover page.

4.1 VCS Master Setup

For information, refer to the following:

- 4.1.1 Considerations when Installing a New VCS Master
- 4.1.2 Migrating a Partition from a Slave to a VCS Master
- 4.1.3 Setting up 2nd and Subsequent Developers to use a VCS Master
- 4.1.4 Release Management with a VCS Master
- 4.1.5 Upgrading a VCS Master

Tip: Visual LANSA does not detect differences between the Repository and the VCS Working Folder. When you modify an object in the Development Environment its saves a copy to the VCS Working Folder and this is then picked up by the VCS as a difference. When you import into the Repository it is not saved into the VCS Working Folder. You must explicitly save the set of objects to the working folder. A useful tool for this is to always use the Development Environment Import which has an option to create a Repository list. You can then use that list to save all the imported objects to the VCS working Folder.

1 4. Visual LANSA Initialization

4.1.1 Considerations when Installing a New VCS Master

Review Independent System Administration with a VCS Master

The process is essentially the same as 4.1.2 Migrating a Partition from a Slave to a VCS Master except that the import step is not required.

Once the first machine is installed follow the steps in 4.1.3 Setting up 2nd and Subsequent Developers to use a VCS Master

1 4. Visual LANSA Initialization

4.1.2 Migrating a Partition from a Slave to a VCS Master

Note: the most efficient way of installing the Visual LANSA Independent System to a VCS Master is to perform a Custom Install and on the options for Initialize Database and Partitions only chose Example Partition Definition. This is because the whole partition will be replaced anyway. Other partition initialization like Visual LANSA Framework need to be done after the partition has been replaced with the Slave data:

LANSA V13 Installation/Upgrade - Options and Summary		
		11.600
Review the options below. Selecting a section header will display the summary for that section. Options which are in bold and have a question mark icon next to them MUST be reviewed before the install can proceed.	Specify whether the database sh created/updated and the specifie Initialize Database	ould be initialized. If so, the database will be d partitions initialized.
 All Options Common Options Windows Start Menu Folder Visual LANSA Options LANSA Network Name Database Data Source Initialize Database and Partitions IBM i Language Options Local Data Directory Listener Service Options User Id for Visual LANSA Web Configuration 	imports such as the VL Framew Partition was to take 10 minute suggests that you choose to in immediately and only with the o	Arted into EVERY partition being initialized. Some ork contain a large amount of data. So, if one es to initialize, 6 Partitions will take 1 hour. LANSA itialize only those Partitions you plan to use options that you want to use in all those Partitions. All n be done from within LANSA, so there is no need to epository Note: Separate multiple partition names with spaces.
LANSA Open Options LANSA Open Translation Table Back without saving Back	Example Partition Definition Example User/Task (PCXUS Visual LANSA Framework Enable for the Web LANSA Client field and file Personnel System Demons Run demonstration	definitions
Export Settings		< Back Install Cancel

Start the Visual LANSA Independent System which is to be connected to your VCS Master. If it's a completely new install the logon will display something like this:

Visual LANSA	Logon	
Partition DEM	Current Language English Français	<u>System Initialization</u> Partition Initialization
ОКСа	ncel Help	

Click OK.

Create a new Partition with the same name as the source system. Lets say its called DEX (New/Partition). This partition is going to be replaced so the actual settings do not matter. You just need to be able to select it on the logon dialog.

So close the Development Environment and select DEX and click OK.

Visual LANSA Lo	ogon	×
Partition DEM DEX	Current Language English	<u>System Initialization</u> Partition Initialization

Leave that system running whilst you prepare the export from your source system.

Note that the export may be produced by an IBM i too

On the Source System (which contains the COMPLETE and up-to-date partition), use the Deployment Tool to export your whole partition to your

Independent System. Use the XEXPORT template to create an application and check the settings as shown below. No other configuration is required. Save the Package and Build it.

Settings	
Parameter Value	
Deploy with LANSA Client Support	
Deploy to Client without local database	
Deploy to Development System	
Deploy System Definition	
Deploy Partition Definition	
Omit Object Definitions	
Deploy Execution Database Support	
Deploy Execution Add-Ons	
Deploy with Component Support	
Deploy with Web Environment Support	
Deploy LANSA Communications	
Deploy Server Support	
Deploy as System Application	
Delete matching Partition Application	
Deploy with Translation List	
Omit the Application Compiled Objects	
Export Whole Partition	
OK Cancel	

Once the Package is built, close the Package and go to the Package Folder. Now navigate to the internal directory of the package you have built. Something like this: C:\PROGRAM

FILES\SlaveSystem\X_WIN95\X_LANSA\X_APPS\EXPORT\X_PKGWRK\6. In this path, C:\PROGRAM FILES\SlaveSystem\ is where LANSA is installed

X_WIN95 X_LANSAX_APPS is the location of all packages.

EXPORT\ is the name of the application

6.0.0\ is the version number you chose for the Package.

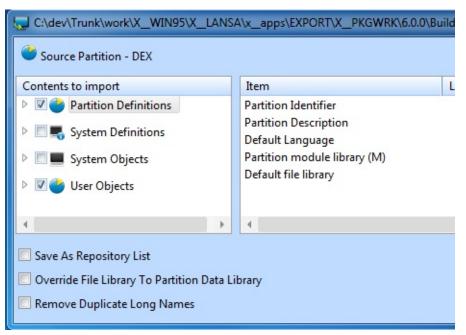
Note: Choosing the Go To Folder option from the Deployment Tool will take you straight to C:\PROGRAM

FILES\SlaveSystem\X_WIN95\X_LANSA\X_APPS\EXPORT.

Copy the path ready for the import.

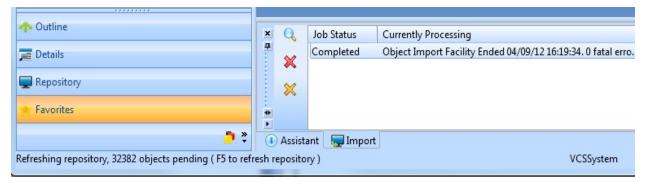
Return to the Visual LANSA Independent System which is to be connected to your VCS Master.

Start the LANSA Import. Paste the path into the Filename field and click open, then double click the lxxdir.del file. A dialog like the following will be displayed. Check Partition Definitions and User Objects. The System Objects are unnecessary and you MUST NOT import the System Definitions. Click Import.



When the import completes, close down the Development Environment and restart it.

Note that once the Job Status is completed, you do not need to wait for the Refreshing Repository to complete nor do you need to press F5. So at this point you can close the Development Environment:



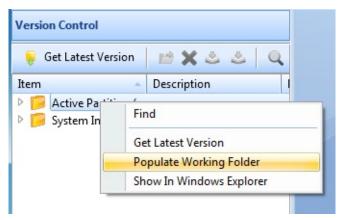
When you restart the Development Environment you will be prompted to initialize the partition if the import came from an older version of LANSA. Choose the options you need to use.

Populate VCS Working Folder

Once the Development Environment is displayed, show the Version Control tab by clicking on this icon:

•	•
🛧 Outline	
肩 Details	
🕎 Repository	
🚖 Favorites	
	(• •)
Ready	\sim

Right click on the Active Partition and click Populate Working Folder.



Then click Ok on the resulting dialog.

All the objects in the repository will be saved into the VCS Working Folder. This can take quite a while for a large partition and it must complete, though it can be stopped and restarted if required. It will continue from where it left off. Repeat for the System Information. This is usually much quicker.

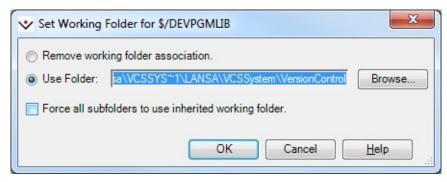
Pointing the Version Control System at the LANSA VCS Working Folder

Now you need to point the VCS at the LANSA VSC Working Folder; how you perform this step will depend on your Version Control System. The example used here is Vault from SourceGear. The use of this as an example is not an endorsement that this particular VCS will suit your development and management style.

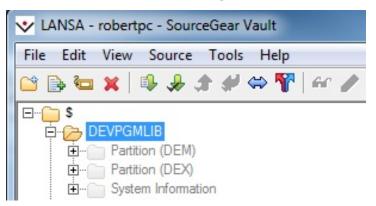
Create a Folder in your VCS to hold the entire LANSA Repository. In this example the Slave System was connected to an IBM i Master that was installed in the program library DEVPGMLIB. To indicate this relationship the folder has

been named after the program library.

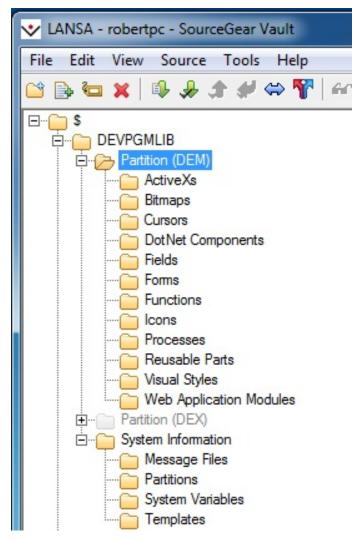
Now go to the LANS Development Environment and on the VCS tab click on the Active Partition and choose Show in Windows Explorer. Navigate up one level to the VersionControl directory and then copy the whole of the path. Paste this into the working folder in the VCS:



And you should see the LANSA directories ghosted in the VCS Folder:

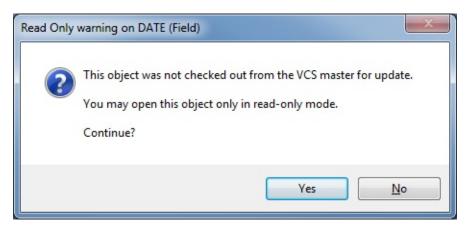


Right click on each ghosted folder and choose Add this Folder. You'll see something like this:



Right click on DEVPGMLIB and choose Get Latest Version. This ensures the file states are up to date. In particular it sets all files to read-only.

LANSA and the VCS are now linked up together. All objects will be unavailable to edit now as they are not checked out from the VCS. (This presumes you use the Check Out/Edit/Check In concurrent development style of version control) If you open an object it will display this message:



If you check it out from the VCS you will be allowed to edit it.

If you use the concurrent development style Edit/Merge/Commit, then of course the objects are always editable and you must resolve the status of the objects using the VCS.

[↑]4.1 VCS Master Setup

4.1.3 Setting up 2nd and Subsequent Developers to use a VCS Master

These instructions describe how to setup a 2nd and subsequent developers to use A VCS Master. It presumes that the VCS already contains the LANSA Repository.

Note: the most efficient way of installing the Visual LANSA Independent System to a VCS Master is to perform a Custom Install and on the options for Initialize Database and Partitions only chose Example Partition Definition. This is because the whole partition will be replaced anyway. Other partition initialization like Visual LANSA Framework need to be done after the partition has been replaced with the VCS Master data:



Either the install will finish here or start the Visual LANSA Independent System which is to be connected to your VCS Master. If it's a completely new install the logon will display something like this:

Visual LANSA I	Logon	
Partition	Current Language English Français	<u>System Initialization</u> Partition Initialization
OK Can	cel Help	

Click OK.

Populate VCS Working Folder

Once the Development Environment is displayed, show the Version Control tab by clicking on this icon:

•	•
< Outline	
📠 Details	
Repository	
- Francisco	
🔶 Favorites	-
Ready	
Ready	

Right click on the Active Partition and click Show in Windows Explorer:

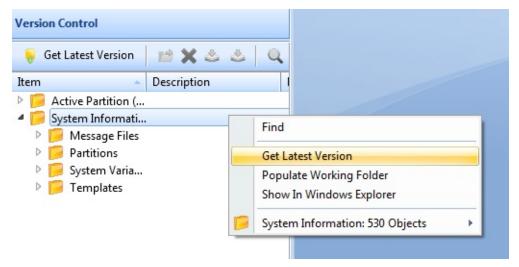
Version Control	
😽 Get Latest Versi	on 📄 🗙 🕹 💩 🔍
Item	 Description
 Active Partit System Information 	Find
	Get Latest Version
	Populate Working Folder
	Show In Windows Explorer

Navigate up one level to the VersionControl directory and then copy the whole of the path. Paste this into the working folder in the VCS:

V Set Working Folder for \$/DEVPGMLIB		
Remove working folder association.		
Use Folder: Nansa\VCSPeer\LANSA\VCSPeer\VersionControl Browse		
Force all subfolders to use inherited working folder.		
OK Cancel <u>H</u> elp		

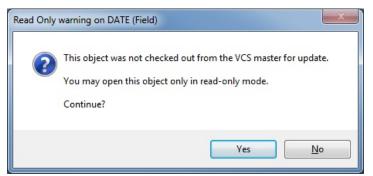
Right click on DEVPGMLIB and choose Get Latest Version.

Go back to the Visual LANSA Development Environment and perform a Get latest Version on the System Information. This will bring in, amongst other data, all the partitions that are in the VCS Master.



Close down the Development Environment and login to the new partition. Perform a Get Latest Version on the Active Partition.

LANSA and the VCS are now linked up together. All objects will be unavailable to edit now as they are not checked out from the VCS. (This presumes you use the Check Out/Edit/Check In concurrent development style of version control) If you open an object it will display this message:



If you check it out from the VCS you will be allowed to edit it.

If you use the concurrent development style Edit/Merge/Commit, then of course the objects are always editable and you must resolve the status of the objects using the VCS.

[↑]4.1 VCS Master Setup

4.1.4 Release Management with a VCS Master

A VCS Master provides the ability to maintain multiple releases of an application by branching the code base when the application is released. This allows fixes to released software to be completely isolated from new development allowing for more stable maintenance releases. Many VCS also semi-automate the task of merging the changes from the latest version to a released version, or vice versa, making it a practical form of software control.

The act of branching your software also requires all developers who will maintain that release of software to install a new LANSA System. This will be in a separate folder using a separate LANSA Repository. Follow the instructions in 4.1.3 Setting up 2nd and Subsequent Developers to use a VCS Master.

Note that it is particularly important to keep the Build PC stable. This means changing as little as possible on that machine.

Also See

Build Machine Compiler Settings ↑4.1 VCS Master Setup

4.1.5 Upgrading a VCS Master

Upgrading an existing Independent Windows Workstation that uses a VCS Master is a matter of ensuring that all developers using the VCS Master upgrade at the same time. Refer to Upgrade LANSA on Windows.

You may need to consider the requirement to maintain released software that was built with a prior version of LANSA. If this is the case you will need to keep the current installation and install a new system to develop your next release. This will be in a separate folder using a separate LANSA Repository.

On the VCS Master you will also need to have branched the released software into a separate location. The VCS working folder of the released system will point to the current LANSA System. See 4.1.4 Release Management with a VCS Master.

4.2 System Initialization

To access the System Initialization dialog, in the Visual LANSA Logon dialog, press the *System Init*... button **before** you press the *OK* button.

System initialization must be performed when a new Visual LANSA system is being launched or when updates are required to ANY of the System Definition items listed below.

The LANSA System Type is selected when the Visual LANSA software is installed. For a definition of the different types of systems you can install, refer to Planning Your Site.

Options

The options shown on the System Initialization dialog depend upon the 4.2.2 LANSA System Type. A Master or Independent System has different options available to those on a Slave System. The System Initialization type will be selected for you automatically.

🖻 Visual LANSA System Initialization 🛛 🛛 🔀	Visual LANSA System Initialization
LANSA System Type	LANSA System Type
C Master/Independent System	Master/Independent System
	C <u>S</u> lave System
Connect to Master	Connect to Master
Connection Options	Options
Main Imports Location of Imports C Local Disk Master	Main Imports Location of Imports C Local Disk C Master
RDML Command Definitions	RDML Command Definitions
🔽 Standard Application Templates	Standard Application Templates
🔽 Standard Built In Function Definitions	Standard Built In Function Definitions
🔽 Standard System Message File	🗖 Standard System Message File
🔽 Standard System Variables (Local Disk)	🔲 Standard System Variables (Local Disk)
🔽 Standard Web Definitions (Local Disk)	🔲 Standard Web Definitions (Local Disk)
Enrolled PC Names (Master's)	Example User/Task (PCXUSER/PCXTASK)
🔽 Enrolled PC Users (Master's)	Example Partition Definition (DEM)
🔽 Current Task List (Master's)	
Partition Definitions (Master's): DEM	
OK Show Last Log Messages Cancel Help	OK Show Last Log Messages Cancel Help

You will need to request a System Definition update when:

- User profiles have been created or modified on an IBM i Master. Refer to 4.2.1 Restricted System Initialization when another user is logged into Visual LANSA.
- You have your own customized Templates, BIF definitions or Messages created or modified on an IBM i Master. Refer to 4.2.23 Show Last Log.
- New or additional partitions are to be accessed.
- New network workstations have been added.
- Task IDs have been created or modified.

On a minimum system configuration, updating all of the above features, including a new partition will only take a few minutes.

You may select from the following options:

4.2.2 LANSA System Type

4.2.9 Standard System Variables

4.2.3 Connect to Master	4.2.10 Standard Web Definitions
4.2.4 Location of Imports	4.2.11 Example User/Task
4.2.20 Use Windows credentials	4.2.12 Example Partition
4.2.5 RDML Command Definitions	Definition
4.2.6 Standard Application Templates	4.2.13 Enrolled PC Names
4.2.7 Standard Built-in Function	4.2.14 Enrolled PC Users
Definitions	4.2.15 Current Task List
4.2.8 Standard System Message File	4.2.16 Partition Definitions

System Initialization and Partition Initialization replace Partition QuickStart, PLUGIN and REFRESH. PLUGIN and REFRESH are still available from the Settings and Administration folder if you prefer this method, but they are not required. The System Initialization dialog uses the same technology as REFRESH and PLUGIN and it is simpler to use.

Connection Tab

A Connection tab is displayed when your PC is a Slave System. Refer to 4.2.2 LANSA System Type.

4.2.17 Master's LU Name
4.2.18 Master's User Name
4.2.19 Master's Password
4.2.20 Use Windows credentials
4.2.21 Client -> Server
Translation Table
4.2.22 Server -> Client
Translation Table

Visual LANSA System Initial	ization 🛛 🛛 🛛
LANSA System Type Master/Independent System Slave System Connect to Master	
Connection Options	
Master's LU Name: Master's User Name:	LANSA3 V11pgmlib
Master's Password:	
Client->Server Translation Table: Server->Client Translation Table:	ANSEBC1140 EBC1140ANS
OK Show Last Log Mess	ages Cancel Help

- 4.3 Partition Initialization
- 3.3 Logon Parameters

4.2.1 Restricted System Initialization when another user is logged into Visual LANSA

Visual LANSA allows a restricted System Initialization when the repository is being used. This enables new users to be added without current users needing to logoff of Visual LANSA. When the repository is being used and System Initialization is selected, this message will be displayed:

LII0162W		×
?	LII0162W You have requested that the database be initialized or it has been automatically detected that the database must be initialized, but it is being used by another. Only PC Names and PC Users will be available for initialization.	
	OK Cancel Help	

If you click OK, only the PC Names and PC Users will be enabled and will already be selected:

Stisual LANSA System Initialization
LANSA System Type
C Master/Independent System
Slave System
Connect to Master
Connection Options
Main Imports
Location of Imports Local Disk
C Master
BDML Command Definitions
 Standard Application Templates Standard Built In Function Definitions
Standard System Message File
Standard System Variables (Local Disk)
Standard Web Definitions (Local Disk)
Enrolled PC Names (Master's)
Enrolled PC Users (Master's)
Current Task List (Master's)
Partition Definitions (Master's): DEM
OK Show Last Log Messages Cancel Help

Click OK to update the PC Names and PC Users. ↑ 4.2 System Initialization

4.2.2 LANSA System Type

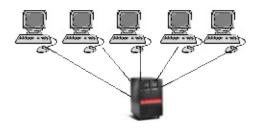
Specify whether the current LANSA System Type is a Master/Independent or a Slave System.

The LANSA System Type is chosen when the Visual LANSA Software is installed. Refer to the Planning a New Installation in the *Installing LANSA on Windows Guide* for the available configurations. If you wish to change the System Type, you must re-install Visual LANSA. In brief:

If you are not connected to an IBM i, the system type is **Master/Independent System**. This includes systems that are connected to a VCS Master.



If you are connected to an IBM i for the LANSA System Definitions and Master Repository, then your system type is a **Slave System**.



Also See

Master and Slave Systems Visual LANSA Master/Independent System Administration Visual LANSA Slave System Administration ↑ 4.2 System Initialization

4.2.3 Connect to Master

If your 4.2.2 LANSA System Type is a Slave System, then specify the IBM i Master Repository to connect to to retrieve the LANSA System Definitions. This option and the 4.2.4 Location of Imports option, on the Options Tab of the LANSA Editor, determine the source of the information used to retrieve 4.2.5 RDML Command Definitions, 4.2.6 Standard Application Templates, 4.2.7 Standard Built-in Function Definitions and 4.2.8 Standard System Message File.

Visual LANSA System Initialization	×
LANSA System Type C Master/Independent System Slave System Connect to Master	
Connection Options	
Master's LU Name:LANSA3Master's User Name:V11pgmlibMaster's Password:******Use Windows Credentials*****Client->Server Translation Table:ANSEBC1140Server->Client Translation Table:EBC1140ANS	
OK Show Last Log Messages Cancel Help	

If you connect to an IBM i Master, you will need to specify the Connection details:

- 4.2.17 Master's LU Name
- 4.2.18 Master's User Name
- 4.2.19 Master's Password

4.2.21 Client -> Server Translation Table
4.2.22 Server -> Client Translation Table
↑ 4.2 System Initialization

4.2.4 Location of Imports

The source of the main imports will be the Local Disk or Master.

This option will determine the source of the information used to retrieve 4.2.5 RDML Command Definitions, 4.2.6 Standard Application Templates, 4.2.7 Standard Built-in Function Definitions and 4.2.8 Standard System Message File.

If you select Local Disk, the imports on your local drive will be imported. These imports were stored on your local drive when the LANSA Software was last installed or upgraded.

If your 4.2.2 LANSA System Type is a Slave System, you should import from the Master LANSA System. You must select the 4.2.3 Connect to Master and specify the Connection details.

1 4.2 System Initialization

4.2.5 RDML Command Definitions

This option imports all LANSA RDML and RDMLX command definitions. This option must be selected when a new system is first initialized or after a system has been upgraded to a new version.

1 4.2 System Initialization

4.2.6 Standard Application Templates

This option imports all standard application templates.

This option must be selected when a new system is first initialized or after a system has been upgraded to a new version.

Note: A large amount of information is transferred when you select application templates. Do not transfer these objects unless you specifically need to.

↑ 4.2 System Initialization

4.2.7 Standard Built-in Function Definitions

This option imports all standard built-in definitions.

This option must be selected when a new system is first initialized or after a system has been upgraded to a new version.

1 4.2 System Initialization

4.2.8 Standard System Message File

This option imports the standard message file.

This option must be selected when a new system is first initialized or after a system has been upgraded to a new version.

Note: A large amount of information is transferred when you select system message file. Do not transfer these objects unless you specifically need to.

↑ 4.2 System Initialization

4.2.9 Standard System Variables

This option imports the standard system variables from the Local Disk. This option must be selected when a new system is first initialized or after a system has been upgraded to a new version.

4.2.10 Standard Web Definitions

This option imports standard technology services and default web pages from the Local Disk.

Select this option if you are developing LANSA for the Web applications. This option can be added at any time.

4.2.11 Example User/Task

This option imports the definition for the example LANSA User ID PCXUSER with Password PCXUSER, and the example task PCXTASK.

This option must be selected when a new system is first initialized. You can logon using the LANSA Development Environment with this User ID and task and then create other users as required.

This option is only used with 4.2.2 LANSA System Type of *Master/Independent System*.

Also See

Visual LANSA Independent System Administration

- 3.3 Logon Parameters
- 4.2.12 Example Partition Definition
- 1 4.2 System Initialization

4.2.12 Example Partition Definition

This option imports the example partition definition, DEM.

This option must be selected when a new system is first initialized. You can start using the LANSA Development Environment with the DEM partition and then create other partitions as required.

This option is only used with 4.2.2 LANSA System Type of Master/Independent

Also See

•

Visual LANSA Independent System Administration

3.3 Logon Parameters

4.2.11 Example User/Task

1 4.2 System Initialization

4.2.13 Enrolled PC Names

This option imports enrolled PC Names from the Master LANSA System.

This option must be selected when a new system is first initialized. It must also be used whenever new PCs have been enrolled or their profiles changed in the Master LANSA System.

This option is only used with 4.2.2 LANSA System Type of Slave Systems.

Also See

Visual LANSA Slave System Administration 4.2.14 Enrolled PC Users 4.2.15 Current Task List

1 4.2 System Initialization

4.2.14 Enrolled PC Users

This option imports enrolled PC Users from the Master LANSA System.

This option imports the table that relates PC user names to the Master LANSA System's user names, group names and their passwords.

This option must be selected when a new system is first initialized. It executed again whenever new users have been enrolled or profiles have changed in the Master LANSA System.

This option is only used with 4.2.2 LANSA System Type of Slave Systems. When using a Master/Independent System, user profiles are defined and maintained locally so their definitions do not need to be imported.

Also See

Visual LANSA Slave System Administration 4.2.13 Enrolled PC Names 4.2.15 Current Task List ↑ 4.2 System Initialization

4.2.15 Current Task List

This option imports the current task list from the Master LANSA System.

This option must be selected when a new system is first initialized. It must also be selected whenever Task IDs have been created or changed in the Master LANSA System.

This option should be used frequently to ensure Task information is up to date.

This option is only used with 4.2.2 LANSA System Type of Slave Systems. When using a Master/Independent System, task lists are defined and maintained locally so their definitions do not need to be imported.

Also See

4.2.16 Partition Definitions

Specify the identifier of the partition on defined on your IBM i Master LANSA system that you wish to add or update to your Slave Repository (e.g. DEM or SYS).

This option may be selected whenever a partition is created or when a partition definition has been updated in the Master LANSA System. For example, if new languages are added to a partition, the definition must be imported again.

This option is only used with 4.2.2 LANSA System Type of Slave Systems. When using a Master/Independent System, partitions are defined and maintained locally so their definitions do not need to be imported.

Also See

Visual LANSA Slave System Administration

4.3 Partition Initialization

4.2 System Initialization - Connection Tab

↑ 4.2 System Initialization

4.2.17 Master's LU Name

Specify the server name that hosts the Master LANSA Repository you are connecting to. The name will default to server name that you last connected to.

The Master's LU Name must be defined in the LANSA Communications Administrator table with the appropriate connection information.

For example, an IBM i Partner LU (logical unit) server name might be specified as APPN.SYDASDEV or SYDASD25. Generally, in CPI-C systems, the name must be formatted APPN.<name>. In Enhanced APPC systems it must be specified as just <name>.

This option is only used with 4.2.2 LANSA System Type of Slave Systems.

- 4.2.3 Connect to Master
- 4.2.18 Master's User Name
- 4.2.19 Master's Password
- ↑ 4.2 System Initialization

4.2.18 Master's User Name

Specify the User ID you wish to use to logon to the server hosting the Master LANSA Repository. The name will default to the last value you entered on the login dialog.

For an IBM i server, there are some significant things to note about this User ID:

- It should be enrolled as a valid user of the system distribution directory. Refer to the built-in function DEFINE_OS_400_SERVER in the LANSA Technical Reference Guide for more details.
- The job description associated with it should have the LANSA for iSeries program library (e.g. DC@PGMLIB) in its initial library list. This means that when you sign on to an IBM i, the LANSA program library (e.g. DC@PGMLIB) should be immediately available in the job's library list.
- It should be no more than 8 characters in length.

- 4.2.3 Connect to Master
- 4.2.17 Master's LU Name
- 4.2.19 Master's Password
- 1 4.2 System Initialization

4.2.19 Master's Password

Specify the password associated with the 4.2.18 Master's User Name. The password will default to the last value you entered on the login dialog.

- 4.2.3 Connect to Master
- 4.2.17 Master's LU Name
- ↑ 4.2 System Initialization

4.2.20 Use Windows credentials

If Single Sign-On (SSO) is in use, you can select this option if you wish to use your your User ID and Password when you logged on to Windows, rather than the User Id and Password on the Logon dialog.

If you are using a slave system with an IBM i Master Repository, you may need to update the Windows user details using the 4.2.14 Enrolled PC Users option.

If you are using a Visual LANSA Master or independent system, you can maintain user profiles by opening the appropriate user in the LANSA Editor.

Also see

Edit User Definitions in the Visual LANSA User Guide 4.4 Single Sign-On (SSO)

↑ 4.2 System Initialization

4.2.21 Client -> Server Translation Table

Specify the Client-to-Server translation table name to be used. The default will be the value entered in the Language Options dialog during the installation.

You should have no need to change this name unless you have specific instructions from your product vendor.

Also See

- 4.2.3 Connect to Master
- 4.2.22 Server -> Client Translation Table
- ↑ 4.2 System Initialization

4.2.22 Server -> Client Translation Table

Specify the Server-to-Client translation table name to be used. The default will be the value entered in the Language Options dialog during the installation.

You should have no need to change this name unless you have specific instructions from your product vendor.

Also See

- 4.2.3 Connect to Master
- 4.2.21 Client -> Server Translation Table
- ↑ 4.2 System Initialization

4.2.23 Show Last Log

The result of the last LANSA import will be displayed in the default editor (Notepad).

The log will display details of the last import including all status, warning and error/failure messages.

4.3 Partition Initialization

To access the Partition Initialization dialog, in the Visual LANSA Logon dialog, enter your *User ID* and *Password* or select *Use Windows Credentials* and press the *Partition Init...* button **before** you press the *OK* button.

Partition initialization must be performed for each new partition created, on each PC that has its own local Repository, including slave workstations and independent workstations. Visual LANSA Client PCs co-operating in a Network Install do not require Partition Initialization.

Partition Initialization
Mandatory Partition Initialization
Visual LANSA Framework
Enable for the Web
LANSA Client field and file definitions
Personnel System Demonstration material
Run Demonstration
OK Show Last Log Messages Cancel Help

The development environment will not allow a partition to be used by a PC until the mandatory *Partition Initialization* has been done. Partition initialization is automatically displayed when you first create or access a new partition.

You may use *Partition Initialization* to add or update options in an existing partition. For example, you may not have included the Visual LANSA Framework when the partition was first initialized. At a later date, you can select partition initialization again to add the Visual LANSA Framework to the partition. Note that you will not have a copy of the *Visual LANSA Framework User Guide* until you import it using the Visual LANSA Framework software using this dialog.

You may select the following options:

4.3.1 Mandatory Partition Initialization	4.3.4 LANSA Client field and file definitions
4.3.2 Visual LANSA Framework	4.3.5 Personnel System Demonstration
4.3.3 Enable for the Web	4.3.6 Run Demonstration

When you have initialized a partition, you will be provided with a report, if any

errors have occurred. Refer to 4.3.7 Show Last Log for further information. When the first time a Partition Initialization is performed, the following dialog is displayed:

Language	Windows codepage	Windows codepage origin	System i codepage	System i codepage origin	
CAT	01252	LANSA partition	00500	LANSA partition	
DEU	01252	LANSA internal CCSID table	00273	IBM language default	
DUT	01252	LANSA internal CCSID table	00037	IBM language default	
ENG	01252	CCSID.txt	01140	LANSA derived default	
FIN	01252	LANSA internal CCSID table	00278	IBM language default	
FRA	01252	CCSID.txt	01147	LANSA derived default	
GRE	01253	LANSA internal CCSID table	00875	IBM language default	
HEB	01255	LANSA internal CCSID table	00424	IBM language default	
HP1	01252	System codepage	01140	Job CCSID	
HPL	01252	System codepage	01140	Job CCSID	
ITL	01252	LANSA internal CCSID table	00280	IBM language default	
JPN	00932	LANSA partition	05035	LANSA partition	
LLL	01252	System codepage	01140	Job CCSID	
NOR	01252	LANSA internal CCSID table	00277	IBM language default	
POR	01252	LANSA internal CCSID table	00037	IBM language default	
ROB	01252	System codepage	01140	Job CCSID	
5CHI	00936	CCSID.txt	00935	IBM language default	
5FRA	01252	LANSA internal CCSID table	00500	IBM language default	
5WE	01252	LANSA internal CCSID table	00278	IBM language default	
	00950	CCSID.txt	00937	IBM language default	1

This dialog lists the codepages and CCSID that are used for multilingual text conversions. Principally this is used when exporting from IBM i to Windows and when using the Host Monitor, but it is also used when exporting from Windows to another Windows repository. Refer to Language Options for more information on this process.

The purpose of the dialog is to

- a) make you aware that LANSA has to make these mapping decisions and,
- b) precisely what decisions LANSA has made in assigning codepages and CCSID to each language.

It is only shown the very first time that a Partition Initialization is performed. If this dialog is not shown at that time then it is possible that a communications error has prevented the retrieval of the CCSID mappings from the server. In this case, an error message will have been added to the import log. To fix the problem, check the listener is started and look in the job logs on the server for further information. If the CCSID mappings were retrieved successfully you will find them in **langmap.txt** for each partition and language in the *Installation Details* tab of the Product Information which you open from the Visual LANSA Editor.

Also See

4.2 System Initialization

3.3 Logon Parameters

4.3.1 Mandatory Partition Initialization

This option is mandatory for a new partition. It makes the partition usable. If this option has not been selected at least once, you cannot use the partition.

Partition Initialization imports Groups, Frameworks, System Fields and Component primitives. It also ensures that the execution environment is initialized to be able to build objects for the partition. For example, if the DEM partition is being initialized, the directories are created under $x_win95\x_lansa\x_dem$

This option is required to be executed once, when a partition is first created.

Also See

- 4.2.16 Partition Definitions
- ↑ 4.3 Partition Initialization

4.3.2 Visual LANSA Framework

This option installs the Visual LANSA Framework. It can be added at any time.

The Visual LANSA Framework can be used to create Windows, client/server and Web-based applications.

No special license is required to use the Visual LANSA Framework.

If you are developing Web-based applications with the framework, you must have a valid LANSA for the Web development license.

The 4.3.3 Enable for the Web option should also be selected if you are creating a Web-based application.

The Visual LANSA Framework is not required for developing Web Application Module (WAM) applications.

Also See

Visual LANSA Framework Guide

4.3.3 Enable for the Web

This option initializes the partition to support LANSA for the Web. It enables the partition for Web usage by importing the default Web Pages into the partition and imports support for the Web Utilities. It can be used at any time.

If you are developing Web applications using the 4.3.2 Visual LANSA Framework, then you must select this option.

Reminder: You must have a valid LANSA for the Web development license to create Web applications with Visual LANSA.

4.3.4 LANSA Client field and file definitions

This option installs the field and file definitions for use by LANSA Client. It can be selected at any time.

4.3.5 Personnel System Demonstration

This option installs the Personnel System Demonstration application. It can be selected at any time.

This application must be installed to use the online tutorials.

You may also use this option to reinstall the Personnel System Demonstration application. It will reset the application to its original form.

If the partition is Enabled for RDMLX, the import of the Personnel System Demonstration application will include Full RDMLX objects.

Also See

4.3.6 Run Demonstration

4.3.6 Run Demonstration

This option specifies that the Personnel System Demonstration application will be executed after the partition initialization is complete.

The 4.3.5 Personnel System Demonstration option must also be selected.

You may wish to execute the demonstrations application to become familiar with the sample database or to test your new partition.

4.3.7 Show Last Log

The result of the last LANSA import will be displayed in the default editor (Notepad).

The log will display details of the last import including all status, warning and error/failure messages.

4.4 Single Sign-On (SSO)

Prior to the introduction of a Single Sign-On (SSO), LANSA users had to supply a user name and password when connecting to each Windows and IBM i system. Single Sign-On gives users access to multiple computer systems within an organization after signing on only once.

Whether to use the Single Sign-On option is specified by selecting the *Use Windows credentials* option on the *Visual LANSA Logon* dialog, or the *System Initialization* dialog.

The concept of Single Sign-On is to allow a user who is logged onto Windows to have their Windows credentials silently authenticated when they wish to use i5/OS machines.

The two key technologies that underpin the SSO mechanism are the **Kerberos Network Authentication Protocol**, and the **IBM i Enterprise Identity Mapping (EIM)** mechanism. These technologies must be understood and in use before using Single Sign-On with LANSA.

The necessary software and set up **must be completed and fully tested** before LANSA's SSO can be used. It is beyond the scope of the LANSA documentation to explain how to configure these two technologies.

Set up Single Sign-On

Following are the basic steps you will follow:

1. Ask your system administrator to configure your IBM i for Single Sign On from your Windows domain (ensure that the HOST principal name is added to the keytab file), and also to configure EIM on your IBM i to map each required Windows domain user to a corresponding IBM i user profile. Note that these must be working and tested before continuing with the next step.

The LANSA listener job user, its group profile or its Supplemental group profile must have the following authorities to the directories and files listed below:

Note: The names used may be different in your system

Configuration file requires data authority of *R and the path must have data authority of *X

/QIBM/UserData/OS400/NetworkAuthentication/krb5.conf

Credential cache file requires data authority of *RW and the path must have data authority of *X

/QIBM/UserData/OS400/NetworkAuthentication/creds/krbcred_

Keytab file requires data authority of *R and the path must have data authority of *X

/QIBM/UserData/OS400/NetworkAuthentication/keytab/krb5.ke

- 2. On the IBM i, run the LANSA CONFIGURE command, and choose the COMMS_EXTENSIONS facility to setup COMMS_EIM_USER with the username and password of an LDAP user authorized to query EIM. This step needs to be done only once per LANSA system on the IBM i.
- 3. Stop and restart the Listener job before continuing.
- 4. Repeat these next steps for each user to be included in Single Sign-On.
 - a. Assuming that one of the mappings set up in EIM maps from, say, Windows domain user user1@MYDOMAIN.COM to LANSA user DEVUSER, log onto Windows as user1@MYDOMAIN.COM.
 - b. Start Visual LANSA, and from the Logon dialog, perform a *System Initialization* using the user name and password of the LANSA user DEVUSER (as per example). It is necessary to do this at least once for a LANSA user before the *Use Windows credentials* option may be used to perform a Single Sign-On as that user.
 - c. When System Initialization is complete, check(select) the *Use Windows credentials* option and click *OK* to log on. Any values in the User ID and Password are ignored.

If the logon fails and a message box appears with the message "User user1@MYDOMAIN.COM specified is not known to LANSA", then this indicates that one of the above steps may not have been completed successfully.

Also see

- 4.4.1 How LANSA SSO Works
- 3.3.3 Use Windows Credentials in 3.3 Logon Parameters
- 4.2.20 Use Windows credentials in 4.2 System Initialization

4.4.1 How LANSA SSO Works

When a Windows user signs onto Windows as part of a domain, their domain user account in the Windows Active Directory includes a property called the Kerberos *User Principal Name* (UPN). The UPN of a user consists of the user name, followed by an '@' character, and then the full domain in uppercase letters. For example, the UPN for Windows user 1 (user1) might be user1@MYDOMAIN.COM.

When user1@MYCOMAIN.COM launches Visual LANSA, and if the *Use Windows credentials* option is selected in the Logon dialog, Visual LANSA Logon checks whether the repository contains a LANSA User which is associated with user1@MYCOMAIN.COM. If it finds such a LANSA User, for example "DEVUSER", then Visual LANSA Logon starts the Visual LANSA session using the LANSA User Id of DEVUSER. If there is no association, the log on step cannot proceed.

The association between a Windows domain user and a LANSA User is specified on an IBM i by an IBM i administrator using the IBM Enterprise Identity Mapping (EIM) facility. In order to automate the access to the IBM i EIM facility, a Distinguished name and password are needed. These are specified using the LANSA Communications Extensions Configuration Items (COMMS_EXTENSIONS) facility, described in the LANSA for iSeries User Guide.

If you are using a Slave System with an IBM i Master Repository, you may need to perform a 4.2 System Initialization and select the 4.2.14 Enrolled PC Users option, to update the association details in the Visual LANSA System Definition. This option will retrieve the most current list of associations between Windows domain users and authorized LANSA User Ids.

The association between a Windows domain user and a LANSA User is specified on a Windows server using the LANSA User definition in the LANSA Editor.

Also see

Edit User Definitions

EIM Authorized User (COMMS_EIM_USER) in the LANSA for iSeries User *Guide*.

5. Remote Systems

LANSA Remote System Monitors are used to move LANSA objects and system information between systems. Review the following topics:

5.1 Concepts

5.2 Host Monitor

5.3 Other Remote System Monitors

5.4 Job Status and Messages

Also See

Remote Systems

5.1 Concepts

In order to understand how LANSA moves objects and system information between LANSA systems, review the following topics:

- 5.1.1 Remote System Terms
- 5.1.2 Host Monitor Concepts
- 5.1.3 Deliver To Concepts

Also See

- 5.2.4 Start and Stop the Host Monitor
- 5.3.1 Define a Deployment System
- ↑ 5. Remote Systems

5.1.1 Remote System Terms

The following table lists the various terms used:

Master system	The system that owns the LANSA object and system definitions. For a further definition, refer to Master and Slave System Configurations. Only an IBM i LANSA system may be a Master system.
Slave system	A Visual LANSA system that is linked to a Master system. For further definitions, refer to: Master and Slave System Configurations. Only a Windows Visual LANSA system may be a Slave
	system.
Independent system	A Visual LANSA system that does not link to a Master or act as a server. For definitions, refer to:
	Independent Windows Server
	Local Client to an Independent Windows Server
	Network Client to an Independent Windows Server
	Independent Windows Workstation to a VCS Master.
Deployment system	A Deployment System is a remote system that receives object and system definitions from a Visual LANSA System. A Deployment System is defined in the Visual LANSA System which connects to and delivers objects to the Deployment System. For details, refer to Promotion and Deployment. Only an IBM i LANSA System or a Linux System may be set up to act as a Deployment System.
Remote System	A Remote System is any LANSA system which the Visual LANSA system can automatically connect to for the purpose of LANSA objects and system definitions. Remote Systems encompass Deployment Systems and the Master System.
Remote System Monitor	Remote System Monitors manage connections and jobs for remote Master or Deployment Systems.
Host	A kind of Remote System Monitor; it is used to move

Monitor definitions between the Slave and the Master.

↑ 5.1 Concepts

5.1.2 Host Monitor Concepts

If you use a Slave system, review the following topics: Connect Master and Slave Systems Host Monitor and Visual LANSA Slave Repositories Access the Master System Example Development Cycle (with Master) ↑ 5.1 Concepts

Connect Master and Slave Systems

When developing with LANSA for iSeries and Visual LANSA, you are developing applications in a distributed development environment, i.e. your LANSA development environment involves more than one machine. It is important to remember that:

- LANSA for iSeries hosts the Master Repository that contains all development objects (your application fields, files, forms, etc.)
- LANSA for iSeries is the Master System that maintains all LANSA System information (partitions, languages, users, tasks, security, settings, etc.)
- One or more Visual LANSA development environments (Slave Systems) may connect to the LANSA for iSeries Master System.

The LANSA Host Monitor is used to connect the LANSA for iSeries with each of the Visual LANSA slave systems. The Repository Synchronization, Propagation, Check In and Check Out facilities are used to move the developed LANSA objects and system information between LANSA systems.

The details of the Master System are required as part of the Visual LANSA System installation when you select a Slave System install. A Remote System definition will be automatically generated for the Master System. Specific details pertaining to the Master System can be maintained using the LANSA Communications Administrator. (Refer to the Communication Setup Guide.)

Before a Visual LANSA Slave System can be used, it must connect to the LANSA for iSeries Master System to be initialized. The System Initialization process will connect to the LANSA for iSeries Master System to transfer the partition, user, security and other system data into the Visual LANSA Slave Systems repository. After System Initialization is completed a user may logon to the Visual LANSA System.

If you are connecting multilingual systems, review Multilingual Text Handling in the LANSA Multilingual Application Design Guide.

The Repository in the Visual LANSA Slave System will be empty (aside from Demonstration objects) until objects are propagated or checked out from the LANSA for iSeries Master System or new objects are created. (Refer to Host Monitor and Visual LANSA Slave Repositories.)

If you are using an Independent Visual LANSA System, you will not use the Host Monitor. All associated tabs and icons will not appear in the LANSA Editor.

Also See

μ

1 5.1.2 Host Monitor Concepts

L,

Host Monitor and Visual LANSA Slave Repositories

LANSA for iSeries stores the Master LANSA Repository and the Visual LANSA Systems contain the Slave Repositories. Again, one or more Visual LANSA development environments may be connected to the LANSA for iSeries System.

The LANSA for iSeries Master Repository is used to store the master definition of all LANSA objects in a partition. As development work is completed on a Visual LANSA slave system the new or modified object should be checked into the Master Repository leaving a read-only copy of the object in the slave repositories. The master definition may then be checked out for update into one of the slave repositories as required. Refer to Example Development Cycle (with Master).

The Visual LANSA Slave Repositories will initially be empty (aside from Demonstration objects). Typically the slave repository is populated by copying the required object definitions from the LANSA for iSeries Master System to the Visual LANSA Slave System. Objects may be copied using the export or check out from master repository features. The objects may be copied as readonly (i.e. the slave systems cannot change the object definition but can view the latest definition from the Master Repository) or the objects may be checked out for update.

Task Tracking is used in LANSA to control access to objects to ensure an object's definition is not being changed by multiple developers at the same time. Repository Synchronization or 5.2.5 Refresh Master Object List is used to make sure that any changes made to the Master System are reflected in the specified Slave Systems.

Also see

Task Tracking

1 5.1.2 Host Monitor Concepts

Access the Master System

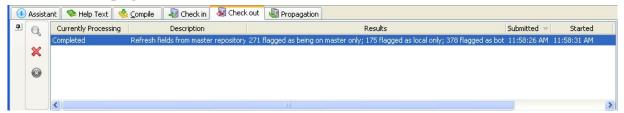
Refresh Master Object List and Repository Synchronization use the LANSA Host Monitor to propagate LANSA objects and LANSA systems information between a LANSA for iSeries Master System and a Visual LANSA Slave System. Specifically, the following tasks may be performed:

- Check Out copies objects from the LANSA for iSeries system to the Visual LANSA system.
- 5.2.2 Check In copies objects from the Visual LANSA system to the LANSA for iSeries system and optionally releases any Task Tracking locks on an object so the object becomes available for other developers to modify.
- Repository Synchronization is an optional feature that allows changes made to a LANSA for iSeries Master Repository to be automatically propagated to all Visual LANSA Slave Repositories in order to maintain current object information. Repository Synchronization is set up on the LANSA for iSeries System.
- 5.2.5 Refresh Master Object List manually initiates a comparison of the Master Repository objects with the locally defined objects.

The LANSA Host Monitor and Repository tab are fully integrated into Visual LANSA. They are not separate applications running along side the LANSA Editor. For example, it is not possible to submit a large check in request and then close LANSA. The Host Monitor will be shut down when LANSA shuts down and all communication with the Master System is terminated.

Visual LANSA incorporates Check In, Check Out and Propagation tabs to detail any movement of objects and information between the LANSA for iSeries Master system and the Visual LANSA Slave systems. Repository

Synchronization and 5.2.5 Refresh Master Object List movements are detailed under the Propagation tab.



Tip: If you are checking out a large number of objects, you may wish to use a LANSA Import.

↑ 5.1.2 Host Monitor Concepts

Example Development Cycle (with Master)

Following is a simple example of a typical development cycle using a LANSA for iSeries Master System with a Visual LANSA Slave System:

- System Initialization is used to prepare a newly installed Visual LANSA Slave System. The initialization will copy the system definition data along with user and task details from the Master System to the Slave System. This step is typically performed during the Visual LANSA installation.
- System Initialization is also used to create a matching partition in a Visual LANSA System. For example, if the LANSA for iSeries has a TRN training partition, the matching TRN partition is created in Visual LANSA using System Initialization.
- When a new partition is defined in Visual LANSA as part of the System Initialization it must be followed by a Partition Initialization to prepare the new partition for use and perform any required standard imports into the partition.
- Repository Synchronization is set up on the LANSA for iSeries Master System so that any changes to objects on the Master System on the IBM i will be automatically propagated to the Visual LANSA Slave Systems.
- A LANSA for iSeries export can be created to move a group of objects from the LANSA for iSeries Repository to the Visual LANSA System. This is particularly efficient when dealing with large numbers of objects. (Refer to Import.)
- If a developer intends to modify an object in Visual LANSA, the object must be exported or checked out for update, and will be locked to a task and to a PC.
- A developer will logon to a partition in Visual LANSA using a Task ID. The developer can update any objects that have been checked out for update with the current task to the current PC. The developer can create objects in Visual LANSA. The developer can view, compile and use the read-only objects in the repository, but read-only objects cannot be changed.
- If a developer wants to modify a read-only object, the developer must 5.2.1 Check Out the object for update from the Master Repository.
- The developer can 5.2.2 Check In new objects and any updated objects to the LANSA for iSeries Master System so that other developers can access the new objects and/or updated object definitions.

• When a change is made to objects in the LANSA for iSeries Master Repository, Repository Synchronization can be used to ensure that the Visual LANSA Repositories remain synchronized.

It is important to note that the following from this example:

- A Visual LANSA System may only have one partition but the LANSA for iSeries Master could have multiple partitions. Visual LANSA must be initialized for any partition where you want to do development with Visual LANSA.
- If system or partition definitions are modified on the LANSA for iSeries Master, system and/or partition initialization may need to be performed again.
- The Visual LANSA Repository does not have to be a complete copy of the LANSA for iSeries Repository. You can export any number of objects from the Master System.
- If you are using Repository Synchronization for a Visual LANSA System, then the complete object lists for the partition will match and system information will constantly be maintained.
- If you are **not** using Repository Synchronization, naming standards must be strictly adhered to, to avoid duplicate objects being created on Slave Visual LANSA Systems.

↑ 5.1.2 Host Monitor Concepts

5.1.3 Deliver To Concepts

Deliver To copies system definitions and objects from a Visual LANSA system to the nominated Remote Deployment System, via a Remote System Monitor. When appropriate, the object is compiled on the Remote System as part of the Deliver To processing.

If you wish to deliver your software to one or more remote systems, review the following topics:

Restrictions and Assumptions

Linux Deployment System Development Cycle Example

IBM i Deployment System Development Cycle Example

↑ 5.1 Concepts

Restrictions and Assumptions

- A LANSA environment must be installed on the Remote System nominated to Deliver To with a compatible version of LANSA software.
- The Visual LANSA user must be authorized to maintain Remote Systems and use Deliver To.
- The Deployment environment must be defined as a Remote System in Visual LANSA before the Deliver To command will be available against the repository objects.
- The Remote System definition must correspond with the LANSA Communications Administrator entry for this LANSA system.
- Delivery will be to a corresponding partition on the Remote Deployment System.

For Remote System of build type IBM i:

- The Partition on the Deployment System must be manually defined and match the partition definition on the Visual LANSA installation.
- Before attempting to deliver any objects to an IBM i remote system you need to Refresh the system to get the workstation names from the Deployment System. Refresh is available as a command in the right click menu against the Remote System object.
- The Deliver To options when delivering to IBM i are the same as standard Check In options.

For Remote System of build type Linux:

- The Partition on the Deployment System must be defined and initialized before partition objects can be delivered; otherwise you can expect Deliver To to fail. Partition Initialization is available as a command in the right click menu against the Remote System object.
- When compilable objects are Delivered To a Remote System of build type Linux they will always be compiled on the target system.
- Objects that are irrelevant on the Deployment system, such as Components (including Forms), which require a Windows GUI, are not available to Deliver To the Remote System.
- Message Files are only available for delivery to Linux Deployment Systems.

↑ 5.1.3 Deliver To Concepts

Linux Deployment System Development Cycle Example

Following is a simple example of a typical development cycle using a Visual LANSA System and a Deployment System on Linux:

- Developers code and test their application on a Visual LANSA installation on Windows.
- LANSA is installed on Linux as described in the Installing Visual LANSA on Linux Guide.
- The Remote Deployment System is defined in Visual LANSA on Windows with a corresponding entry in the LANSA Communications Administrator (refer to 5.3.1 Define a Deployment System).
- The Partition Initialization menu option, available in the Remote System right click menu, is used to initialize the Partition on the Deployment System, that is the Visual LANSA on Linux installation. For example, if the application was developed in the training partition TRN, the matching TRN partition is created on the Deployment System and populated with the partition definition from the Visual LANSA on Windows installation.
- Deliver To is used to deliver application objects, message files and so on to the Deployment System. Only non-Visual objects are required. Visual components, for example Forms, cannot be delivered to the Deployment System, as execution is not supported (or required) on the LINUX environment.

It is important to note that the following from this example:

• The Deployment System does not have to be a complete copy of the Visual LANSA Repository. Only the application related objects are required to be Delivered To the Deployment System.

1 5.1.3 Deliver To Concepts

IBM i Deployment System Development Cycle Example

Following is a simple example of a typical development cycle using a Visual LANSA System and a Deployment System on IBM i:

- Developers code and test their application on a Visual LANSA installation on Windows.
- LANSA is installed on IBM i as described in the Installing LANSA on IBM i Guide.
- Partition is defined on IBM i matching the partition definition on the Visual LANSA installation.
- The Remote Deployment System is defined in Visual LANSA with a corresponding entry in the LANSA Communications Administrator (refer to 5.3.1 Define a Deployment System).
- The Refresh menu option, available in the Remote System right click menu, is used get workstation names from the Deployment System, that is the LANSA on iSeries installation.
- Deliver To is used to deliver application objects to the Deployment System. "Create an IBM i export list" can be selected on the Deliver To dialog to automatically create an export list on the deployment system including the delivered objects.

1 5.1.3 Deliver To Concepts

5.2 Host Monitor

The LANSA Host Monitor is used to move LANSA objects and systems information between a LANSA for iSeries Master System and a Visual LANSA Slave System. This process is described n the following topics:

- 5.2.1 Check Out
- 5.2.2 Check In
- 5.2.3 Propagation
- 5.2.4 Start and Stop the Host Monitor
- 5.2.5 Refresh Master Object List

The Host Monitor is not used by Independent Visual LANSA Systems.

Also See

5.1.2 Host Monitor Concepts

Visual LANSA Slave Administration

↑ 5. Remote Systems

5.2.1 Check Out

Check out is used to copy the definition of an object that is currently stored in the LANSA for iSeries Master Repository into a Visual LANSA Slave Repository.

You can check out an object as Read-only or for Update. In order to check out an object for update, you must have appropriate authority to access the object. (Refer to Using Task Tracking.)

To check out objects, you can use the Repository tab and right click on the object. The check out option will be displayed in the pop-up menu. You may select multiple items to check out.

In order to check out an object from the Master Repository, you must have a current list of the objects in the Master Repository. You can get this list by using 5.2.5 Refresh Master Object List or by using the LANSA Import.

Warning If you simply want to make a local copy of part or all of an application from the master repository, use export and import. It will be much quicker than using Check Out. Check Out processing has not been designed or optimized for this type of use.

Also See

Check Out Options 5.2.2 Check In 5.2.4 Start and Stop the Host Monitor ↑ 5.2 Host Monitor

Check Out Options

When objects are selected to be checked out they are automatically flagged to be checked out for Update. Use the Read only and Write (Update) options in the toolbar to indicate how each object, or group of objects should be checked out. The Read-only column indicates the current setting for each object.

Objects can be removed from the check out dialog by using the delete \Join option in the toolbar.

and the check out Opt	ions		<u> </u>
🗙 🖻 🖻	E.		
Name /	Description	Qualifier	Readonly
i Empno	Employee Number		\checkmark
E PSLMST	Personnel	V11WEXLIB	\checkmark
E PSLSKL	Personnel skills	V11WEXLIB	
APSLSYS	Personnel System Main Menu		
		<u>o</u> k	Ca <u>n</u> cel

Use the Cross References **toolbar** option to select associated objects to be checked out at the same time. For example if you review the references for a file you can select to check out all or some of the fields used in the file at the same time.

On the resulting Cross References dialog, the Master only stoolbar option can be used to selected all cross reference objects which exist on the Master Repository but do not exist in the local Visual LANSA Repository.

Vame	Local	Description	Qualifier
E PSLMST	\checkmark	Personnel	V11WEXLIB
🕀 📃 *BLANKS	\checkmark	Blank / blanks variable	
🕀 📃 *ZERO	\checkmark	Zero (0) variable	
🕀 🧐 @@RRNO	\checkmark	Relative record number	
🕀 🧐 @@UPID	\checkmark	Field update / access identifier	
🕀 闷 ADDRESS1	\checkmark	Street No and Name	
🕀 闷 ADDRESS2	\checkmark	Suburb or Town	
🕀 闷 ADDRESS3	\checkmark	State and Country	
🕀 闷 DEPTMENT	\checkmark	Department Code	
🕀 闷 Empno	\checkmark	Employee Number	
🕀 闷 GIVENAME	\checkmark	Employee Given Name(s)	
🕀 🍋 MNTHSAL	\checkmark	Monthly Salary	
🕀 🍋 Phonebus	\checkmark	Business Phone Number	
🕀 闷 Phonehme	\checkmark	Home Phone Number	
🕀 🍋 Postcode	\checkmark	Post / Zip Code	
🕀 🍋 SALARY	\checkmark	Employee Salary	
🕀 闷 SECTION	\checkmark	Section Code	
🕀 闷 SKILCODE	\checkmark	Skill Code	
🕀 🍋 STARTDTE	\checkmark	Start Date (DDMMYY)	
🕀 🍋 STARTDTER	\checkmark	Start date (YYMMDD)	
🕀 🎯 SURNAME	\checkmark	Employee Surname	
🕀 🎯 TERMDATE	\checkmark	Termination Date (DDMMYY)	
🕀 🎯 TERMDATER	\checkmark	Termination Date (YYMMDD)	
🕀 🔲 DEPTAB	\checkmark	Department code table	V11WEXLIB
🕀 🖽 PSLSKL	\checkmark	Personnel skills	V11WEXLIB
🕀 🔲 SECTAB		Section code table	V11WEXLIB

↑ 5.2.1 Check Out

5.2.2 Check In

Check in is used to put the definition of an object that has been created or modified by Visual LANSA into the LANSA for iSeries Master Repository.

You can check in LANSA objects such as processes, functions, files, fields, components or variables. To check in objects, you can use the Repository tab and right click on the object. The check in option will be displayed in the pop-up menu.

As part of the Check In Options for some objects, you can choose various check in and compile related options as appropriate for the selected objects.

When the object is checked in to the Master Repository, the status of the object locks is controlled by the LANSA for iSeries system's task tracking settings and use of the *Keep Locks* option on the check in dialog. (Refer to Using Task Tracking.)

The locking check is enforced so that only the user who currently has the object locked is allowed to check in the object. This prevents two normal users, that is users who are not security officers, from simultaneously checking in the same object to the Master. For example, two developers, Bob and John, are using a shared database on their Visual LANSA installations. They are working on the same partition in the database but using different tasks, *uTask1 and *uTask2. Bob creates a field under *uTask1. John can immediately see and open this field as read-only but cannot check it in until Bob releases the lock on the field by checking it in.

However, this rule only applies to normal users. Security officers can check in any objects and as such should not be used as development profiles. Refer to Using Task Tracking for more details.

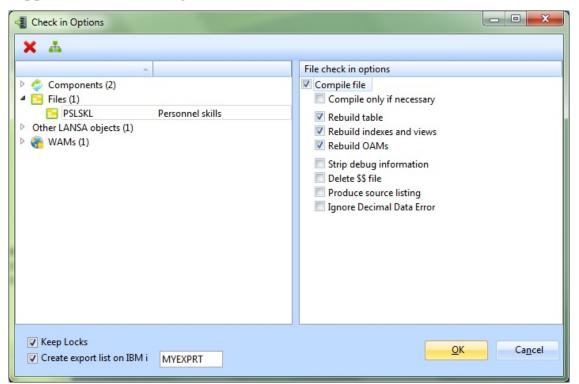
An export list including any objects checked in can be automatically generated during the check in processing.

Also See

Check In Joblog Viewer
5.2.1 Check Out
5.2.4 Start and Stop the Host Monitor
↑ 5.2 Host Monitor

Check In Options

When an object is being checked in, you will be prompted for the actions to be performed when the check in is performed. For example, the following dialog will appear when checking in a File:



Keep Locks

This option, when checking objects in to the LANSA Master Repository on the IBM i, controls whether the objects remain locked to the PC and locked to the Task after check in is complete.

System Tasks (*N) are unaffected by this option. If there is a PC Lock it will remain and if there is a Task Lock it will remain. Use of System Tasks does not affect the current status of the PC Lock or Task Lock. System Tasks override the current lock status and allow the user to do anything whilst not affecting its lock status when used with other types of task.

For non-System Tasks, when unchecked (implies clearing the locks), on Windows, the objects are always unlocked from both the PC and the Task. Thus the objects will need to be checked out for update in order for them to be edited. On IBM i, if the Task is a Normal Task the object is only unlocked from the PC. The Task Lock remains. On IBM i, if the Task is *U or *T, and the system has been configured to allow locks to be released, then the objects are unlocked from both the PC and the Task.

Create Export List on IBM i

This option allows an export list containing the objects selected for check in to be automatically generated on IBM i during the check in processing. By default the export list will be created with the name CKInnnn where nnnn is the next available descending number based on existing export lists e.g. the first export list will be created as CKI9999, on the subsequent check in the next export list will be created as CKI9998.

To create an export list with a specific name or append to an existing export list, nominate the name in the space provided (do not use the prefix DLV or CKI).

If you select an object (or more than one) and press this icon ^E, the Local Cross References dialog will open to show you all the objects used by the selected objects.

References		<u>_ ×</u>
Description	Allow check-in	Qualifier 🔺
	×,	DC@DEMOLIB
	~	
	2	
	3	
Street No and Name	V	
Suburb or Town	×	
State and Country	\checkmark	
Department code table	\checkmark	DC@DEMOLIB
Department Code	\checkmark	
Employee Number	1	
		⊆lose
	Description Personnel Blank / blanks variable Zero (0) variable Relative record number Field update / access identifier Street No and Name Suburb or Town State and Country Department code table	Description Allow check-in Personnel Image: Constraint of the sector of the

From the *Local Cross References* dialog you can select objects for inclusion in the check in by clicking on the 🖶 icon can add them to the Check In Options dialog to check them in as well.

Depending upon the type of object being checked in, you may wish to review the following in the *Technical Reference Guide*:

• File Compile Options

• Process/Function Compile Options.

Check In Joblog Viewer

Any joblogs created on the master for check in jobs can be viewed in the Check In Joblog Viewer. The whole file is downloaded to Visual LANSA and displayed to you one page at a time.

Job name	V6R1M0 080215		<i>i</i>	🎲 🖖 Page	1 /18			
Job name	V6R1MU U8U215							_
			Job Log	LANSA	01 03/08/09	16:13:38	Page 1	
	e	U0000332 QBATCH	User Library	STEWART Number DEVTOOLLIB	er	81	13237	
MSGID	TYPE	SEV DATE	TIME FROM	PGM LIBRARY	INST TO	PGM LIBRARY	INST	
CPF1124	Information	Message	16:13:34.372619 QWTPI : Job 813237/STEW in subsystem QBATCH in C	JART/U0000332 starte	ed on 03/08/0	XT 9 at 8/09 at	*N	
CPI1125	Information	00 03/08/09 Message Cause started us dstributes OSVS2 DEVPGMILE QTEMP LOGOUTPUT DATE(*SVS1 SRTSEQ(*N.)	16:13:34.373053 QUTP Job 813237/STEK Job 813237/STEK Job 813237/STEK Job 813235/STEWART/M JOBETY(5) OUTFTY(5) E QHLPSYS QUSRYS) CL DEVDTALIB IDETESTIIB QGPL) INLASPGRP(=KNONE) (*JOBEND) OUTQ(QGPL_QPRI /AL) SWS(00000000) MSGC *HEX) LANGID(ENU) CNTRY *NO) SPLFACN(*KEEP).	JART-VU0000332 submi JART-VU0000332 submi JON STU_LA. Job 813. JOB) command with "RTITK() RTGDTA(QCM IRLIE(DEVDFTLIB) IN DEVMODLIB DEVSRCI. LOG(4 00 *SECLVL) NT) PRTDEV(PRT01) (QUUSRSYS-STEWART) (tted. tted to job c 237/STEWART/U the following DB) SYSLIBL(C LIBL(DEVCOMI UB DEVTOOLLI LOGCLFCM(*NC NQMSGRPY(*RC CCSID(65535)	0000332 was job SYS IB B DEVLOADLIB) D) HOLD(*NO)	*N	
NONE	Request	03/08/09	16:13:34.374049 QWTSC 	CSBJ ILIB/LANSA) PARM(': 'IDE' 'ENG ' '	LDA ' '	ND QSYS 'STU_LAPTOP'	0195	
CPF9897	Completion	40 03/08/09 From user . From module From procedu Statement . Message . wait	ire DCX DCX DCX A Event: Starting	THPRDOWN 198602 198602AH 9 Package Installat:	ion Processin		*N	
CPCA087	Completion	From user From module From procedu		THPRDOWN DLCEXH c_sendpmFv	*STMT QC	adie. 25YS QSYS	*STMT	

Joblogs can now be saved using the save icon on the toolbar.

Find and *Find Next* is also available. *Find Next* will keep searching until the end of the file and automatically wrap around to the start if nothing is found.

Toolbar buttons are provided for paging up or down, or you can enter the specific page number.

You can also navigate around the spool file using the keyboard.

- *Page Up* Show the previous page
- *Page Down* Show the next page
- *Ctrl+Home* Show the first page
- *Ctrl+End* Show the last page

Arrow up and *down* move the cursor up and down. They will also scroll the spool file one line at a time when at the top or bottom of the display area. This means that you can see the bottom of one page and the top of the next.

5.2.3 Propagation

Repository Synchronization is a feature that allows changes made to a LANSA for iSeries Master Repository to be automatically propagated to Visual LANSA Slave Repositories in order to maintain current object information and relevant system information in the Slave Repositories.

For example, if a Visual LANSA developer checks in a new field to the LANSA for iSeries Repository, the new field would be propagated to other Visual LANSA Slave Repositories in order to match the LANSA for iSeries Master System.

When a field is deleted from the Repository, this change may also need to be propagated. The Repository Synchronization Options in Visual LANSA control how deleted objects are propagated.

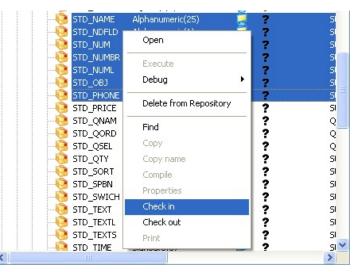
Alternatively, if Repository Synchronization is not in use, Visual LANSA users must manually execute the 5.2.5 Refresh Master Object List to see this new field. They can then check out the field to get the field definition in their local repository if required.

↑ 5.2 Host Monitor

5.2.4 Start and Stop the Host Monitor

There is no need to specifically start Host Monitor unless Repository Synchronization (refer to 5.2.3 Propagation) is required. You can start or stop the Host Monitor using the appropriate button in the LANSA Editor Toolbar.

Note: When Check In or Check Out is requested, the Host Monitor starts automatically.



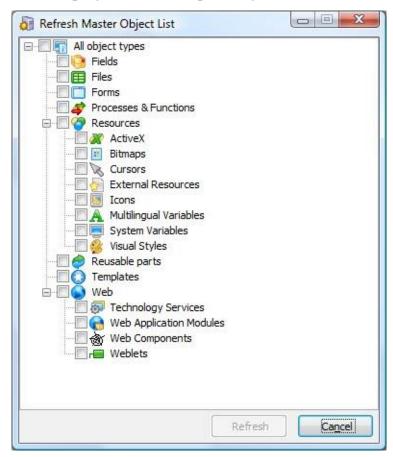
Before you can start the Host Monitor, be sure that:

- The appropriate communications router software has been set up so that the workstation can access the LANSA for iSeries Master system.
- The communications router is active on the workstation.
- The workstation has been defined to LANSA for iSeries.
- ↑ 5.2 Host Monitor

5.2.5 Refresh Master Object List

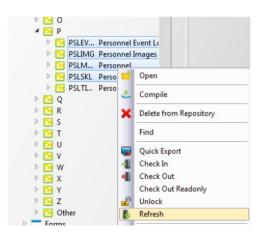
Refresh Master Object List returns a complete set of data from the Master system for the selected object types.

Once the data has been downloaded, the information is merged with the existing Repository data and displayed in the Repository tab.



It is recommended that a complete Refresh be performed initially. This initial refresh may return many thousands of objects that do not currently exist in the local Repository.

Alternately to Refresh the definitions of a group of objects already available in the local repository, select the group of objects and right click to select Refresh.



Object data is downloaded from the Master in batches and then trickled into the local Repository to avoid impacting performance on the local machine. You can use F5 at any time to force the objects to be refreshed immediately.

Job 9	Status	Results	Description	Currently Processing
	ompleted	190 found on master	Refresh templates from master repository	
< Co	ompleted	10 found on master	Refresh Technology Services from master repository	
	ompleted	68 found on master	Refresh web components from master repository	
Co	ompleted	106 found on master	Refresh weblets from master repository	
Co	ompleted	18 found on master	Refresh WAMs from master repository	
Co	ompleted	7783 found on master	Refresh system variables from master repository	
Co	ompleted	2818 found on master	Refresh multilingual variables from master repository	
Co	ompleted	170 found on master	Refresh processes from master repository	
Co	ompleted	6 found on master	Refresh activeX controls from master repository	
Co	ompleted	3 found on master	Refresh cursors from master repository	
Co	ompleted	50 found on master	Refresh visual styles from master repository	
Co	ompleted	273 found on master	Refresh icons from master repository	
Co	ompleted	39 found on master	Refresh bitmaps from master repository	
Co	ompleted	1377 found on master	Refresh reusable parts from master repository	
Co	ompleted	282 found on master	Refresh forms from master repository	
Co	ompleted	322 found on master	Refresh files from master repository	
Co	ompleted	3718 found on master	Refresh fields from master repository	

State columns in the Repository tab allow you to see whether an object is known locally, on the master or both. Further, objects known only to the Master appear as gray to highlight the fact that they are Master only and will therefore only have limited capabilities on a local system.



Master state is persistent. Closing and restarting Visual LANSA will show the same Master and local data.

As this is a snapshot, it is recommended that you regularly run a Refresh to ensure that you have an up to date view of the Master. Alternatively, repository synchronization can be used to ensure that changes made both on the Master and, through Host Monitor, are reflected in local repositories.

Tip: Improve local repository update time by minimizing the number of open objects when performing a Refresh or alternately selective to refresh a set of objects.

↑ 5.2 Host Monitor

5.3 Other Remote System Monitors

Remote System Monitors are used to move LANSA objects and system information from a Visual LANSA System to a Deployment System. Review the following topics:

- 5.3.1 Define a Deployment System
- 5.3.2 Refresh System Information from the Deployment System
- 5.3.3 Initialize the Partition on the Deployment System
- 5.3.4 Deliver To
- 5.3.5 Disconnect from a Remote System Monitor

Also See

Deploying LANSA Applications to a Linux Server in the Deploying Visual LANSA Applications on Linux Guide.

↑ 5. Remote Systems

5.3.1 Define a Deployment System

- 1. Create a host route entry for the Deployment system using the LANSA Communications Administrator. Refer to Add or Change a Host Route in the *LANSA Communications Setup Guide*.
- 2. Define the Deployment System in Visual LANSA by selecting New Remote System. The **Remote System Name** must match the LU Partner name of the Host Route created in Step 1. For information about the New Remote System dialog, go to Create Remote System in the *User Guide*.

New Remote System		×
Remote System Name	LINUX2	Create
Remote System description	Linux System - LINUX2	Cancel
Build environment	Linux	Ca <u>n</u> cel
		Open in editor
Refresh from Remote System		
Partition Initialization		

3. Indicate if the Deployment System is a Linux or IBM i build environment.

If the build environment is **IBM i**, the option to Refresh System Information from the Deployment System will be available and pre-selected.

If the build environment is **Linux**, the option to 5.3.3 Initialize the Partition on the Deployment System will be available and pre-selected.

The Remote System's Partition must be Initialized before any objects can be delivered to the Deployment System. This can be performed when defining the Remote System or later from the context menu associated with the Remote System.

Note - The 5.3.4 Deliver To command will only become available after Refresh System Information from the Deployment System or 5.3.3 Initialize the Partition on the Deployment System has been completed successfully for this Deployment System for the current partition.

↑ 5.3 Other Remote System Monitors

5.3.2 Refresh System Information from the Deployment System

This step is only required for Deployment Systems with IBM i build environment.

LANSA01_1	5C	Rhode	MI TO
Templates	ø	Open	-
Users Tasks	×	Delete from Repository	
È.		Find	
		Refresh	
		Copy Name	
	Q	Properties	

Before you can deliver any objects to an IBM i deployment system, you must refresh the workstation names from the Deployment System. Ensure the following before you initialize the partition, or attempt to use 5.3.4 Deliver To:

- The Remote Deployment System is installed and at the same LANSA version and EPC level as the Visual LANSA on Windows installation.
- The corresponding partition is defined on the deployment system and matches the local partition's definition.
- TCP/IP is set up and active so that the workstation can access the remote system.
- You have a login to connect to the remote system and use the target LANSA installation.

Note - The partition RDML/RDMLX setting, Job CCSID and default language **MUST** match or the Refresh from the Remote System will fail. Other partition settings, such as available field types, should match, otherwise unpredictable results may occur when using the Deliver To command.

An IBM i Refresh can be performed when creating the Remote System definition or by right clicking on the Remote System in the Repository tab and choosing *Refresh*. If the User and password differs from the current user in Visual LANSA you must provide the user and password to connect to the remote system.

Refresh definit	tions from LANSA04_V12
User Password	
	ws credentials
	Refresh Cancel

5.3.3 Initialize the Partition on the Deployment System

This step is only required for Deployment Systems with Linux build environment.

Remote Syst LANSA05	
	c Dpen
± 💽 Templates € 🔏 Users 🖇	Celete from Repository
∃ 🛃 Tasks	Find
6	Partition Initialization
	Copy Name
(Reporties
And a state of the second second	and the second

Before you can deliver any objects to a Linux deployment system, you must initialize the partition on the Deployment System. Ensure the following before you initialize the partition, or attempt to use 5.3.4 Deliver To:

- The Remote Deployment System is installed and at the same LANSA version and EPC level as the Visual LANSA on Windows installation.
- TCP/IP is set up and active so that the workstation can access the remote system.
- You have a login to connect to the remote system and use the target LANSA installation.

Right click on the Remote System in the Repository tab and choose Partition Initialization, then select the appropriate checkboxes and press Okay. If you are deploying web applications ensure the partition is enabled for the web.

Partition Initialization on RHESUS
Mandatory Partition Initialization
Enable for the Web
OK Cancel

When starting a connection to a remote system, the Visual LANSA Editor attempts to use the same login used for the Visual LANSA session. If this does

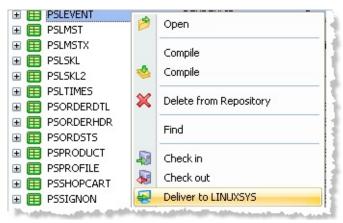
not connect successfully, a connection dialog will prompt for the connection details. This will occur the first time Deliver To is used for a Remote System and **every time** a joblog is reviewed on a remote system. The user be defaulted to the last user to connect to a remote system on the connection dialog.

G Connect to Re	mote System RHESUS		
User Name	user 1		
Password			
Use Windows cr	edentials		
Execution Override	Options		
Messages			
		Connect	Cancel

↑ 5.3 Other Remote System Monitors

5.3.4 Deliver To

Deliver To is used to copy an object that has been created or modified on a Visual LANSA on Windows installation to a Deployment System.



You can Deliver LANSA objects such as processes, functions, files, fields, reusable parts and variables. When the Deployment System is a Linux build environment you can also deliver message files.

To Deliver LANSA objects, you can use the Repository tab and right click on the object(s). The Deliver To Options will be displayed in the pop-up menu. Editor Lists can also be selected to deliver the object(s) associated with the list definition.

Also see

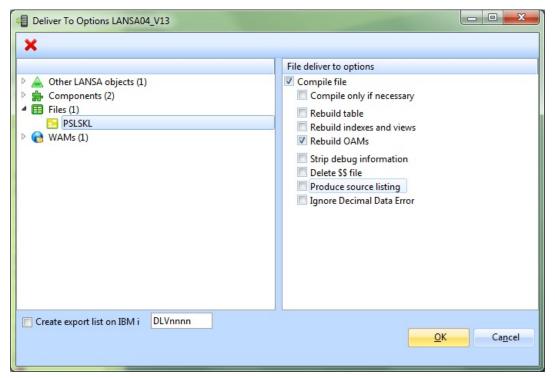
Deliver to Joblog Viewer

↑ 5.3 Other Remote System Monitors

Deliver To Options

When an object is being delivered, you will be prompted for the actions to be performed when the deliver to is performed.

When delivering to IBM i, these options correspond with the Check In options. For example, the following dialog will appear when delivering in a File to an IBM i:



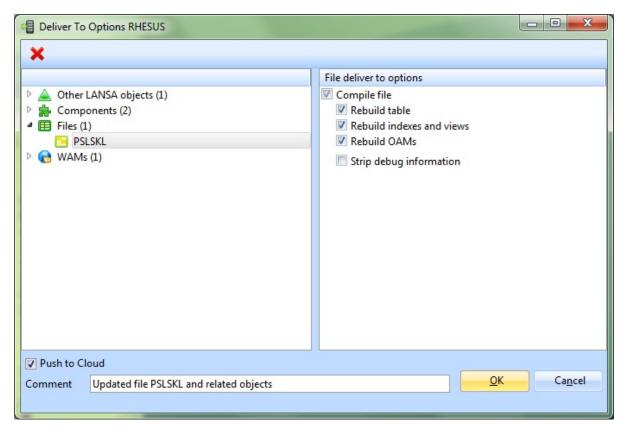
Create Export List on IBM i

When delivering to an IBM i a checkbox is available to indicate if an export list of the delivered objects should be automatically created on IBM i during the deliver to processing.

By default the export list will be created with the name DLVnnnn where nnnn is the next available descending number based on existing export lists e.g. the first export list will be created as DLV9999, on the subsequent check in the next export list will be created as DLV9998.

To create an export list with a specific name or append to an existing export list, nominate the name in the space provided (do not use the prefix DLV or CKI).

The options available when delivering to Linux are different. For example, the following dialog will appear when delivering in a File to a Linux system:



Push to Cloud

When delivering to a Linux system a checkbox is available to indicate if the Linux environment should be pushed to the cloud. A supporting comment should be provided when Push to Cloud is selected.

To Deliver Non-LANSA objects, create a file named deliverto.txt in the LANSA System directory. Open the file and include each object with a fully qualified path. Add only one object per line in the file. For example:

C:\LANSA\LANSA_plugin\WebServer\Images\style\myapp.css C:\LANSA\LANSA_plugin\WebServer\Images\banner.gif

A Remote System Monitor automatically starts when a Deliver To request is made.

Each time you deliver objects, either to IBM i or Linux, any entries in deliverto.txt will be processed and included in the delivery. Objects are delivered to the partition directory on the target system, for example, /LANSA_devpgmlib/x_lansa/x_dem

When connecting to the Remote System on IBM i you must supply a Task ID

that is valid on the IBM i system. This task is assigned to the object on the IBM i. It overwrites the current assignment on IBM i. It does not use the Task that the object is assigned to on the PC. If the Task is a special task which has *Unlock on Check In* set to **Yes**, the Task Id will be cleared on IBM i.

Before you can use Deliver To, be sure that you have followed the steps that are relevant to the target remote system. Refer to:

- 5.3.1 Define a Deployment System,
- 5.3.2 Refresh System Information from the Deployment System and
- 5.3.3 Initialize the Partition on the Deployment System.
- ↑ 5.3.4 Deliver To

Deliver to Joblog Viewer

Joblogs created on the deployment system can be viewed in the Deliver to Joblog Viewer.

When delivering to IBM i this is the same as the Check In Joblog Viewer.

When delivering to Linux a section of joblogs are available and the whole file is downloaded to Visual LANSA and displayed.

6.9	Joblog file dpinstal 👻 🖪 🗃
op	:/lansa.rg132/x_lansa/x_hmrqst/atealone/c0001194/dpinstal.log
I	stallation of Package - C0001194 - started 2013-11-28 05:09:53.
I	mport of LANSA internal data started 2013-11-28 05:09:53.
> :	Export Version 003 found. (05:09:53) 2013-11-28 05:09:54.
> :	
	(1110) - WARNING: Skipping import of file LXXF29 as it is only required for Development systems. (05:09:53) 2013-11-28 05:09:54.
	(1110) - WARNING: Skipping import of file LXXF74 as it is only required for Development systems. (05:09:53) 2013-11-28 05:09:54.
	Jse File's Library override if included. (05:09:53) 2013-11-28 05:09:54.
	coading format information from file /opt/lansa.rg132/x_lansa/x_hmrqst/atealone/c0001194/internal/lxxbdf.asf. (05:09:53) 2013-11-28
	coading format information from file /opt/lansa.rg132/x_lansa/x_hmrqst/atealone/c0001194/internal/lxxf60.asf. (05:09:53) 2013-11-28
	coading format information from file /opt/lansa.rg132/x_lansa/x_hmrqst/atealone/c0001194/internal/lxxldf.asf. (05:09:53) 2013-11-28
	coading format information from file /opt/lansa.rg132/x_lansa/x_hmrqst/atealone/c0001194/internal/lxxf20.asf. (05:09:53) 2013-11-28
	coading format information from file /opt/lansa.rg132/x_lansa/x_hmrqst/atealone/c0001194/internal/lxxf3.asf. (05:09:53) 2013-11-28
	coding format information from file /opt/lansa.rg132/x_lansa/x_hmrqst/atealone/c0001194/internal/lxxf66.asf. (05:09:53) 2013-11-28
	Loading format information from file /opt/lansa.rg132/x_lansa/x_hmrqst/atealone/c0001194/internal/lxxf67.asf. (05:09:53) 2013-11-28
	coding format information from file /opt/lansa.rg132/x_lansa/x_hmrqst/atealone/c0001194/internal/lxxfan.asf. (05:09:53) 2013-11-28
	Loading format information from file /opt/lansa.rg132/x_lansa/x_hmrqst/atealone/c0001194/internal/lxxf02.asf. (05:09:53) 2013-11-28
	.oading format information from file /opt/lansa.rg132/x_lansa/x_hmrqst/atealone/c0001194/internal/lxxfcr.asf. (05:09:53) 2013-11-28 Intermediate import file name is /opt/lansa.rg132/x lansa/tmp/exp9d99d.tmp. (05:09:53) 2013-11-28 05:09:54.
	intermediate import file name is /opt/lansa.tg1s2/X_lansa/tmp/expassd.tmp/ (05:09:53) 2013-11-20 05:09:54. Jonverting data from file LXKBDF for import to the LANSA environment. (05:09:53) 2013-11-28 05:09:54.
	converting data from file LARDUF for import to the LARDA environment. (US:US:S3) 2013-11-20 US:US:S4.
) pectras converted from file LAXF60 for import to the LANSA environment. (05:09:53) 2013-11-28 05:09:54.
	benverving dates from file LAXF60 for Import to that Area environment. (05.05.53) 2015-11-20 05.05.54.
	Deverting converting file LXLDF for import to the LANSA environment. (05:09:53) 2013-11-28 05:09:54.
	inconverting date from file LXXLDF (05:09:53) 2013-11-28 05:09:54.

Joblogs can now be saved using the Save or Save All icons on the toolbar.

↑ 5.3.4 Deliver To

5.3.5 Disconnect from a Remote System Monitor

Select the required *Remote System* in the *Repository* tab and open the context menu by right clicking on the mouse. Select *Disconnect* to shut down the *Remote System Monitor*—provided the Remote System Monitor is active. If it is not, the Disconnect option is not available.

This is not normally required as the Remote System Monitor is automatically shut down when the LANSA Editor is closed.

ø	Open
×	Delete from Repository
	Find
6	Disconnect
6	Refresh
	Copy Name
Q	Properties

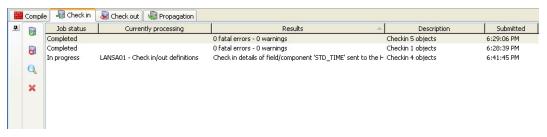
1 5.3 Other Remote System Monitors

5.4 Job Status and Messages

Each 5.2.2 Check In, 5.2.1 Check Out, 5.2.3 Propagation or 5.3.4 Deliver To request is handled as a job. As the job is processed, the associated tab view will update with the latest messages available. When complete a Completed job status is shown to indicate that the requested job has completed.

A job is deemed complete when every activity associated with the job is complete. For example, if a job contains a check in of a file and subsequent compile on the master system, it will remain in progress until the file compile is complete.

For most jobs, the final result is all that is important. Assuming the job is successful, there is no need to see the messages. If any errors are detected, the job item will be shown in red, and a double click on the job will show the job detailer listing all messages for the job. Refer to 5.4.1 Detailed Messages for further information.



Further Information

- **5.4.1 Detailed Messages**
- 1 5. Remote Systems

5.4.1 Detailed Messages

On occasion, a request to a Remote System Monitor will encounter an error. For example, checking in a file for compile will result in a failure if the file is locked for use on the Master. At any time during the job processing, the user can review the state of the job and see all of the messages returned by the Remote System Monitor.

Each job is split into three main areas (if appropriate): Check in/out definitions, RDML Compiles and RDMLX Compiles. The messages are shown in the appropriate area. As each part of the job is completed, the area of the job will show as complete. Once all areas are complete, the job is deemed complete.

	Date	Time	Message	
V LANSA01		Completed		
🖃 🧹 Check in/out definitio	ins	Completed		
7074	2005-03-02	09:57:53	Check in details of File 'IDETESTLIB/PSLSKL' sent to the Host Repository.	
7075	2005-03-02	09:58:24	FILE definition PSLSKL from IDETESTLIB now locked by this job	
7078	2005-03-02	09:58:24	FILE definition PSLSKL from IDETESTLIB has now been unlocked	
7079	2005-03-02	09:58:24	Object IDETESTLIB/PSLSKL type FILE check in to iSeries host repository completed.	
7080	2005-03-02	09:58:24	Check in/out transactions for host monitor job 000000244 have completed.	
🖃 🧹 RDML Compiles		Completed		
7072	2005-03-02	09:57:53	Creation of Package - D0000244 - Started	
7073	2005-03-02	09:57:53	Copy of file pslskl.ctd was successful	
7076	2005-03-02	09:57:53	Creation of Package - D0000244 - ended successfully	
7077	2005-03-02	09:57:53	Copying Package to /LANSA_devpgmlib/x_lansa/x_hmrqst	
7082	2005-03-02	09:58:00	Request to install Application Package 'STUPCD0000244' sent to the Host Repository.	
7083	2005-03-02	09:58:00	Request to compile File 'IDETESTLIB/PSLSKL' sent to the Host Repository.	
098	2005-03-02	09:58:39	Unit of work U0000244/STUPC has started processing object STUPC/D0000244.	
7100	2005-03-02		Installation of Package - D0000244 - Started	
7101	2005-03-02		Copy of file pslskl.ctd was successful	
Q 7102	2005-03-02		Installation of Package - D0000244 - ended successfully	
7103	2005-03-02		Unit of work U0000244/STUPC has started processing object PSLSKL/IDETESTLIB.	
Q 7104	2005-03-02		Compile transactions for host monitor job 000000244 have completed	
Q 7117	2005-03-02		FILE definition PSLSKL from IDETESTLIB now locked by this job	
Q 7118	2005-03-02		FILE definition PSLSKL from IDETESTLIB has now been unlocked	
🔍 7119	2005-03-02	09:59:39	Create or re-create of file PSLSKL from IDETESTLIB completed normally	
🖃 🧹 RDMLX Compiles		Completed		
7084	2005-03-02	09:58:01	Creation of Package - C0000244 - Started	

All compiles on a Remote System will produce a joblog. LANSA now allows the user to see the joblog within the Visual LANSA Editor. Where a message relates specifically to processing done on the Remote System, the entry is shown with an icon to indicate that a joblog is available.

Following is a sample joblog from a request sent to a LANSA for iSeries Master system:

A			
Job nam	V5R2MO 020719	Job Log LANSA01 02/03/05 09:59:40 Pa U0000244 User STEVART Number	ge 1 29
ISGID SFT1124	TYPE Information	DEFICIENT LIDIARY	INST *N
CPI1125	Information	<pre>00 12/03/05 09:58:39.185336 OWTPCRJA OSYS 0108 *EXT Message: Job 134629/STEWART/U0000244 submitted. Cause Job 134629/STEWART/U0000244 submitted to job queue QBATCH in QOPL from job 134629/STEWART/WON STUPC. Job 134629/STEWART/U0000244 was started using the Submit Job (SBMJOB) command with the following job attributes: JOBPTY(5) OUTFTY(5) PRTTXT() RTGDTA(QCMDB) SYSLIBL(OSYS GSYS2 (D4FYS) OUSKYS) CURLID(DEVICIDIB) INLIEBL(DEVCOMLIB DEVPCMLIB DEVDTALLB IDETESTLIB DEVMODILB DEVSRCLIB DEVTOOLLB DEVICADLIB QTEMP QGPL ASPSYDER(M)LOG(4 00 *SECUV1) LOCCPG(*NO) INOMSGRPY(*ROD) OUTQ(QGPL/QFRINT) PRTDEV(PRT01) HOL(*NO) DATE(*SYSLL) SUS(0000000) NSQ(QUSERSYS/STEWART) CCSLD(5553) SKTSEQ(*M**HEX) LANGID(ENU) CHTRYID(US) JOBMSGOMX(64) JOBMSGOFL(*PFTWRAP) ALWALTTHD(*NO)</pre>	*N
NONE	Request	122/03/05 09:56:39 188272 OUTSCENI HARM(*1DA ***********************************	0178
CPF9897	Completion	40 02/03/05 09:58:45:197520 DCXS06XX DEVLCOLIB *STMT *EXT From module DCXP8602 From procedure DCXP8602AH Statement 4 Message Event: Starting Package Installation Processing. Please vait	*N
CPF9897	Completion	Cause No additional online help information is available. 40 02/30/50 99:58:45.308312 DCX586KX DEVLCOLIB *STMT *EXT From module	*N

Following is a sample joblog from a request sent to a Linux Deployment system:

QPJOBLOG - 005884/CASSIEH/C0000254
Table PSLEVENT created/altered. Attempting to create Index PSLEVENT_PR. Index PSLEVENT_PR created. Attempting to create Index PSLEVENTB_IR. Index PSLEVENTB IR created.
Attempting to create Index PSLEVENTA_IR. Index PSLEVENTA_IR created. Attempting to create Index PSLEVENTB_I. Index PSLEVENTB_I created.
Attempting to create Index PSLEVENTA_I. Index PSLEVENTA_I created. Attempting to create Index PSLEVENTEMPEVTDOC_I. Index PSLEVENTEMPEVTDOC_I created. Attempting to create Index PSLEVENT_R.
Index PSLEVENT_R created. RUNSQL completed normally. Return code is 0. (S0022) Compling OAM for pslevent gcc -fFIC -w -march-pentium4 -c -IDX_OPERATING_SYSTEM_LNX -DX_OPERATING_SYSTEM_UNIX -DX_32BIT_COM
-DX_DATABASE_ODBC -DX_DBMS_COL_ALIAS -I/opt/t/lansa2/x_lansa/source /opt/t/lansa2/x_lansa/x_dex/sour ld -shared -E -o /opt/t/lansa2/x_lansa/x_dex/bin/libpslevent.so -L. pslevent.o -L/opt/t/lansa2/x_lan

1 5.4 Job Status and Messages

6. Change Management

A change management strategy needs to address the fundamental questions of:

- What object can be changed?
- Who can make a change to this object?
- When can the object be changed (sequence)?
- Where can the object be changed (environment)?

To understand how LANSA can assist in your designing and implementing your change management strategy, review the following topics:

6.1 Change Management Concepts6.2 Using Task Tracking in LANSA6.3 Repository Synchronization

Also See

Task Maintenance System Information

6.1 Change Management Concepts

6.1.1 Object	6.1.2 Task	6.1.3 Impact	6.1.4	6.1.5 Third
Locking	Tracking	Analysis	Deployment	Party
Concepts	Concepts	Concepts	Concepts	Packages

There are many different strategies for controlling software development and maintenance projects. The strategies often depend upon factors such as project size, project complexity, required development/test/production environments, deployment and other issues. For instance, 3 developers maintaining an application on a single system will have different requirements from an international company with 30 developers using 10 systems in 3 different countries.

Typically, the following types of topics are included in a discussion of change management:

- impact analysis for changes
- locking and releasing objects
- source comparison
- identification and grouping of changes
- tracking object history and movements of objects
- synchronization to prevent lost or overwritten changes
- testing changes before migration
- migration of changes between environments (development, testing, production)
- distribution of software changes

A simple change management scenario might be as follows:

- a program enhancement is requested to a completed program in the production environment
- developers determine the impact of enhancement request
- all objects impacted are identified and locked in development environment
- a developer modifies the objects and tests the modifications in development
- any modified objects are moved to the test environment for review and approval
- after approval the modified objects are moved into production environment
- objects are unlocked in development

As more products, types of objects, machines, interfaces and operating systems are introduced, the challenges of change management increase.

There are many 6.1.5 Third Party Packages that can be integrated with the LANSA environment to provide comprehensive change management solutions for the complete LANSA family of products.

LANSA provides an easy to use suite of basic change management features that are designed to be used as part of company's overall change management strategy and procedures.

1 6. Change Management

6.1.1 Object Locking Concepts

There are 3 levels of Object Locking implemented by the Visual LANSA development environment:

- **System-wide locks** across all installations of LANSA that are connected together. For example, the IBM i Master and all the Slave systems that are connected to it.
- **Database-wide locks** across all installations of LANSA logged in to a single instance of a database.
- **Workstation-wide** locks on a single installation of LANSA on a single database. These locks do not effect developers who share the same database. They control a specific developer's use of a specific database.

Note: For indirect methods of modifying this behaviour in LANSA, refer to Special Authorities and Task Tracking.

Also note that the use of Specialised BIFs and the ROM interface count as another instance of Visual LANSA.

All these locks are automatically applied when the developer performs a specific action. For example, the developer attempts to edit a Form on a Slave system. Under the covers, Visual LANSA will obtain a System-wide lock on the Form to ensure that no other workstation has it locked and that it is locked to the current Task Id. A Database-wide Lock will also be obtained to ensure that only one developer on the current database is editing it at this moment.

The following is what occurs when it's a Slave system. There is a section at the end of this topic on differences with Independent Systems.

System-wide Locks

A System-wide Lock consists of a Task ID and a Workstation. Users may share the right to use a Task Id, but the Workstation must be unique. When an object is created or is checked out for update from the Master, the Task Id and Workstation are assigned to the object. Refer to 6.1.2 Task Tracking Concepts and Unlock Objects in Task Tracking for more information on Task Id.

The Workstation is the principal means of ensuring that a single developer anywhere in the System is the only one who may modify an object. The Workstation is assigned when creating the object or when checking it out for update. The Workstation is removed when checking the object into the Master (Keep Locks is unchecked) or when specifically unlocking the object either in Visual LANSA or on the IBM i Master under Work With Tasks. The use of System-wide Locks cannot be changed. There are no direct options in Visual LANSA to alter this behaviour.

Database-wide Locks

Database-wide Locks ensure the single update of the whole or some part of a particular database. The Workstation is also used to control this type of lock. Examples of Database-wide Locks are:

- Nobody else may use the database during System Initialisation
- Nobody else may use the Partition during Partition Initialisation
- Only one developer in one instance of Visual LANSA may update an object at a time. Even the same developer running a second instance of Visual LANSA will be locked out.

Database-wide Locks can be switched off. Refer to Object Locking for details.

Workstation-wide Locks

Workstation-wide Locks help a developer by ensuring that they do not hinder themselves. The lock is specific to a Workstation plus the Windows Process Identifier. For example:

• If two instances of Visual LANSA are being used by the developer, only one compile will be allowed at a time

Workstation-wide Locks can be switched off. Refer to Object Locking for details.

Independent System Differences

The System-wide Lock is not based on the Workstation. It's only the Task Id.

A single user, independent, workstation may choose to switch off Object locking, but be aware that the protections offered by Database-wide Locks and Workstation-wide Locks listed above will not be available. The default for such as system is to set Object Locking to Off.

Also See

Object Locking Workstation Locks ↑ 6.1 Change Management Concepts

6.1.2 Task Tracking Concepts

The fundamental objective of change management is to be able to identify and control how changes are made as applications are developed or maintained. The concept of task tracking is to create a Task Identifier that is used to control and record changes to individual objects in the LANSA development environment. Once a developer logs on with an assigned Task ID, the Task ID can be used to check authority to access objects, lock objects, log activity on objects, and so on. Specific Task Identifier Rules are defined to implement site standards.

Change management strategies might use task tracking in one of the following ways:

- To define units of work. A Task ID can be created to lock a group of objects for a specific application change.
- To a control changes to a product.
- To monitor or control the work done by individual developers. In this case each developer may be assigned their own Task ID to create a record of the work performed.

LANSA's task tracking is very flexible. For example, it allows you to define how object locking should be performed when task tracking is used. Refer to Unlock Objects in Task Tracking.

Task tracking is an optional feature. The LANSA development environment provides a variety of settings to control how Task IDs are used. For more details, refer to 6.2.2 Approaches for using Task Tracking.

Also See

- 6.1.1 Object Locking Concepts
- ↑ 6.1 Change Management Concepts

6.1.3 Impact Analysis Concepts

Impact analysis is a concept that helps developers evaluate how a change to an object might affect other objects or an application. For example, if the length of the EMPNO field were increased from 7 characters to 10 characters, what objects would be affected by this change? What files must be recompiled? What application programs would have to be reviewed? How do you find a list of all impacted objects?

Impact analysis tools are an important part of an overall change management strategy. Impact analysis is often the starting point to identify all objects that are impacted by a change. The resulting list of impacted objects can be used to create a unit of work that can be controlled using 6.1.2 Task Tracking Concepts.

Visual LANSA supports impact analysis in a number of ways:

- cross reference generation
- find facilities
- lists (static or dynamic).

With Visual LANSA's impact analysis features, you can search for LANSA objects that have particular features and then find out what relationship those objects have to other LANSA objects. A list of the objects found can be built up and held in an impact list. Refer to Repository Find in the *User Guide*.

Also See

- 6.1.4 Deployment Concepts
- ↑ 6.1 Change Management Concepts

6.1.4 Deployment Concepts

Once an application has been completed in a development environment, it must be migrated or deployed to a production environment for use by the end-users of the application. Change management strategies use many different approaches for application deployment and these strategies may involve moving objects to test environments before moving to production environment. The common element of these strategies is the need to move an identified group of objects from one place to another. The group of objects may be identified as described by the 6.1.3 Impact Analysis Concepts.

LANSA provides two techniques for moving objects:

Export/Import

The export and import facilities in LANSA are provided to enable the LANSA objects to be 'exported' from one LANSA partition and then 'imported' into another LANSA partition. Developers create export lists that may also contain objects that are not part of the LANSA system. (These lists can be based on Task IDs as described in 6.1.2 Task Tracking Concepts.) From an export list, a set of export files are generated. These files can be moved to the target machine and imported into the LANSA System.

Deployment Tool

Using the Visual LANSA Deployment Tool, you create a package or series of related packages containing the relevant objects for an application. In addition to the LANSA objects the package details options and settings that relate to the environment in which the package will be deployed. Once you have completed the selection process the package is built and prepared for deployment.

Deliver To

In Visual LANSA you can create a Remote Deployment System, and after appropriately configuring this system, you can deliver LANSA and Non-LANSA objects to this system using the Deliver To command. Single objects, a selection of objects or all objects associated with a task or editor list can be delivered to the remote deployment system.

This utility is intended to assist move objects between a development and test environment on different platforms. It should not be used to move objects directly into a production.

↑ 6.1 Change Management Concepts

6.1.5 Third Party Packages

A number of Change Management Software companies have created packages specifically for LANSA or have written interfaces to their software packages to support the LANSA software. These Third Party Vendors specialize in Change Management software and can provide a very comprehensive set of features. Most packages support development on Windows or IBM i using a variety of different configurations.

As you develop your change management strategy, it may be beneficial to contact some of these vendors for a description to their software packages.

To review a list of current software packages provided by Third Party Software Vendors, please check the LANSA Web site at www.lansa.com.

↑ 6.1 Change Management Concepts

6.2 Using Task Tracking in LANSA

To understand LANSA's Task Tracking, refer to the following topics as appropriate:

- 1. If you have not used task tracking, please refer to 6.1.2 Task Tracking Concepts and 6.2.1 Introduction to Task Tracking .
- 2. Review the 6.2.2 Approaches for using Task Tracking. This section will help you in Choose Your Task Tracking Approach.
- 3. Once you have selected your approach, you must perform the 6.2.3 Task Tracking Set up.
- 4. Review 6.2.4 What Happens When Task Tracking Is Active? and 6.2.5 Typical Development Cycle Example.
- 5. Finally, review 6.3 Repository Synchronization.
- 6. If you need to deploy objects, refer to 6.2.6 Task Tracking and Import/Export.

Remember, Task Tracking is just one part of a complete change management strategy.

1 6. Change Management

6.2.1 Introduction to Task Tracking

Task Tracking can be used in order to manage and control development within LANSA. Task identifiers can be allocated to individual tasks, developers, or products (groups of objects). Tasks are defined at system level but lock objects and records changes at the partition level. (Refer to Task Identifier Rules.) Task Tracking will allow you to:

- Keep strict control of development within LANSA by controlled allocation of Task identifiers to individual development staff.
- Enforce locks on objects during development by allocating a specific Task identifier against an object.
- Review an audit/history log of all work performed on objects within the task.
- Create an export list of all objects changed for a particular maintenance item or enhancement using a specific Task identifier.

If Task Tracking is enabled for the selected partition, a developer must select a Task ID when logging on to Visual LANSA. (Refer to Logon Parameters.) Once logged on, a developer is only allowed to modify objects that have already been locked to the specified Task ID and PC Name, or objects that have not yet been assigned to a Task. The current Task ID is displayed on all the relevant LANSA windows. The developer may switch to a different Task ID if required. (Refer to Change Current Task.)

Generally, an object that is locked out by one task cannot be used by another task. (Refer to Tracked Objects.)

Task tracking can, and should be, tailored to your site's specific needs. You can decide how much control you want to have over the development effort, how much administration overhead you are willing to bear, how big is your team and whether development needs to be controlled by individual developer, by product or by both. For further information, refer to 6.2.4 What Happens When Task Tracking Is Active?

If you are using a Visual LANSA Slave System with a LANSA for iSeries Master System, you will find a description of Task Tracking and how to set up your Task Identifiers in Task Tracking in *The Housekeeping Components of the LANSA for iSeries User Guide*.

Also See

Tracked Objects

Task Identifier Rules6.1.2 Task Tracking Concepts↑ 6.2 Using Task Tracking in LANSA

Tracked Objects

Task Tracking will record all development work performed in LANSA at partition level on the following LANSA objects:

- Fields
- Visual LANSA components (includes WAMs)
- Files
- Functions
- Processes
- Application templates
- System variables
- Multilingual variables
- Weblets

Whenever the term "object" is used in relation to task tracking, it is intended to describe one or all of these objects.

Also See

Task Identifier Rules ↑ 6.2.1 Introduction to Task Tracking

Task Identifier Rules

The following rules apply to Task Identifiers:

- Tasks IDs can only be defined in a LANSA for iSeries Master LANSA System or an independent Visual LANSA system. Task IDs cannot be defined or modified in a Visual LANSA Slave system.
- Task IDs exist at the LANSA System level, i.e. they can be used in one or more partitions in LANSA.
- Task IDs can be assigned different statuses such as Open, Working, Closed, and Finished. (Refer to Task Status.) These statuses determine how a Task ID can be used.
- A Task ID requires that a developer's user profile must be authorized to use it. One or more developers may be assigned to a Task ID. (Refer to Share Task IDs.) Authorities are based on individual Task ID settings. (Refer to Set Special Task ID.)
- A LANSA object can be locked by a single Task ID at any given time. The duration of a lock is defined by the Task ID status. (Refer to 6.2.3 Task Tracking Set up.)

LANSA provides a very flexible system for defining Task IDs and authorities. You must understand the task tracking settings in order to Task IDs effectively.

Also See

Tracked Objects

1 6.2.1 Introduction to Task Tracking

6.2.2 Approaches for using Task Tracking

Task tracking enforces locks when changing or deleting objects. An object which is locked to one task cannot be used by another task. Task Tracking is used to record all work performed on objects for a particular task. The task identifier can also be used to export all objects worked on for a task. Task tracking is very important for controlling a LANSA for iSeries Master LANSA System used by multiple Visual LANSA Slave Systems. When Visual LANSA Slave Systems are used, task tracking also locks an object to a PC.

When setting up task tracking you need to consider the following types of questions:

- How much control do you want to maintain over the development effort?
- How many developers are on your team?
- Does development need to be controlled by individual developer? Or by product? Or by both?

Your task tracking approach is also determined by your system type. You should review:

- Task Tracking in Master/Slave Systems
- Task Tracking on Independent Systems

Based on your site requirements, you can select from several different approaches:

- Full Task-Oriented Tracking
- Tracking by Product
- Tracking By Developer
- Minimum Tracking
- Combined Approaches

To determine the best solution for your site, refer to Choose Your Task Tracking Approach.

Task tracking enforces locks when changing or deleting objects. An object which is locked to one task cannot be used by another task. Task Tracking is used to record all work performed on objects for a particular task. The task identifier can also be used to export all objects worked on for a task. Task tracking is very important for controlling a LANSA for iSeries Master System used by multiple Visual LANSA Slave Systems.

Also See

Set Special Task ID6.1 Change Management Concepts1 6.2 Using Task Tracking in LANSA

Choose Your Task Tracking Approach

When you set up task tracking, you need to choose the approach that best suits your business needs. (Refer to 6.2.2 Approaches for using Task Tracking.) In many cases, full task-oriented tracking is not required. For a LANSA for iSeries Master System used by multiple Visual LANSA Slave Systems, you can choose from these four basic task tracking approaches:

Full Task- Oriented Tracking	Developers are assigned a new task identifier for each new unit of work they perform on any product. Tasks could involve multiple developers and multiple products.
	Suitable for large sites, for sites with task-oriented methodologies and sites with product and developer crossover.
	+ Development is strictly controlled.
	+ Small incremental changes can be controlled and migrated.
	+ Task IDs can be used to migrate changes.
	- Administrative overhead.
Tracking	Task identifiers are assigned to products.
by Product	 Suitable for small teams with little crossover between products. + Little administrative overhead. Open one task per product. + Task IDs can be used to migrate the complete set of product
	changes.
	- No product completion points per developer.
Tracking	Task identifiers are assigned to individual developers.
By Developer	Suitable for small teams where there is little crossover between developers.
	+ Little administrative overhead. Open one task per developer.
	+ Task IDs can be used to migrate changes.
	- No product completion points. Tasks can stay open indefinitely.
Minimum	One task for the entire system.
Tracking	Suitable for small teams with trusted developers and minimal modifications.
	+ Minimal administrative overhead.

- There is no locking of objects.
- You cannot export by task.
- There is no task history.
- No definable units of work

By using the LANSA's ability to transfer objects between tasks, it is also possible to combine some (but not all) of these approaches. Refer to Combined Approaches.

Task tracking is very flexible. You are not limited to these four techniques. You may design your own approach for using task tracking as part of your change management strategy, or you might use 6.1.5 Third Party Packages that work with LANSA's task tracking features. When designing a custom approach, you may also want to refer to Set Special Task ID, Share Task IDs, Unlock Objects in Task Tracking, Transfer Object Locks, Special Authorities and Task Tracking and Task Tracking Recommendations.

↑ 6.2.2 Approaches for using Task Tracking

Full Task-Oriented Tracking

Full task-oriented task tracking assigns a new task identifier for each new unit of work to be performed on any product to ensure that development work is carefully controlled. Small incremental changes can be controlled and migrated. A unit of work can be a new product feature or a product fix. When a task is completed, the task identifier is closed and removed from use. This style of task tracking is suited for large sites, for sites with task-oriented methodologies, and sites with product and developer crossover. It is recommended for large, professional development teams. It should be noted that it involves a relatively high level of planning and administration.

This approach uses normal Task IDs, i.e. Special Task IDs are not required. Developers are assigned a new task identifier for each new unit of work they perform on any product. Tasks could involve multiple developers and/or multiple products.

A typical development cycle might be as follows:

- A new task is created. Its Task Status is automatically set to OPN.
- As the required objects are locked to this task, the task status is automatically set to WRK.
- Once the work is completed, the task status is manually set to CLS. The locked objects can be exported to a test environment and finally a production environment.
- Once in production, a task status can be manually set to FIN and the objects are unlocked.

Also See

Combined Approaches Share Task IDs ↑ 6.2.2 Approaches for using Task Tracking

Tracking by Product

Product-oriented task tracking assigns unique task identifiers to products (groups of objects). This approach is suitable for small teams which have little crossover between products. In product-oriented tracking, all developers share the same task identifier when working on a particular product. Task IDs can be used to migrate the complete set of product changes.

Product-oriented task tracking requires less administration than Full Task-Oriented Tracking.

To use Task Tracking by Product you must follow the steps as described in Set Special Task ID.

For example, Task IDs such as *T00000A and *T00000B could be created for Product A and Product B, respectively. These special Task IDs will allow multiple developers to be authorized to the task. This approach assumes that there are no shared objects between Product A and Product B during the time that development work is being performed with these two tasks.

Using this scenario you should use the Keep Locks option when checking objects in to the iSeries Master repository from a Visual LANSA Slave. This will ensure that any new or modified objects will remain locked to the Task ID after they are checked into the LANSA for iSeries Master LANSA System and will be subject to status-based unlocking. (Refer to Unlock Objects in Task Tracking.)

If you use check-in unlocking, the object will be set to read-only on the workstation so that it is less likely that a developer can overwrite changes. They must check-out the object again before they can update it. However, if the object lock is released at check-in, it can now be used by a different task. You must be sure that the object is locked again so that it is included with the group of objects as part of the product.

Also See

Combined Approaches Share Task IDs ↑ 6.2.2 Approaches for using Task Tracking

Tracking By Developer

Task Tracking by developer assigns a single Task ID to a developer and is suited to small teams where there is little crossover between developers. Each developer uses their own task identifier regardless of the product they are working on. This approach primarily provides an audit trail of a developer's work. There are no product completion points as tasks can stay open indefinitely.

To use Task Tracking by Developer you must follow the steps as described in Set Special Task ID.

Tasks such as *U000BOB or *U00MARY can be created. These special Task IDs are defined to allow only one user to be authorized to the task.

Using this scenario you will typically not use the Keep Locks option when checking objects in to the iSeries Master repository from a Visual LANSA Slave. This will ensure that any new or modified objects are released from the Task ID after they are checked into the LANSA for iSeries Master LANSA System. (Refer to Unlock Objects in Task Tracking.).

If you use status-based unlocks, you must transfer the object to another task before another developer can access the object. (Refer to Transfer Object Locks.) With status-based locks, no other developer can access the object until the task is set to FIN releasing all object locks. Transferring is an acceptable practice when using Combined Approaches. For example, you may transfer the locked object from task *U00BOB to a full task-oriented Task ID such as TSK001. When all objects required under TSK001 have been transferred, TSK001 can be closed and used to export a group of changes. Hence Bob always locks his own work under *U00BOB, but objects are transferred to another task when a unit of work or product completion point is required. Task *U00BOB is never closed.

Also See

Choose Your Task Tracking Approach ↑ 6.2.2 Approaches for using Task Tracking

Minimum Tracking

It is recommended that Task Tracking is always enabled, however minimum task tracking is suited to small teams with highly trusted developers. Using this approach, there is only one task per system. All developers share the same task. (Refer to Share Task IDs.)

Only system internal task tracking activities are performed. There is no locking of objects. You cannot export by task. You cannot inquire on task history in any meaningful way.

Using *NONE (or equivalent Task ID)

You must follow the steps as described in Set Special Task ID.

A single permanent task such as *NONE can be created. Developers do not have to be authorized to this special Task ID. This task is generally never set to closed (status CLS) or finished (status FIN). If the *NONE task is closed, another Task ID such as *N01 can be created.

When you use this method, there will be no lock performed on the master system when an object is checked out because everyone is allowed to access the object.

Other forms of task tracking can be used in combination with this method. For example, you can create normal tasks that use Full Task-Oriented Tracking .

Sharing Task ID or Profiles

Another method of performing minimum task tracking is to create a normal Task ID and then authorize all developers to share this task ID. This approach can be used with Independent Visual LANSA Systems which cannot use Special Task IDs.

Another option of performing minimum task tracking with Independent Visual LANSA Systems is to allow developers to share a common user profile. Because locks are performed by workstation, not by user, Visual LANSA can still perform object locking.

Also See

Special Authorities and Task Tracking

↑ 6.2.2 Approaches for using Task Tracking

Combined Approaches

By using the LANSA's ability to Transfer Object Locks, it is possible to combine some (but not all) of the task tracking approaches.

Full Task-Oriented Tracking can be combined with Tracking By Developer. In this scenario, a task is created for managing a unit of work. This is the task that will also be used to export objects when the unit of work is complete. A developer can use their own task to lock and work on the required objects, and then transfer these objects to the full task-oriented Task ID. This combined approach simplifies task tracking for each developer while still having manageable units of work for exporting changes. It requires some additional administration for the assignment of objects to developers and checking that objects are properly transferred once work is completed.

Tracking by Product can be combined with Tracking By Developer. In this scenario, a task is created for working on a product. This is the task that will also be used to export objects when the project is complete. A developer can use their own Task ID to lock and work on the required objects, and then transfer these objects to the product-oriented Task ID. This combined approach allows simple tracking by developers while still having project level tasks for exporting changes. It requires some additional administration for the assignment of objects to developers and checking that objects are properly transferred once work is completed.

Minimum Tracking should not be combined with other task tracking approaches.

These are just a few examples of task tracking approaches. You may design an approach that meets your specific site needs.

Also See

Choose Your Task Tracking Approach

↑ 6.2.2 Approaches for using Task Tracking

6.2.3 Task Tracking Set up

Once you have selected from the 6.2.2 Approaches for using Task Tracking, you must set task tracking options to suit your approach. These steps will include:

- 1. Set Task Tracking System Options
- 2. Set Task Tracking Partition Options
- 3. Set Special Task ID
- 4. Define Task IDs

It is also recommended that you review the following:

Share Task IDs

Change Current Task

Unlock Objects in Task Tracking

Process/Function Locking

Transfer Object Locks

Task Tracking and Repository Synchronization

Special Authorities and Task Tracking

Task Tracking in Master/Slave Systems

Task Tracking on Independent Systems

Task Tracking Recommendations

Also See

6.2.4 What Happens When Task Tracking Is Active?

↑ 6.2 Using Task Tracking in LANSA

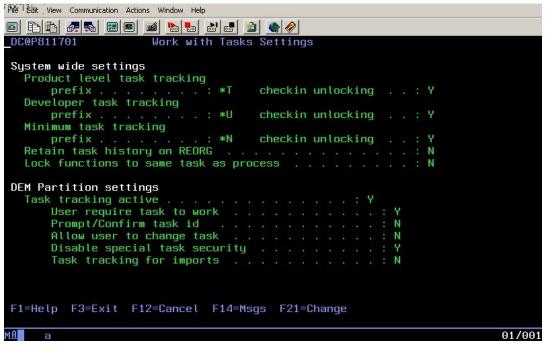
Set Task Tracking System Options

All Task IDs are defined at the system level in LANSA. If you have the appropriate authority, Task IDs can be created when logged on to any partition.

LANSA for iSeries Master System

The Lock functions to same task as process can be maintained on a LANSA for iSeries Master LANSA System under the Task Tracking System Settings available from the Administration Menu.

If this value is set to "Y", then all functions are required to be locked with the same Task ID as the parent process. If this value is set to "N" then functions within the same process may be locked to different tasks. This may be used when multiple developers using different tasks are working on functions in the same process.



This settings is stored in the data area DC@OSVEROP as *TTG6FUNCLOCKING.

A Process can be checked out as "Read only" and the function can be moved as "Update". This has the effect of checking out/ in the function without the process.

Setting Special Task Ids also updates the System Definition. (Refer to Set Special Task ID.)

Note: When Lock functions to same task as process is changed the system

definition in Visual LANSA must be refreshed and the host monitor stopped and restarted.

Refer to Task Tracking Settings in the *iSeries User Guide*.

Visual LANSA Independent System

It is recommended that you use Task tracking in Visual LANSA.

No specific system settings are required. For details about configuring a Visual LANSA Independent System, refer to Task Tracking on Independent Systems.

Also See

Set Task Tracking Partition Options

1 6.2.3 Task Tracking Set up

Set Task Tracking Partition Options

While the Task IDs are defined at the system level, some task tracking features must be set at partition level. If you are using a LANSA for iSeries Master System, then you will use the following partition settings:

• Enable task tracking (should be set on if using Visual LANSA)

Specify that task tracking is active in this partition. Object and task authority checks will be performed and all events that have taken place for work performed on objects will be recorded.

• Task is required

Specify that the user requires a task identifier before any work can be performed on a selected object.

• Confirm task ID

Specify that the confirm task identifier pop-up is required when work has completed on a selected object. The user will be prompted to confirm or change (if CHANGE function key is enabled) the task identifier to be allocated for work performed on the selected object.

• Allow task to be changed

Specify that the user is allowed to change the task identifier that is allocated to the selected object on which work was performed but only if.

• Disable task "work with" security

Specify that the security checks should be disabled within task administration.

• Active task tracking on imports

Specify that task tracking is active for import jobs.

If you are setting task tracking for LANSA for iSeries, refer to Partition Definitions - Create, Change or Delete and Task Tracking Settings in *the LANSA for iSeries User Guide*. (Refer to Task Tracking in Master/Slave Systems.)

If you are using a Visual LANSA Independent System, then you will use the Task Tracking Partition Settings. (Refer to Task Tracking on Independent Systems.)

Note: When Task Tracking Partition Settings are changed in a LANSA for iSeries Master LANSA System the partition definition in Visual LANSA must be refreshed.

Also See

Set Task Tracking System Options 1 6.2.3 Task Tracking Set up

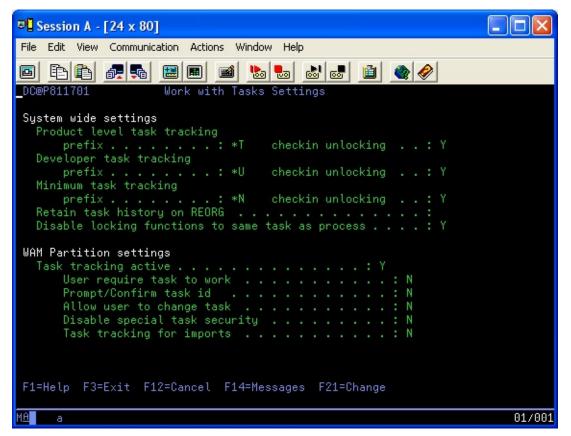
Set Special Task ID

Special Task IDs are only used when Visual LANSA Slave Systems are used with a LANSA for iSeries Master System. You cannot use special task IDs with a Visual LANSA Independent System.

Note: Full task-oriented tracking is the default and requires no additional setting up. (Refer to 6.2.2 Approaches for using Task Tracking.)

Three types of Special Task Id are supported, corresponding with product task tracking, developer task tracking and minimum task. The Special Task IDs setting is performed at system level.

The Special Task IDs can be maintained on a LANSA for iSeries Master LANSA System under the Task Tracking System Settings available from the Administration Menu.



A unique two letter prefix is required for each type of Special Task Id (that is, for product task tracking, developer task tracking and minimum task tracking). It is recommended that the default prefixes are used.

Tracking by Product	Insert the prefix for product-oriented tasks. It is recommended product-oriented task identifiers use "*T" as the first two characters.
	If you follow this recommendation, you could, for example, create task identifiers *T00000A and *T00000B for a Product A and Product B.
	Users need to be authorized to product-oriented tasks as usual.
	Note: Review this information via Task Tracking Settings, described in the <i>iSeries User Guide</i> .
Tracking By Developer	Insert the prefix for developer-oriented tasks. It is recommended developer-oriented task identifiers use "*U" as the first two characters.
	It is strongly suggested that an appropriate naming convention be used to identify tasks with users. If you use the recommended prefix, you could, for example, create task identifiers such as *U000BOB or *U00MARY.
	It is recommended that only one user is authorized to a developer- oriented task.
	Note: Review this information via Task Tracking Settings, described in the <i>iSeries User Guide</i> .
Minimum Tracking	Insert the prefix for Minimum Tracking tasks for the system. The recommended prefix for the single permanent task is "*N". A suggested name for this task, as only one task is required, is "*NONE".
	All users can use this task and they do not need to be authorized to it.
	Note: Review this information via Task Tracking Settings, described in the <i>iSeries User Guide</i> .
Unlocking fl	each type of Special Task Id is associated with a Check-in ag. When set to "Y", an object is unlocked from the task both on

the iSeries Master and on the Visual LANSA Slave when the object is checked in without the Keep Locks check-in option set in Visual LANSA. At the same time, it is changed to read-only on the Visual LANSA Slave. To change an object again on the workstation, you must check it out for update.

Note: The Check-in Unlocking flags can be viewed in Task Tracking Settings,

described in Task Tracking Settings in the LANSA for iSeries User Guide.

Note: When the Task Tracking prefixes or the Checkin Unlocking flags are changed the system definition in Visual LANSA must be refreshed and the host monitor stopped and restarted.

Also See

Set Task Tracking Partition Options ↑ 6.2.3 Task Tracking Set up

Define Task IDs

Once you have decided upon a task tracking approach (refer to 6.2.2 Approaches for using Task Tracking), updated the partition Task Tracking Settings (refer to Set Task Tracking Partition Options) and finished Set Special Task ID (if required), you will begin to define your Task IDs.

Before creating tasks, you should define a set of site naming standards for your Task IDs. These standards need to conform to the Special Task Ids if they are being used. Naming standards can help developers to identify the purpose of a task.

The basic steps to defining new tasks include:

- 1. Create a Task ID. (Refer to Task Maintenance.)
- 2. Assign the user or users to the Task ID. (Refer to Share Task IDs.)
- 3. If necessary, lock objects to the task and PC using an export. (Refer to How to Export from the IBM i in the *Installing LANSA on Windows Guide*.)
- 4. Inform developer(s) about the new tasks. Developers may need to update their current task lists. (Refer to Current Task List.)

How the developers use the tasks will depend upon your approach and how you have specified Unlock Objects in Task Tracking. A Task ID must be entered when a developer starts Visual LANSA. (Refer to Logon Parameters.)

Also See

Transfer Object Locks ↑ 6.2.3 Task Tracking Set up

Share Task IDs

One of the most important questions to ask when planning your task tracking strategy is, "How many developers will be allowed to use a Task ID?". The answer to this question will determine the types of controls you are able to implement.

There are two basic approaches:

One Developer per Task ID

You can assign a single developer for each Task ID. Using this approach, you are able to identify the work done by a single developer. One developer per Task ID is required if using the Tracking By Developer approach. It is also used with the Full Task-Oriented Tracking approach for greater control over developers.

If you wish to limit one developer per Task ID, it is your responsibility to assign just a single developer to a Task ID. LANSA allows up to 10 users or group profiles per task.

Multiple Developers per Task ID

You can allow more than one developer to share a Task ID. In the case of the Minimum Tracking approach, there is just one Task ID for the whole system and it is shared by all developers. The Tracking by Product and Full Task-Oriented Tracking approaches can be defined with more than one developer using the same Task ID. Conceptually, allowing multiple developers per Task ID is like allowing multiple developers to use the same user profile to access LANSA.

If more than one developer is allowed to use a specific Task ID, the task history must be reviewed at a detailed level to track an individual developer's work (assuming developers use different user profiles).

Multiple developers can be authorized to use a Task ID when it is created. LANSA allows up to 10 users or group profiles per task.

Also See

Unlock Objects in Task Tracking Set Special Task ID ↑ 6.2.3 Task Tracking Set up

Change Current Task

A developer must select a Task ID when logging on to Visual LANSA. (Refer to Logon Parameters.) You may change the current task by exiting Visual LANSA and logging on using a new Task ID, or you may use the *Set Current Task* option to change the task you are currently working with.

The Set Current Task option is accessed using the Repository tab. (Refer to Using the Repository Tab in the User Guide.) Expand the list of Tasks and select the desired Task ID. Right-click to display the pop-up menu and select the Set Current Task option. This option can only be used if there are no objects open in the editor.

LANSA will check if you are authorized to work under the task you have selected. If you are not allowed to work under the task, then an error message will be displayed. The new task you select will take effect immediately. The new task you have selected will apply to all the work you perform in Visual LANSA. The title bar of the LANSA window will reflect the change in the task ID.

1 6.2.3 Task Tracking Set up

Unlock Objects in Task Tracking

When controlling changes, the concept of task tracking is based on locking objects to a task. When an object is locked by a Task ID, it cannot be allocated to another Task ID until the lock is released. Hence you can control when changes are being made.

Normal Task IDs, that are used with the Full Task-Oriented Tracking approach, support only status-based unlocking.

For the specialized Task IDs, LANSA supports two basic approaches:

Status-based Unlocking

- By default, all tasks use a status-based unlocking mechanism. When a task has a status of Open (OPN), Work (WRK) or Closed (CLS), associated objects are locked to the task. Once a task status is changed to Finished (FIN), the objects are unlocked. (Refer to Task Status.)
- If you are using status-based unlocking, you will typically be creating many different tasks. As work is completed you must set tasks to finished or Transfer Object Locks.
- This form of unlocking ensures that a group of objects are locked and unlocked as a unit of work. You can create an export list of objects using a Task ID.
- Status-based tracking must be used if you wish to use a Full Task-Oriented Tracking approach.

Note: All normal tasks can use only this method of unlocking.

PC Name Unlocking

- By default, all tasks will allow this form of unlocking. When an object is checked out, in addition to being (or remaining) locked to a task, it is also (or remains) locked to a PC Name.
- This form of unlocking is primarily used to ensure that an object is being updated on only one PC at a time.
- If a developer wants to check-in source changes to be archived before they have finished development on the object, they can use the Keep locked to PC option on check-in. This will retain a PC Name lock for the object both on the server (Master) and PC (Slave). This can be used to ensure no other PCs can check-out the object until development has been completed on the original PC. In RDML partitions, it will also ensure that no other

development on the object can be done in LANSA for iSeries.

• Check-in Unlocking will also perform PC Name Unlocking.

Note: If you are using an Independent System, the PC Name is not used to lock the object permanently. Refer to Task Tracking on Independent Systems.

Check-in Unlocking

- This form of unlocking can only be used with Special Task Ids and must be set up on the iSeries Master. (Refer to Set Special Task ID.)
- By default, an object is unlocked from a task both on the server (Master) and workstation (Slave) system when it is checked into the server. At the same time, it is changed to read-only on the workstation system. To change an object again on the workstation, you must check it out for update.
- This form of unlocking is primarily used to ensure that an object is not being updated by more than one developer at a time. The task is not being used to group objects under a task.
- If a developer wants to check-in source changes to be archived before they have finished development on the object, they can use the Keep Locks option on check-in. This will retain an update lock for the object both on the server (Master) and workstation (Slave). This can be used to ensure no other developer can allocate a different task to the object until development has been completed.
- Note: If you are using an Independent System, you cannot use this type of unlocking. Refer to Task Tracking on Independent Systems.

When you decide to unlock objects will determine the types of controls you are able to implement.

Also See

Transfer Object Locks ↑ 6.2.3 Task Tracking Set up

Process/Function Locking

In LANSA, whenever a function is moved, the process is also moved to protect the structure, that is, there must be a process for the function to be contained within.

The Lock functions to same task as process setting on the LANSA for iSeries Master System will disable the task tracking logic that enforces that all functions to be locked with the same Task ID as the parent process. This flag only applies when the Allow user to change tasks while working? task tracking option is set to NO. This setting is stored as the *TTG6FUNCLOCKING value in the DC@OSVEROP data area.

Object	Task	Check Out Mode: U=update, R=read only	*TTG6FUN CLOCKING	DC@A07 check-in unlocking flags	Checked In	I t I a (
Process	Normal	U	OFF	N/A	Y	Ŋ
Function	Task	U			Y	Ŋ
Process	Normal	R	OFF	N/A	Y	Ŋ
Function	Task	U			Y	Ŋ
Process	Normal	U	ON	N/A	Y	Ŋ
Function	Task	U			Y	Ŋ
Process	Normal	R	ON	N/A	Ν	Ţ
Function	Task	U			Y	Ŋ
Process	*N	U	OFF	N	Y	1
Function		U			Y	1
Process	*N	R	OFF	N	Y	1
Function		U			Y	1
Process	*N	U	OFF	Y	Y	1
Function		U			Y	1

Summary of Process/Function movement with various system settings

Process	*N	R	OFF	Y	Y	1
Function		U			Y	1
Process	*N	U	ON	N	Y	1
Function		U			Y	1
Process	*N	R	ON	N	Ν	1
Function		U			Y	1
Process	*N	U	ON	Y	Y	1
Function		U			Y	1
Process	*N	R	ON	Y	Ν	1
Function		U			Y	1

Note: *U and *T tasks perform the same way.

Object	Task	Check Out Mode: U = update, R = read only	*TTG6FUN CLOCKING	DC@A07 check-in unlocking flags	Checked In	Locke task ir IBM I after Check
Process Function	*U & *T	U U	OFF	-NN	Y Y	Y Y
Process Function	*U & *T	R U	OFF	-NN	Y Y	Y Y
Process Function	*U & *T	U U	OFF	-YY	Y Y	N N

Process	*U	R	OFF	-YY	Y	Ν
Function	& *T	U			Y	Ν
Process	*U	U	ON	-NN	Y	Y
Function	& *T	U			Y	Y
Process	*U	R	ON	-NN	Ν	unchar
Function	& *T	U			Y	Y
Process	*U	U	ON	-YY	Y	Ν
Function	& *T	U			Y	Ν
Process	*U	R	ON	-YY	Ν	unchar
Function	& *T	U			Y	Ν

If the process and function are placed in Visual LANSA by propagation, they will be read only.

If the process and function are moved to Visual LANSA by export, the process can be set as read only or update.

WEBEVENT applications require special consideration as there are several HTML definitions associated with a process which can cause overwriting problems (e.g. LAYOUT, MENU, etc). Various web settings for Import and Export processing are available on the iSeries Master. These are described in Export and Import settings . These settings can be set as required to restrict the movement of associated HTML pages.

For example, you may want to move the HTML for the function but not the HTML for the process. This can be achieved by moving the Process as "Read only". The function can be moved for "Update". This has the effect of checking out/ in the function without the process. This can also be achieved by using the Lock functions to same task as process setting. The HTML definitions included in the import / export are determined by the Import and Export system settings.

Pl Session A - [24 x 80]	×
File Edit View Communication Actions Window Help	
DC@P811901 Work with Export and Import Settings	
Compile and Export OS/400 target release *CURRENT Export LANSA object authority MD	
Include VL components in iSeries export/import : Y Include VL components between iSeries and VL : Y Include Web details in export Y Include Web details in import	
Include XML details in export Y Include XML details in import	
User defined Export/Import message logging program : in library	
Reference field propagation in import Y	
Export \$\$DTALIB\$\$ substitution default value : V11WEXLIB and Prompt on Import default : Y	
F1=Help F3=Exit F12=Cancel F14=Messages F21=Change	
MA a 81/	/001

Note: When Lock functions to same task as process setting is set or the Checkin Unlocking or Export and Import settings are changed, the system definition in Visual LANSA must be refreshed and the host monitor stopped and restarted. Ensure that the host monitor has stopped before restarting.

1 6.2.3 Task Tracking Set up

Transfer Object Locks

LANSA's task tracking can be set to allow a locked object to be transferred from one Task ID to another Task ID. This is a very powerful ability that increases the flexibility of how you manage tasks.

When an object is locked by a task, it cannot be accessed by another task until the current task releases its lock. When status-based locks are used, the Task Status must be set to Finished (FIN) for the lock to be released. By transferring a locked object to another task, you no longer have to set the current task to Finished.

The ability to transfer object locks supports Combined Approaches to task tracking. For example, when using Tracking By Developer, you do not want to be closing a developer's task to release object locks. Instead, you can transfer a locked object to another task supporting Tracking by Product.

Also See

Unlock Objects in Task Tracking

1 6.2.3 Task Tracking Set up

Task Tracking and Repository Synchronization

Repository Synchronization allows changes made to the LANSA for iSeries Repository to be propagated to Visual Repositories. Repository synchronization includes updates to task tracking and other partition information.

When a change is made to an object, it is automatically propagated by LANSA for iSeries sending read-only version of the transaction to the Visual LANSA Slave Repositories. If an object is currently checked out for update in a Visual LANSA Slave Repository, the propagation overwrites this object and the Visual LANSA Slave Repository user will need to check the object out for update again before they can perform any modifications. This situation can be avoided by ensuring the developer who has checked out the object for update is the only developer authorized to the task they are using and hence the only developer modifying the object.

Also See

6.3 Repository Synchronization6.2 Using Task Tracking in LANSA↑ 6.2.3 Task Tracking Set up

Special Authorities and Task Tracking

LANSA has two special user authorities:

LANSA System Owner

- The LANSA System owner has authority to all objects in LANSA.
- Task tracking and security rules do not apply to the LANSA System owner. This user ID can use any open task ID and will be allowed to access any objects in the system regardless of the task ID being used. Be very careful when using the LANSA System Owner ID.

Partition Security Officer

- The Partition Security Officer is nominated on the partition definition. This user profile has authority to all objects in the LANSA partition.
- A user ID, which is the partition security officer or part of the QSECOFR group, is exempt from Special Authorities and Task Tracking rules. These user IDs can use any open task ID and will be allowed to access any objects in the system regardless of the task ID being used. Be very careful when using these types of user IDs.

It is recommended that proper controls be placed on these users. The use of these profiles is not recommended when using a Minimum Tracking approach.

If you are configuring LANSA for iSeries Master System, refer to Partition Definitions - Create, Change or Delete and the System Definition Data Area in *the iSeries User Guide*.

If you are configuring a Visual LANSA Independent System, refer to System Data and Partition Description.

Also See

6.2.2 Approaches for using Task Tracking

1 6.2.3 Task Tracking Set up

Task Tracking Recommendations

Following are some general recommendations when using task tracking:

- Remember, task tracking is required in Visual LANSA as one of the Logon Parameters if you are using an iSeries Master Repository. You should plan your task tracking strategy before you begin LANSA development.
- Train your developers about task tracking. All developers must understand how you are using task tracking at your site. You must explain the development procedures relating to your task tracking strategy.
- Task tracking is just one part of you change management strategy. It is not the complete solution. To use task tracking effectively, you need properly defined standards and procedures in place. For example, you need procedures for creating Task IDs and assigning the Task IDs to developers, as well as procedures for updating current task lists.
- Create site naming standards for your Task IDs. Naming standards are very important and can help to simplify task tracking. These naming standards may need to consider Set Special Task ID.
- If you are using multiple LANSA Systems, you must create a task tracking strategy that properly manages Task IDs across the LANSA Systems. For example, if you create an identifier, TASK001, it should have the exactly the same definition on all LANSA Systems to allow objects to be migrated using Task IDs.
- If you have a large, complex, development environment, you should consider using 6.1.5 Third Party Packages to assist in implementing your change management strategy.
- Repository synchronization can be used to ensure that task tracking details and modifications are updated in all Visual LANSA Repositories. (Refer to Task Tracking and Repository Synchronization and 6.3 Repository Synchronization.)
- Keep it simple. Choose a task tracking strategy that allows developers to be productive without excessive administration.

Remember, you should have an overall change management strategy with properly defined standards and procedures for your application development environment. (Refer to 6.1 Change Management Concepts.)

Also See

Task Tracking in Master/Slave Systems
Task Tracking on Independent Systems
6.2.2 Approaches for using Task Tracking
↑ 6.2.3 Task Tracking Set up

Task Tracking in Master/Slave Systems

If your Visual LANSA System is connected to a LANSA for iSeries Master System, then the following rules will apply:

- All Task IDs will be created and maintained in the LANSA for iSeries Master System. (Refer to Working with Tasks in *the LANSA for iSeries User Guide*.)
- All partition and other task tracking settings are set up in the LANSA for iSeries Master System. (Refer to Task Tracking Settings in *the LANSA for iSeries User Guide*.)
- You must have task tracking turned on in the LANSA for iSeries Master System in order to ensure that all objects are properly tracked. Do not turn off task tracking in the Master System if you have Slave systems with active development. (Refer to Create, Review, Change or Delete Partition Definitions of the LANSA for iSeries User Guide.)
- Task tracking must be used in Visual LANSA. It is a required parameter at logon. (Refer to Logon Parameters.)
- All user profile and Task ID information must be refreshed in the Slave System whenever a change is made to the LANSA for iSeries Master System. (Refer to Current Task List.)
- You must decide if Repository Synchronization will be used. (Refer to 6.3.1 Repository Synchronization Concepts and Task Tracking and Repository Synchronization.)
- You must be aware of the Task Tracking settings for Unlock Objects in Task Tracking when Set Special Task ID.
- When objects are exported from a Master system, they can be assigned a Task ID and a PC Name. (Refer to How to Export from the IBM i in the *Installing LANSA on Windows Guide*.
- When objects are checked out for update from a Master system, they are assigned both a task ID and the PC Name of the PC they are checked out to.
- The partition security officer and partition owner has special authorities. (Refer to Special Authorities and Task Tracking.)

Also See

Share Task IDs Task Tracking and Repository Synchronization Task Tracking on Independent Systems
6.2.2 Approaches for using Task Tracking
Maintaining Users and Tasks on Slave Systems.
↑ 6.2.3 Task Tracking Set up

Task Tracking on Independent Systems

If your Visual LANSA System is an Independent System, it can directly maintain its own LANSA System definition data including task details. The following rules will apply:

- All Task IDs will be created and maintained directly in Visual LANSA.
- There are fewer settings required for Independent Visual LANSA systems because a distributed development environment is not used (i.e. there are no slave repositories). There is no need for any form of repository synchronization. (Refer to 6.3.1 Repository Synchronization Concepts.)
- All partition and other task tracking settings are set up in Visual LANSA. (Refer to Task Tracking Partition Settings.)
- Check-in unlocking is not relevant as this is not a distributed development model. (Refer to Unlock Objects in Task Tracking.)
- Object locking should be turned on in Visual LANSA. For example, it is the only method that guarantees that two developers cannot edit an object concurrently. (Refer to 6.1.1 Object Locking Concepts .)
- The partition security officer and partition owner has special authorities. Refer to Special Authorities and Task Tracking.
- You cannot use the Special Task Ids. (Refer to Set Special Task ID.) Hence, you typically use a Full Task-Oriented Tracking or a Minimum Tracking.
- Task tracking is typically required with Visual LANSA server configurations where multiple developers are sharing a single repository.

If you have a single user independent workstation, minimal task tracking may be required as you are not sharing the repository with other developers. However, you may still wish to use Task IDs when objects are being deployed to other Visual LANSA Systems.

Also See

Task Tracking in Master/Slave Systems
6.2.2 Approaches for using Task Tracking
Maintaining Developers and Tasks IDs
↑ 6.2.3 Task Tracking Set up

6.2.4 What Happens When Task Tracking Is Active?

When task tracking has been activated for a partition, all development work within LANSA will be monitored. When an object is selected (to create, compile, change, delete or review), task tracking will execute the following procedures every time:

- 1. After the user has selected an object to work with and the normal LANSA object security checks have allowed work to commence, task tracking makes sure that:
 - A valid task identifier has been specified if the user requires a task identifier to work in LANSA. A valid Task identifier must have a status of either "OPN" or "WRK" and must include the user in the list of authorized users/groups for the Task Identifier, or the User Identifier must be the LANSA partition security officer user/group profile or QSECOFR user/group profile.
 - The user is allowed to change Task Identifiers and is authorized to work with the task that is allocated to the object if the object that has been selected to work with is allocated to another task identifier.
 - The object is either not already locked to a task or is already locked to the user's task. If the object is already locked to the user's task that it is either not locked to any PCs or is already locked to the user's PC.
- 2. After the user has completed work on the selected object, task tracking does the following:
 - A task identifier has been allocated to the selected object.
 - If any of the checks performed in Step 1 have been found to be false, a message will be issued stating that "Work has not been committed" and LANSA database changes will not be performed.
- 3. If work has completed successfully for the object, LANSA internal database files are updated with the object changes and the Task Tracking database files are updated with the details of the events that have taken place.

Note: The current user Task ID is shown in the Visual LANSA status bar. The Task ID locked to an object can be displayed in the repository.

Also See

Task Tracking and Repository Synchronization 6.2.6 Task Tracking and Import/Export

6.2.5 Typical Development Cycle Example

Following is an example of a typical development cycle that might be used with Full Task-Oriented Tracking with a LANSA for iSeries Master System:

- 1. Create a Task ID on the iSeries following site naming conventions.
- 2. Use System Initialization or propagation to make the new Task ID available in Visual LANSA.
- 3. Check out the required objects for update, as required, using the Task ID or start creating objects using this Task ID.
- 4. Maintain the Objects (Create, Edit, Compile).
- 5. Check in the objects to the iSeries Master System and Compile (and test if required).
- 6. Close the Task ID on the iSeries once the project is completed.
- 7. Use System Initialization or propagation so that the Task ID status is updated in Visual LANSA.
- 8. Create or maintain iSeries Export lists using the Task ID as required.

It is recommended that you create an export list for that specific task and the addition of that task to the export for the next release.

9. Create or maintain a Deployment Package in Visual LANSA using Task ID.

It is recommended that you create a package for that specific task and the addition of that task to the package for the next release.

- 10. Export the Task specific export list and import to your test partition on iSeries.
 - a. Update any test scripts to include new capabilities.
 - b. Test the changes delivered in that task.
 - c. Update documentation with new capabilities.
- 11. Deploy the Task specific package to your test partition on Windows.
 - a. Update any test scripts to include new capabilities.
 - b. Test the changes delivered in that task.
 - c. Update documentation with new capabilities.
- 12. Deploy the Task specific package to your test partition on Linux.

- a. Update any test scripts to include new capabilities.
- b. Test the changes delivered in that task.
- c. Update documentation with new capabilities.
- 13. If issues exist:
 - a. Change the status of the task on the iSeries back to 'WRK'.
 - b. Use System Initialization or propagation so that the Task ID status is updated in Visual LANSA.
 - c. Go to step 3.
- 14. Change Task ID to 'FIN' status.
 - a. Refresh the Windows Development Server (Releases Objects from Task ID).
 - b. Use System Initialization or propagation so that the Task ID status is updated in Visual LANSA.
- ↑ 6.2 Using Task Tracking in LANSA

6.2.6 Task Tracking and Import/Export

When creating an export list on LANSA for iSeries, you can add all objects to the export list (fields, files, functions, processes, application templates, system variables, multilingual variables and web components) for the current partition that have been worked on under a task identifier. This process can be repeated within a single export list to include more than one task identifier.

All task identifiers that are currently set to 'CLS' (Closed) status, and that the user is authorized to, will be available for selection to export. When a task identifier has been selected, all objects for the current partition that have been worked on under the task identifier will be automatically included in the export list.

The ability to include objects in export lists based on tasks is very helpful when using a Full Task-Oriented Tracking approach. All objects for a unit of work can be exported from a development system to a test or production system.

The Visual LANSA Deployment tool also has the ability to add objects to packages based on task.

Also See

Add All Objects Worked on Under a Task Identifier in the LANSA for iSeries User Guide.

Objects from Tasks in the Visual LANSA Deployment Tool Guide.

1 6.2 Using Task Tracking in LANSA

6.3 Repository Synchronization

Repository Synchronization is set on in the LANSA for iSeries Master System to maintain a current list of objects and other information in specific Visual LANSA Slave Repositories.

- 6.3.1 Repository Synchronization Concepts
- 6.3.2 What Are Repository Groups?
- 6.3.3 What are Work Groups?
- 6.3.4 Rules for Repository Synchronization
- 6.3.5 Synchronization Options in Visual LANSA
- 6.3.6 PC Options on Server
- 6.3.7 Repository Synchronization Tips & Techniques

Also See

LANSA PC Development in the LANSA for iSeries User Guide.

Host Monitor

1 6. Change Management

6.3.1 Repository Synchronization Concepts

Repository Synchronization is a feature that allows changes made to a LANSA for iSeries Master Repository to be propagated to Visual LANSA Slave Repositories in order to maintain current object and system information in the Slave Repositories. For example, if a Visual LANSA developer checks in a new field to the LANSA for iSeries Repository, the list of fields shown in other Visual LANSA Repositories would need to be updated in order to match the LANSA for iSeries Master System.

Repository Groups are set up in LANSA for iSeries to reflect how the Visual LANSA repositories are connected to the iSeries repository so that information can be automatically propagated from the iSeries to the Visual LANSA repositories. The propagated information consists of changes to task tracking and partitions on the iSeries, updated information checked in to the iSeries by a member of a work group, and any user initiated requests from the main development environment "work with" panels on the iSeries.

Not all types of changes to objects made by a developer on the iSeries are automatically propagated, but the changes may be manually propagated. For example, if a field is created by a developer using LANSA for iSeries, this new field is not automatically propagated to Visual LANSA. Also, when a developer using Visual LANSA creates a field, it is not automatically propagated. But when a Visual LANSA developer checks-in a new field to the LANSA for iSeries Repository, this change is automatically propagated to other Visual LANSA Repositories. Repository synchronization is based upon a Visual LANSA centric development model where LANSA for iSeries hosts the master repository but all development work is performed using Visual LANSA workstations. (Refer to 6.3.4 Rules for Repository Synchronization.)

A change is propagated by automatically sending a check out for read-only transaction to a slave workstation. Note that if an object is checked out for update in a Visual LANSA Repository, the propagation still overwrites this object. Thus, it is recommended that only one user updates an object at any one time. (Refer to 6.2 Using Task Tracking in LANSA and Task Tracking and Repository Synchronization for more information.) A user on this Visual LANSA Repository would need to check the object out for update again in order to be able to modify it. Remember, check out is always partition-specific. Objects are only checked out to the allowed partitions defined in the PC definition of the member.

A manual developer option also exists in LANSA for iSeries to propagate changes. Refer to Propagating Objects from the iSeries and the Host Monitor.

Also See

LANSA PC Development in the LANSA for iSeries User Guide.

6.3.2 What Are Repository Groups??

6.3.2 What Are Repository Groups?

A repository group is a list of all the Visual LANSA PCs that have a repository that must be synchronized with a LANSA for iSeries System. The Visual LANSA Repository can be part of either a workstation or a server configuration. The use of repository synchronization requires repository groups to be created on LANSA for iSeries. Each Visual LANSA PC can only be in one repository group.

The word "group" can be misleading. There is just one Visual LANSA Repository allowed in each repository group. If you have three Visual LANSA Slave Workstation PCs, then you must create three separate repository groups. When Visual LANSA is used in a server configuration, there is still just one Visual LANSA Repository in each group, but there may be several PCs listed that use the repository. Hence, the group of PCs that share the single Visual LANSA Repository make up a "repository group".

How you set up repository groups depends on how the PCs and the repositories are connected to the iSeries repository:

- In the simplest case, you create a repository group for a single Visual LANSA PC which has a repository (i.e. a slave workstation configuration).
- In a configuration with a network server containing a repository and several client PCs, you create one repository group to include both the server and the PCs.

In a repository group containing more than one PC, one PC must be identified as the repository gateway. This PC uses the Host Monitor to receive the propagations and store them in the Visual LANSA Repository. (In a slave workstation configuration, the single PC is the gateway as there is only one PC in the repository group.)

In a network server configuration, the PC selected to be the gateway must have the host monitor installed, but it does not have to be the PC with the repository. In this configuration, the gateway is often the server. If the PC identified as the repository gateway is deleted without another PC being identified as the new repository gateway, then the first PC in the repository group is considered to be the repository gateway for the repository group.

In order to set up repository groups, you need to answer at least the following questions:

• What Visual LANSA repositories need to take part in repository

synchronization?

This gives you the repository groups, one for each repository.

- Which PC will be the gateway for each repository? The gateway is the PC that will be sent the propagations for a particular repository. This is the first member of the repository group.
- What other PCs use each repository? This gives you the other members of each repository group.

A further refinement can be added to restrict which changes go to which repositories. This is referred to as a work group. When no work groups are defined, all propagations go to all repositories. (Refer to 6.3.3 What are Work Groups??)

Also See

6.3.1 Repository Synchronization Concepts

- 6.3.4 Rules for Repository Synchronization
- ↑ 6.3 Repository Synchronization

6.3.3 What are Work Groups?

A work group is a list of Visual LANSA PCs that share the same group of objects and need to be synchronized as a separate group. This use of work groups is optional. Use of repository synchronization does not require work groups to be created in order for changes to be propagated. When no work groups are defined, all propagations go to all repositories. If one work group exists, and the Visual LANSA PC originating a change is not in a work group, the change is not propagated.

For example, the Visual LANSA PCs of a group of developers working on financial applications for a company could form one work group so that they share the latest versions of objects used by the financial applications.

A Visual LANSA PC can be in many work groups or in no work groups at all. It does not have to belong to a repository group. A work group must contain at least two PCs.

The members of a work group can be in the same or different repository groups. When a PC in a work group checks in objects to the iSeries using the host monitor, copies of all these objects are checked out to the repositories of the other members of the group. Where a work group contains two or more PCs defined in the same repository group, the propagations are sent only once to the repository gateway.

In order to set up work groups you just need to know which Visual LANSA PCs share the same work and want to automatically receive changes that a co-worker makes.

Consider the following scenarios involving three developers:

Scenario 1

John, Bob and Mary each have their own Visual LANSA PC workstation. Each PC is defined as a repository group, i.e. there are three repository groups defined. No work groups are defined. Whenever John, Bob or Mary checks-in a change to LANSA for iSeries, the change is propagated to all developer PCs using repository synchronization.

Scenario 2

Work Group 1	
Bob	
Mary	

Bob and Mary are working on the same application. When Bob checks-in his new objects to LANSA for iSeries, Mary needs to know about these changes, but not John. A work group is created for Bob and Mary to keep their repositories synchronized. John is not part of this work group. When John checks-in his changes, repository synchronization will not update Bob or Mary.

Scenario 3

Work Group 1	Work Group 2		
Bob	John		
Mary	Bob		

John and Bob have created a second work group. When John checks-in his changes, they are sent to Bob but not to Mary. Likewise, Mary's changes are sent only to Bob. Because Bob is part of both work groups, his changes are sent to both Mary and John.

Also See

6.3.1 Repository Synchronization Concepts

6.3.2 What Are Repository Groups??

6.3.4 Rules for Repository Synchronization

↑ 6.3 Repository Synchronization

6.3.4 Rules for Repository Synchronization

The following rules summarize repository synchronization for LANSA for iSeries and Visual LANSA:

- A fundamental part of repository synchronization is LANSA for iSeries. Without LANSA for iSeries, no propagations can occur.
- A Visual LANSA PC must be in a repository group or work group in order for its repository to receive any propagations. (Refer to 6.3.2 What Are Repository Groups?)
- Each Visual LANSA PC can be in only one repository group.
- A repository group contains only one Visual LANSA Repository, but could have many PCs listed if a server configuration is being used.
- When no work groups are defined, all propagations go to all repository groups. (Refer to 6.3.3 What are Work Groups?)
- A Visual LANSA PC can be in one or more work groups.
- The Host Monitor must be running on the repository gateway for changes to the current partition to be received. Propagations are queued on the iSeries until the Host Monitor is started in the appropriate partition.
- Changes to objects made by a developer using LANSA for iSeries are not automatically propagated, but can be manually propagated. (Refer to Propagating Objects from the IBM i.)
- Changes to objects made by a developer using Visual LANSA are automatically propagated when the change is checked in to the LANSA for iSeries Repository. (Refer to table below.)
- When LANSA messages are created, changed or deleted using LANSA for iSeries, they are not propagated to Visual LANSA.
- When LANSA messages are created, changed or deleted using Visual LANSA, they are propagated to LANSA for iSeries and other Visual LANSA Repositories.
- You can specify for each Visual LANSA partition how deleted objects are propagated. (Refer to 6.3.5 Synchronization Options in Visual LANSA.)

The following table summarizes how changes made to objects using the LANSA for iSeries development environment are propagated to Visual LANSA:

Operation performed in LANSA for iSeries

Delete Object*	Yes
Change Object	No
Create Object	No

The following table summarizes how changes made to objects using the Visual LANSA development environment are propagated to LANSA for iSeries and other Visual LANSA Repositories:

Operation performed in Visual LANSA	Propagate to LANSA for iSeries	Propagate to other Visual LANSA Repositories
Delete Object*	Yes	Yes
Change (without check-in)	No	No
Create (without check- in)	No	No
Check-in Changed Object	Yes	Yes
Check-in Created Object	Yes	Yes

* Developers can control how deleted objects are propagated.

Also See

- 6.3.1 Repository Synchronization Concepts
- 6.3.7 Repository Synchronization Tips & Techniques
- ↑ 6.3 Repository Synchronization

6.3.5 Synchronization Options in Visual LANSA

Options are available within the Visual LANSA *Partition Maintenance* settings to control the action of propagated deletes. These are:

- Ignore propagated deletes
- Perform checks before propagating.

Selecting the *Ignore propagated deletes* option will cause propagated deletes to be ignored. When an object is deleted in LANSA for iSeries, the local Visual LANSA repository is not updated.

Selecting the *Perform checks before propagating* option will cause propagated deletes to perform referential integrity checks before performing the delete. This will prevent, for example, the deletion of a field that has been used on a local file.

When repository groups are added (but not creating PC definitions), a refresh of PC definitions is not required. The next Check In made from a PC in a repository group will be propagated to the other repositories.

When new PCs are defined, the enrolled PCs must be updated. Refer to Maintaining Users and Tasks on Slave Systems for information.

Also See

6.3.4 Rules for Repository Synchronization

6.3.7 Repository Synchronization Tips & Techniques

6.3.6 PC Options on Server

Host Monitor

↑ 6.3 Repository Synchronization

6.3.6 PC Options on Server

The LANSA PC Development section in the *LANSA for iSeries User Guide* describes a complete set of PC tasks relating to Visual LANSA. This includes:

- Defining Personal Computers to LANSA
- Repository Synchronization
- Working with Repository Groups
- Working with Work Groups
- Propagating Objects from the iSeries

Also See

6.3.4 Rules for Repository Synchronization6.3.5 Synchronization Options in Visual LANSAHost Monitor

↑ 6.3 Repository Synchronization

6.3.7 Repository Synchronization Tips & Techniques

Following are some general tips when using repository synchronization:

- Do not leave lots of developers in a work group if they are not actively working on the development. Consider removing inactive developers from the group, and then when they need to get the updates, get a complete refresh by means of an import. After the import, you can add them back into the work group if required.
- When there is a large volume of changes, an import is an efficient update option to consider. For example, if a developer has been away on vacation for two weeks, you should remove them from the work group and use an import when they return to work.
- When developers are working, leave the Host Monitor running. Changes will be propagated as they happen.
- Developers should consider checking in changes more frequently, in logical units of work, instead of performing a large volume of changes all at once.
- If the Visual LANSA Host Monitor is stopped before a synchronization is complete, data may be left in transit. If the Visual LANSA Host Monitor is restarted before a LANSA for iSeries database reorganization, then it will continue from where it stopped and data will not be lost. A LANSA for iSeries database reorganization removes data that's in transit. So, to ensure there is no data in transit, make sure that synchronization is complete before stopping the Visual LANSA Host Monitor.
- When deleting a field on a slave system, an integrity check is run **only on that system**. Slave systems cannot be relied upon to have the whole repository and therefore this check is not a true integrity check. When the delete is propagated to the Master, an independent integrity check is performed. If the field is found to be used in a file, then the field will not be deleted on the master.

There are at least two ways you can successfully use the slave system to delete an object from both master and slave:

- 1. If the field to be deleted is used in a file and if that file is no longer required, on the slave system, delete the file definition, then delete the field. **Note:** Make sure you select the option *Delete from host repository* for each delete, to propagate the delete to the Master.
- 2. If the file is still required, on the slave system, delete the field from the

file definition and check the updated definition into the Master. Then delete the field, making sure you select *Delete from host repository* to propagate the delete to the Master.

↑ 6.3 Repository Synchronization

7. System Information

System maintenance is carried out on a LANSA Master System. This means that in a Visual LANSA Slave system, partition information is displayed as read only.

If you are using an Independent Visual LANSA System, partitions are maintained directly in Visual LANSA.

To access the System Information in the LANSA Editor, select *System Information* in the *Repository* tab.

You must be logged on to Visual LANSA as the Security Officer or Partition Security Officer to perform system information maintenance tasks.

The items that can be accessed from within the System Maintenance list are:

7.1 System Definitions	7.7 Primitives
7.2 X_LANSA.PRO Maintenance	7.8 Remote Systems
7.3 Partitions	7.9 User and Security Maintenance
7.4 Language Settings	7.10 Task Maintenance
7.5 Frameworks	7.11 Object Maintenance
7.6 Groups	7.12 Application Templates

Also See

Environment Settings

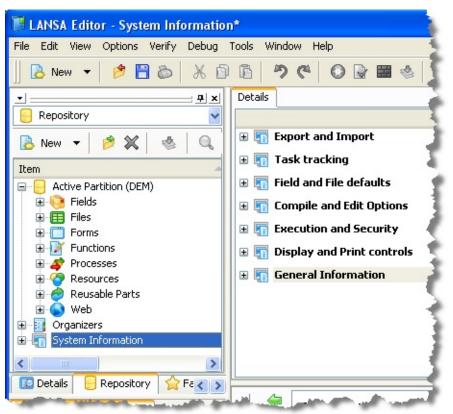
7.1 System Definitions

In a Visual LANSA Slave system, all System Definitions are is displayed as read-only in the *Details* tab as all maintenance is performed on the related LANSA Master System.

You will find the equivalent IBM i system definitions in the Review System Settings which are obtained from the *Administration* menu. For details of how to keep these system definitions up-to-date on a Slave system, refer to Visual LANSA Slave Systems Administration.

If you are using an Independent Visual LANSA System, you can change these details directly in Visual LANSA.

You can access the System Definitions by either double clicking *System Information* in the Repository tab or selecting *System Information* from the *File* menu.



Specific Definitions

- 7.1.1 Export and Import
- 7.1.5 Execution and Security7.1.6 Display and Print controls
- 7.1.2 Task tracking

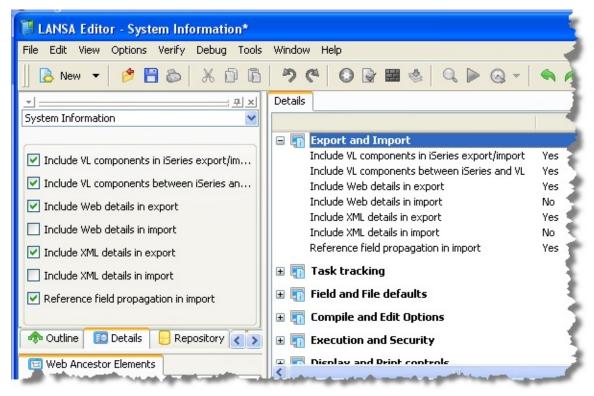
7.1.3 Field and File defaults 7.1

7.1.7 General Information

7.1.4 Compile and Edit Options

↑ 7. System Information

7.1.1 Export and Import



Include VL components in IBM i export/import

Controls how Visual LANSA component-related information is exported and imported between LANSA for the IBM i systems.

Include VL components between IBM i and VL

Allows Visual LANSA component-related information to be transferred between a LANSA for the iSeries system and Visual LANSA systems. The option affects the Export, Check out and Check in functions.

Include Web details in export

Allows the export of all Web details. This includes Web components as well as web details associated with fields, functions and system variables.

Include Web details in import

Allows the import of all Web details. This includes Web components as well as web details associated with fields, functions and system variables.

Include XML details in export

Allows the export of all XML details. This includes XML components as well as XML details associated with fields, functions and system variables.

Include XML details in import

Allows the import of all XML details. This includes XML components as well as XML details associated with fields, functions and system variables.

Reference field propagation in import*

If set to No, when an Import executes, Reference Field characteristics are not propagated to fields that reference those Reference Fields.

If a Reference Field is changed subsequently, the changes are propagated to the fields that reference it, as usual.

Note : *IMPREFFLDNOPROP is also used by the Host Monitor and LANSA Import to decide if reference field changes should be propagated. Prior to V9.1 no updates were performed during these operations and "no update" is the initial default setting.

This option now offers the choice of propagating changes or not.

The use of this option is not recommended as fields may become out of sync with their nominated reference field and therefore it should be removed from DC@OSVEROP and set to 'N' in Visual LANSA.

When this option is set to Yes, the input and output attributes to a field which references another field are not protected. This was was the default before V9.1.

It should be remembered that when a field which is referenced by other fields is changed using the field maintenance options or the PUT_FIELD Built-In Function, all the referencing fields are also updated.

↑ 7.1 System Definitions

7.1.2 Task tracking



Lock function to same task as process

Enables the task tracking logic that enforces that all functions are locked with the same Task Id as the parent process. This option only applies when the Task Tracking option: *Allow user to change tasks while working?* is set to No. For further details, refer to Task Tracking Settings in the *iSeries User Guide*.

↑ 7.1 System Definitions

7.1.3 Field and File defaults

👅 LANSA Editor - System Informatio	n* l
File Edit View Options Verify Debug	Tools Window Help
📙 🔁 New 🔻 🛛 🤌 🔚 🦝 🛛 🔏 🖞	〕
<u>▼</u>]	Details
System Information 🛛 👻	
Field defaults	🗉 🌆 Export and Import
Input attributes (alpha) FE	🖃 🛐 Task tracking
Input attributes (numeric) FE RB	Lock functions to same task as pro No
Output attributes (alpha)	🖃 🛐 Field and File defaults
	Field defaults
Output attributes (numeric)	Input attributes (alpha) FE
File defaults	Input attributes (numeric) FE RB Output attributes (alpha)
File SIZE parameter 10000 20	Output attributes (apria)
	File defaults
File LVLCHK parameter *YES	File SIZE parameter 10000 2000 3
File commitment control	File LVLCHK parameter *YES
	File commitment control No
no Outline 🔝 Details 🗧 Repos 🔇 🔊	🕀 🔄 Compile and Edit Options
Web Ancestor Flemmats	line and the second

Field defaults

Field input attributes are ONLY used in an IBM i RDML environment. All RDMLX and Visual LANSA RDML field attribute settings are set at the Partition Level. Refer to Field Type enabled in Partition for more information.

New Field Attributes Concepts

These field input attributes are ONLY used in an IBM i RDML environment. Input attributes are optional and define how a field is displayed when it is used as an input field in a function or on a form. Most input attributes, such as color, are used in functions because Visual LANSA forms define how fields are displayed using visual styles. Input attributes such as Hidden Field (for passwords) or Lowercase (allow lower case text to be entered) are used in both. If no input attribute is selected, it defaults to either the alphanumeric (type A) or

numeric (type P or S) system default values.

These are the valid input field attributes by field type:

Attribute Description / comments

Alpha Packed Signed

AB	Allow to be blank.	Y	Y	Y
ME	Mandatory entry check required.	Y	Y	Y
MF	Mandatory fill check required.	Y	Y	Y
M10	Modulus 10 check required.	Ν	Y	Y
M11	Modulus 11 check required.	Ν	Y	Y
VN	Valid name check required.	Y	Ν	Ν
FE	Field exit key required.	Y	Y	Y
LANSA Client	Lowercase entry allowed. Refer also to Locale upper casing required.	Y	Ν	Ν
RB	Right adjust and blank fill.	Y	Y	Y
RZ	Right adjust and zero fill.	Y	Y	Y
RL	Move cursor right to left.	Y	Y	Y
RLTB	Tab cursor right/left top/bottom	Y	Y	Y

Input attributes (alpha)

Specify the default input attributes to be assigned to a new alpha field.

Input attributes (numeric)

Specify the default input attributes to be assigned to a new numeric field.

Output attributes (alpha)

Specify the default output attributes to be assigned to a new alpha field.

Output attributes (numeric)

Specify the default output attributes to be assigned to a new numeric field.

File defaults

These default are used when LANSA files are initially created.

File SIZE parameter

This default is used when initially creating LANSA files. The initial number of records, the increment number of records and the maximum increments may be

specified. The parameter must be a valid OS/400 size parameter.

File LVLCHK parameter

This default level check, LVLCHK, parameter is used when initially creating LANSA files.

File commitment control

This default commitment control value is used when initially creating LANSA files. It specifies whether or not the file is to be placed under commitment control. Refer to Commitment Control for information.

Always Build using SQL (IBM i)

Set this option to Y to indicate that physical files and logical files on IBM i are to be built using SQL as much as possible when in an RDMLX partition. The physical file will be built as an SQL table. Logical files will still be built using DDS, but in such a way that their access path will implicitly share the access path of an SQL index where the i5/OS allows it. Refer to the *Always build using SQL* option in Field and File Defaults in the LANSA for iSeries User Guide.

↑ 7.1 System Definitions

7.1.4 Compile and Edit Options

📜 LANSA Editor - System Information			
File Edit View Options Verify Debug Tools Wind	dow H	lelp	
📙 🗟 New 👻 🤌 🛱 😂 🗮 🚳 🛛 🔏 👘 ಶ	9 (4		
<u>т</u> <u>д</u> х	Detai	ls	
System Information			<u> </u>
Process and function compile defaults	± Ę	Field and File defaults	
Web validate numerics		Compile and Edit Options	-
Duccess function and file compiles		Process and function compile defaults	
Process, function and file compiles		Web validate numerics	Yes
		Process, function and file compiles	
ILE bind RPG/IV modules into program		Compile using RPG/IV code	No
OTHER file I/O modules		ILE bind RPG/IV modules into program	No
Use *DATETIME conversion option		OTHER file I/O modules Use *DATETIME conversion option	Yes
Use *VARCHAR conversion option		Use *VARCHAR conversion option	No
No we della secondaria		I/O module compiles	
I/O module compiles ✓ I/O modules to have use adopted authority		I/O modules to have use adopted authority	Yes
		Support Client, SuperServer or Server	Yes
Support Client, SuperServer or Server		Support high speed record blocking in physical file	Yes
Support high speed record blocking in physical fil		Support high speed record blocking in RRN order Support LANSA/Server ODBC interface	Yes No
Support high speed record blocking in RRN order		Allow extended files to be added to high speed ta	
Support LANSA/Server ODBC interface		Suppress FATAL Crude Element Complexity Rating	
	 ⊕ ⊑	Execution and Security	
			-
Allow extended files to be added to high speed t		Display and Print controls	1
Suppress FATAL Crude Element Complexity Rati	± [General Information	
🔝 Details 🗧 Repository 🦙 Favorites 🔍 🔪	<		
		A second with the second	

Process and function compile defaults

Web validate numerics

Default = Yes.

Process, function and file compiles

Compile using RPG IV code. See Note 1.

Specifies that each program is to be compiled using RPG IV code, then bound as a single module ILE type program.

ILE bind RPG IV modules into program

See Note 1.

Activates the second level of ILE implementation. That will bind any GUI and multilingual program into the function program and use supplied service programs to dynamically call (CALLB) LANSA internal programs.

Must be used in conjunction with RPG IV. See Note 1

OTHER file I/O modules

Use *DATETIME conversion option

Indicates that the conversion option *DATETIME is to be used when OTHER file I/O modules are compiled. This allows date (L), time (T) and timestamp (Z) fields to be accessible in LANSA.

Use *VARCHAR conversion option

Indicates that the conversion option *VARCHAR is to be used when OTHER file I/O modules are compiled. This allows variable length (VARLEN or varchar) fields to be accessible in LANSA.

Note that this setting does not allow variable length character fields to be used as keys within LANSA. If the physical file or any logical views made known to LANSA have a varchar field as a key, the I/O module will fail to compile.

I/O module compiles

I/O modules to have use adopted authority

Indicates that all I/O modules created by LANSA are to have USEADPAUT(*NO). That is, do not use program adopted authority for I/O modules.

Support Client, SuperServer or Server

Corresponding to system flag *IOMXSERVER. See Note 2.

Use this option to indicate that I/O modules should be compiled to allow support of:

LANSA Client applications

and/or

LANSA Open applications using blocked I/O methods or the "receive immediate" option

and/or

Visual LANSA SuperServer applications.

It is recommended that you set the full list of values *IOMXSERVER, *IOMBLOCKBYKEY and *IOMBLOCKBYRRN into data area DC@OSVEROP as a system default for all LANSA systems.

Support high speed record blocking in physical file key order

Corresponding to system flag *IOMBLOCKBYKEY. See Note 2.

Use this option to indicate that I/O modules should be compiled to support high speed record blocking in physical file key order. You must use this option when using LANSA/Client. You must use this option when using the LANSA/ Server *BLOCKBYRRNnnnn selection option or the "receive immediate" option. You must use *IOMXSERVER if you use this option.

Support high speed record blocking in RRN order

Corresponding to system flag * IOMBLOCKBYRRN. See Note 2.

Use this option to indicate that I/O modules should be compiled to support high speed record blocking in relative record number order. You must use this option when using LANSA/Client. You must use this option when using the LANSA/Server *BLOCKBYRRNnnnn selection option or the "receive immediate" option. You must use *IOMXSERVER if you use this option.

Support LANSA Open ODBC interface

See Note 2

Use this option to indicate that I/O modules should be compiled to allow support of the LANSA Open ODBC Interface. You must use *IOMXSERVER if you use this option.

Allow extended files to be added to high speed tables

Corresponding to system flag *HSTABEXTEND.

Allows database files with record lengths up to 1988 bytes to be added to a user index for high speed lookup.

Warning: Refer to **Database File Attributes** in *Files* in the *iSeries User Guide* before using this option.

Warning: It is strongly recommended that if option *HSTABEXTEND is added to system data area DC@OSVEROP to make the extended entry record length available, or is removed to limit entry length, that all files tagged as high speed tables, all read only functions that use these files and all other I/O modules and Dboptimized functions that use high speed tables for lookup validation rules be recompiled AFTER deleting the current user index which is DC@TBLIDX if adding *HSTABEXTEND, DC@TBLIDY if removing *HSTABEXTEND.

Suppress FATAL Crude Element Complexity Rating in function

Indicates that a function that would cause the "Crude Element Complexity Rating" to return a FATAL will cause a WARNING only. It is not recommended to use this setting as the function may subsequently fail to compile.

Note 1

RPGIV and ILE: Before you attempt to use any of the RPGIV and ILE related switches, it is strongly recommended that you first read ILE Implementation in the *iSeries User Guide*.

If you have grossly exceeded the recommended limits for the number of logical views created (or made known to LANSA) then you may find that an existing I/O module may not (re)compile when these options are used.

If you have grossly exceeded the recommended number of real or virtual fields in a physical file then you may find that an existing I/O module may not (re)compile after these options are used.

In either case, temporarily remove the options from data area DC@OSVEROP while recompiling the I/O module that is experiencing the problem.

Also note that RPG IV (V3 version of RPG from IBM) has removed the total file, total static initialized storage and total subroutine limits that may be causing such problems to occur.

↑ 7.1.4 Compile and Edit Options

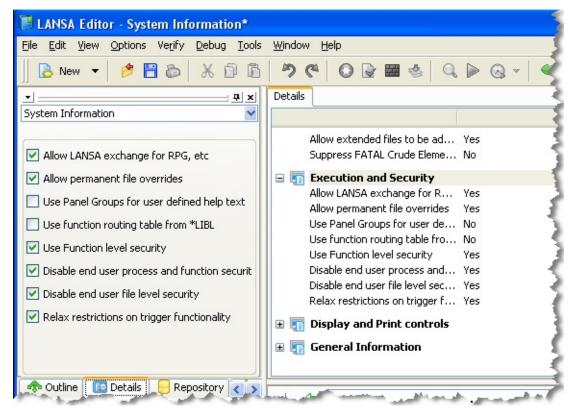
Note 2

*IOMXSERVER, *IOMBLOCKBYKEY, *IOMBLOCKBYRRN and *ODBC increase the number of files declared in an I/O module, the amount of static (literal initialized) storage used by an I/O module and the number of subroutines in an I/O module. Note also that if either *IOMBLOCKBYKEY, *IOMBLOCKBYRRN or *ODBC are specified, *IOMXSERVER must also be specified.

↑ 7.1.4 Compile and Edit Options

↑ 7.1 System Definitions

7.1.5 Execution and Security



Allow LANSA exchange for RPG, etc,

For IBM i use only.

Specify that the exchange list capability is required. This will allow the EXCHANGE list capability in RPG, CL, COBOL, etc programs so that values can be returned from LANSA to these programs using the EXCHANGE command. Values can also be put on the LANSA exchange list from RPG, CL, COBOL, etc programs. Refer to the EXCHANGE command in the *Technical Reference Guide* for more details. Recompilation is necessary for changes to this setting to take effect.

Allow permanent file overrides

For IBM i use only.

Allows permanent file overrides to be used. When you specify permanent file overrides you are telling LANSA that "every time I use this file, I really want to use this other file". This is useful when you want to use files with 10 character file names or files with a "." in their name, and so on. Refer to the The Permanent File Overrides Facility in the *iSeries User Guide* for details.

Use Panel Groups for user defined help text

Only relevant in an IBM i environment.

Indicates that the IBM's Panel Groups are to be used for the presentation of user defined help text, rather than the LANSA help text display facility. *Popup window to use OS400 window facility* in 7.1.6 Display and Print controls must also be selected.

Use function routing table from *LIBL

Controls the use of Function Routing table X_FUNRTR from *LIBL. If this is not set, the Function Routing table in the partition module library will be used. Refer to What is Function Routing? in the *iSeries User Guide* for details.

Use Function level security

Specify that the system uses function level security. Note that using function level security increases overall system resource usage. Processes (NOT functions) compiled prior to a change to this value will ignore the change and should be recompiled.

Disable end user process and function security

Yes specifies that end user process and function level security is disabled in this system. Disabling process and function level security improves system performance because no security checking is performed before accessing end user applications.

This option has no effect on access to process or function definition details in a development environment. Yes is appropriate for installations that use an external menu system to control and secure access to LANSA applications. Additional security checking within the LANSA application is a waste of resource and may mean "double updating" of end user access rights. The option's setting is interpreted dynamically by all applications.

Disable end user file level security

Yes specifies that end user file level security is disabled in this system. Disabling file level security improves system performance, because no file level security checking is performed in end user applications.

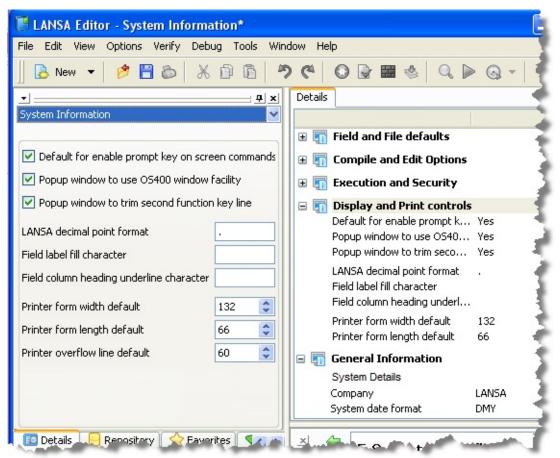
This option has no effect on access to file definition details in a development environment. **Yes** is appropriate for installations that use the approach "if the program is accessible from your menu then you can access all the files the program requires". The setting of this option is interpreted dynamically by all applications.

Relax restrictions on trigger functionality

Indicates that the normal LANSA rules that prevent database event triggers from calling other functions and using "user interface" commands should be relaxed. This use of this option is **not** recommended in most circumstances.

↑ 7.1 System Definitions

7.1.6 Display and Print controls



Prompt Key Enabled

Specify the default enabling of the PROMPT_KEY parameter on DISPLAY, REQUEST and POP_UP RDML commands.

When the prompt key is not enabled, the RDML command parameters must be used to specifically enable it.

Popup window to use OS400 window facility

Only relevant in an IBM i environment.

Yes indicates that pop-up windows created by LANSA should use the i5/OS windowing facilities.

Popup window to trim second function key line

Only relevant in a IBM i i environment.

Yes indicates that an existing LANSA defined pop-up window has the 2nd function key line trimmed from the display. Only set this option to Yes if the

Popup window to use OS400 window facility is Yes. This option enables existing functions that have been specifically sized onto line 24 of the display device to recompile without change.

LANSA Decimal format ("." Or ",")

Mandatory. Default value is ".".

Specify the character to be used for the decimal point.

Field label fill character

Specify the character to be used to fill the remaining space of the field label.

Field column heading underline character

Specify the underline character to be used for column headings.

Default printer width (80 – 198)

Specify the default width for the printer to be used in a 5250 execution environment. If no printer width is specified the system will use this default value.

Default printer form length

Specify the default printer for length to be used in a 5250 execution environment. If no printer form length is specified the system will use this default value.

Default printer overflow line number

Specify the default printer overflow line number to be used in a 5250 execution environment. If no printer overflow line number is specified the system will use this default value.

↑ 7.1 System Definitions

7.1.7 General Information

🏋 LANSA Editor - Syst	em Information*			
File Edit View Options	Verify Debug Tools Wi	ndow H	elp	
]] 🔁 New 🔻 🛛 🤌 🖪	6 8 6 6 4	26	0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	> @ - •
×		Detail:	5	
System Information	*			
System details Company	LANSA		Printer form width default Printer form length default Printer overflow line default	132 66 60
System date format	DMY			
-LANSA Details			System details	
Product	LANSA		Company System date format	LANSA DMY
			LANSA Details	Dim .
Version	11.4		Product	LANSA
Minor Version Level	I1		Version	11.4
Owner	QOTHPRDOWN		Minor Version Level	11
Program Library	DC@PGMLIB		Owner	QOTHPRDOWN
			Program Library	DC@PGMLIB
Database Library	DC@DTALIB		Database Library	DC@DTALIB
				1
🔝 Details 🛛 📒 Repositor	ry 🙀 Favorites 🔍 🕻			
And in the local division of the local divis	milit de minutes	1	and the second	and the second

System Details

Company

Mandatory.

Specify the name of your organization. This will be used with the *COMPANY system variable.

Rules

• Maximum 30 characters.

System date format

Mandatory.

Specify the format of date to be used by the system.

LANSA Details

Name of product (LANSA)

Mandatory. This information should not be modified.

Indicates the name of the LANSA product.

Version

Mandatory. This information should not be modified.

Indicates the release level of the LANSA product.

Minor Version Level

Identifies the last PC installed on the system. This information should not be modified.

Name of LANSA system owner

Mandatory.

Specifies the user ID that is the owner of the LANSA system.

LANSA program library/path

Mandatory. This information should not be modified.

Indicates the name of the library in which compiled RDML programs associated with this partition are kept.

LANSA data Library / Database Name

Mandatory. This information should not be modified.

Indicates the name of the library / database name that is to be the default for new files created in this partition. Note that this is a default value only and does not restrict users of the partition from creating files in other libraries.

↑ 7.1.1 Export and Import

7.2 X_LANSA.PRO Maintenance

The X_LANSA.pro file contains X_RUN parameters to be permanently specified and applied to the LANSA environment it is associated with.

To review the X_LANSA.pro file associated with the current Visual LANSA environment, select the X_LANSA.pro command on the *System Information* context (right click) menu.

4 💐	System Information	
⊳	剩 Message Files	Open
Þ	Partitions	Find
⊳	Primitives	
4	Remote Systems	X_LANSA.pro

To review the X_LANSA.pro file associated with a remote system select the X_LANSA.pro command on the *Remote System*'s context menu.

4	8	Remote Systems		
⊳	3	Technology Servi	-	Open
⊳	\boxtimes	Templates		Find
Þ	2	Users		Find
Þ	G	Tasks		X_LANSA.pro

The X_LANSA.pro maintenance interface is primarily provided to allow confidential details to be encrypted in the file but it also allows for general review and maintenance of unencrypted information.

X_LANS	A.pro *		
Load X_LA	NSA.pro from C:\PROC	GRA~2\LANSA\P32MAS~1\X_WIN95\X_LANSA\x_lansa.pro	
		🦻 🗙 🔚	
Sequence*	X_RUN Parameter	X_RUN Value	Encrypt
100	DBII	P3213SP2	
200	DBUT	MSSQLS	
300	GUSR	QOTHPRDOWN	
400	PSWD	*****	
			-

Add

Add entries to the X_LANSA.pro file. Typically an X_RUN parameter and it's associated value is entered on a single line. The value may be selected to be encrypted when the file is saved, in which case the value must be entered twice for verification.

Update profile ent	у	_
Sequence	500 \$	
Encrypt		
X_RUN Parameter	PSWD	
K_RUN Value	****	
Confirm X_RUN Value	****	
		OK Cancel

To enter a comment, enter the text in the X_RUN Value preceeded by a semicolon.

Update profile er	try	×
Sequence	500 \$	
Encrypt		
(_RUN Parameter		
(_RUN Value	; my comments	
		OK

Modify

Modify the selected entries. Only unencrypted details can be modified. Any encrypted information must be deleted and reentered.

Encrypt

Encrypt the selected entries when the file is saved.

Create Trace Entries

A set of standard X_RUN details is added to the file to enable tracing in the selected system. The default values are set for highest level tracing and can be modified as required.

```
ITRO = Y
ITRL = 9
ITRM = 999999999
ITRC = ALL
```

Comment

Comment selected (unencrypted) lines.

Delete

Delete the selected lines.

Save

Save the X_LANSA.pro.

Note: Modifications to the X_LANSA.pro must be saved before they are applied to the file.

If X_LANSA.pro details are not encrypted the file can alternately be maintained using a standard text editor such as Notepad.

7.3 Partitions

A LANSA system partition is a means of "dividing up" or "partitioning" one LANSA system. Each partition is completely separate from other partitions, but partitions may share some common system objects such as System Variables. Each partition has its own development characteristics and each partition has its own logical repository. Partitions are maintained at the LANSA System level.

Generally, partitions should be used to:

- Separate large and independent classes of users from one another (for example, production users, acceptance testing users and developers).
- Separate completely independent application systems or software packages.

Generally, partitions should NOT be used to:

- Divide up applications or software packages unless they are (and will remain) completely independent from one another.
- Satisfy a developer's urge to start new applications with a "blank page". The resulting impacts of satisfying such an urge may be poor reuse of existing objects and unnecessary and avoidable maintenance and deployment issues.

When you create a partition, please note that:

- There is no supported inter-partition communication. Fields, files, forms, processes and functions defined in one partition are not normally accessible from another partition.
- Each partition has a separate repository. Having multiple development partitions violates the fundamental concept of a single shared development repository, and creates unnecessary and avoidable maintenance and deployment issues.
- Each partition has a separate security system. If you have too many partitions this can become a maintenance overhead.

All partition information is stored on a LANSA Master System. This means that in a Visual LANSA Slave system, partition information is displayed as read only.

If you are using an Independent Visual LANSA System, partitions are maintained directly in Visual LANSA.

To access the Partition settings, in the *Repository* tab of the Editor, open the *System Information* list, then select *Partitions* and then the partition you wish to look at.

You must be logged on to Visual LANSA as the Security Officer or Partition Security Officer to perform partition maintenance tasks.

The characteristics of a partition definition are grouped into the following categories:

7.3.2 Partition Definition	7.3.4 RDMLX Partition	7.3.7 Universal
Tab	Settings	interface options
7.3.3 Task Tracking	7.3.6 Available Field	7.3.8 Supported
Partition Settings	Types in Partition	Database Products

See also

7.4 Language Settings 7.5 Frameworks 7.6 Groups

If you are using Visual LANSA Slave System connected to a LANSA for iSeries Master System, refer to

- System Partition Definitions in the iSeries User Guide.
- Visual LANSA Slave Systems Administration.
- ↑ 7. System Information

7.3.1 RDML and RDMLX Partitions Concepts

When to use an RDMLX Partition When to use an RDML Partition

LANSA uses the partition characteristics to allow you to control the types of objects that are used to create an application.

An **RDML Partition** supports only LANSA RDML Object Types. All LANSA partitions prior to Version 11 are RDML Partitions.

An existing RDML Partition can be enabled to become an RDMLX Partition as described in 7.3.5 Enable Existing Partitions for Full RDMLX.

An **RDMLX Partition (**identified by the *Enable for full RDMLX* option) extends the concept of the RDML Partition so that it supports both LANSA RDML Object Types and LANSA RDMLX Object Types.

- Once a partition has been enabled for RDMLX, all LANSA object development must be completed using the Visual LANSA development environment.
- Once enabled, an RDMLX Partition cannot be changed back to an RDML Partition.
- You may use the Visual LANSA development environment with RDML Partitions.

If applicable, LANSA for iSeries on an IBM i can continue to be the master system. It is the central repository that stores all object definitions and will be used to maintain the LANSA System Definitions and LANSA Partition Definitions. In this case, all LANSA Partitions are maintained from the LANSA for iSeries Housekeeping Menu. (Refer to LANSA Housekeeping in the *iSeries User Guide*.)

All IBM i objects will be exported to other IBM i servers using the LANSA for iSeries environment. Objects are not exported from the Visual LANSA development environment.

A **Multilingual Partition** is simply a partition that has multiple language support enabled.

Processes are not considered either RDML or RDMLX objects. Processes may contain a mix of RDML Functions and RDMLX Functions.

If you are starting new development with LANSA, it is generally recommended that you begin with an RDMLX Partition.

Following are some general recommendations regarding the creation of new

partitions:

When to use an RDMLX Partition

- If you are a new Version 11.0 (or later) LANSA customer, it is strongly recommended that you create all of your partitions as RDMLX enabled so that you can benefit from the full extended features of the LANSA development environment.
- If you are loading other files and require support for RDMLX Field Types, you will need an RDMLX Partition.
- If you are creating applications that require LANSA RDMLX Object Types such as WAMs, you must use an RDMLX Partition.
- Because all RDML objects can be created and used in an RDMLX Partition, most new partitions should be created as RDMLX Partitions.

When to use an RDML Partition

- If you are creating 5250-based applications that are solely developed with LANSA for iSeries (that is, you do not have Visual LANSA), you must use an RDML Partition for your development.
- If you are importing and maintaining a LANSA application created before Version 11.0, you may initially want to preserve the RDML Partition definition. Also, if the application must be exported to other RDML Partitions, you may wish to preserve the RDML Partition settings for simplicity.

17.3 Partitions

LANSA RDML Object Types

The following LANSA Repository objects are considered RDML objects:

- Alphanumeric, Packed and Signed Fields
- RDML Files using only Alphanumeric, Packed and Signed fields
- RDML Functions using only RDML Files, RDML Fields and RDML commands
- RDML Components using only RDML Files and RDML Fields.

RDML objects are fully supported in RDMLX Partitions. For example, in an RDMLX Partition, a File using only Alpha, Packed and Signed fields is still considered an RDML File.

Prior to Version 11, all LANSA objects (including components) are considered RDML object types. If you import objects from LANSA Systems Version 10 (or earlier), these objects will be imported as RDML Objects.

Also See

LANSA RDMLX Object Types 1 7.3.1 RDML and RDMLX Partitions Concepts

LANSA RDMLX Object Types

You may wish to enable LANSA RDML Object types to become LANSA RDMLX Object Types for the following reasons:

- To enable the object to interact with newly created RDMLX objects. For example, a Fully Enabled RDMLX Component may use all RDMLX Field Types and RDMLX Files.
- To enable the object to use the full RDMLX Language Features.

Note

- RDMLX Fields include all field types **except** alphanumeric. Refer to Field Types in the *LANSA Technical Reference* for a list and details of the field types available.
- RDMLX Files may use any type of field in the Repository.
- RDMLX Functions may use any type of field or file in the Repository. RDMLX Functions must use *DIRECT.
- An RDMLX Component is identified as being *Enabled for Full RDMLX*. A component must be *Enabled for Full RDMLX* in order to use other RDMLX objects and the RDMLX Language features.

RDMLX Language features include:

- use of intrinsic field methods
- use of function libraries
- use of expressions in many parameters and properties
- enhanced RDMLX command support such as assignment statements
- simplified RDML statements due to the removal of quotes.
- A WAM (Web Application Module) is always an RDMLX Component.

↑ 7.3.1 RDML and RDMLX Partitions Concepts

RDML and RDMLX Object Deployment

Applications developed with Visual LANSA can be deployed to other servers including Windows, Linux and IBM i. The method of deploying LANSA objects will depend upon the server that will host the application.

IBM i Deployment

The standard method for deploying RDML and RDMLX Objects to another IBM i server is to check the objects in to the LANSA for iSeries Master Repository. LANSA for iSeries import/export utilities are used to deploy objects to other IBM i servers.

Alternately, RDML and RDMLX objects can be delivered directly to a remote IBM i using the Deliver To command.

If you deploy an RDMLX Visual LANSA application using LANSA SuperServer to an IBM i, please note:

- If you already use an automatic connection to an IBM i using X_RUN, you should replace DBID=AS400 with DBID=*ANY.
- If you have used the Built-In Function DEFINE_OS_400_SERVER, in your source code you should replace it with DEFINE_ANY_SERVER.

Windows Deployment

To deploy applications to other Windows workstations or servers, use the Visual LANSA Deployment Tool. For details, refer to Introduction to Deployment Tool in the *Deployment Tool Guide*.

Linux Deployment

RDML and RDMLX objects can be delivered to a Linux runtime environment using the Deliver To command.

↑ 7.3.1 RDML and RDMLX Partitions Concepts

7.3.2 Partition Definition Tab

Partition Name Partition Description Unique Object Prefix Partition Security Officer Module library Default file library name Include in library list Partition is Multilingual Web Enabled SAA/CUA standard apply Enable Documentor User access enforced Perform checks before propagated Ignore propagated deletes *ENDWHERESQL

↑ 7.3 Partitions

Partition Name

A unique partition name must be entered. Specify the identifier to be assigned to the partition.

Rules

- Must be 3 characters long.
- Name must consist of characters A to Z, 0 to 9. Characters @, # and \$ are allowed, but are not recommended because of language translation issues.
- Name must be unique in the LANSA System. No two partitions can have the same identifier.

Also See

Unique Object Prefix ↑ 7.3.2 Partition Definition Tab

Partition Description

A partition description must be entered.

Enter a description that will assist developers to understand the purpose of the partition. This description is also used as the partition description for the default language.

Rules

- Cannot be blank.
- Maximum 40 characters.

Unique Object Prefix

A unique object prefix must be entered.

This prefix is used internally by LANSA within its database and externally in the name of the compiled programs it produces from RDML source statements. In Visual LANSA a list of unused prefixes is provided when a new partition is being defined.

Once saved the object prefix for a partition cannot be changed.

Rules

- The prefix must be alphabetic in the range D to Z, 0 to 9. Characters @, \$ and # are also allowed.
- The Object Prefix must be unique in the LANSA System. No two partitions can have the same prefix.

Partition Security Officer

A partition security officer must be entered.

Specify the name of the user profile to be used as the security officer for the partition. This user profile does not have any special rights in other partitions or outside of the LANSA system.

Rules

• The user profile must exist in the operating system.

Warning

• This user profile should not be used for development. Object security and task tracking rules will not apply to this user profile.

Module library

A module library must be entered.

Specify the name of the library in which compiled RDML programs associated with this partition are kept.

Rules

• The library must not be the same as the Module library used by any other partition.

Warnings

• Once specified the name of the module library cannot be changed.

Default file library name

A default file library name must be entered.

Specify the name of the library that is to be the default library for files created in this partition. This is a default value only and does not restrict users of the partition from creating files in other libraries.

Warnings

- Avoid using the library specified here as the IBM i "current library" in any interactive or batch job.
- When the default file library is changed, all Files that have a library equal to the previous default file library need to have JUST THE OAM re-built. This will retain existing data. If the table is re-built, existing data will be lost.

Also See

Module library

Include in library list

Specify that the **Default file library name** for this partition is to be automatically included in a LANSA job's library list when accessing this partition.

Platform Considerations

- IBM i: The Library list concept is used to control the hierarchy if libraries which are used by a job during its execution.
- 1 7.3.2 Partition Definition Tab

Partition is Multilingual

Indicate if this partition is intended to use more than one language.

In a multilingual partition, a set of languages can be defined which are then available for use in the partition. Partition multilingual attributes must be defined for each language including: the language code and description, the type of language support required (if DBCS or RLTB), and the translations for menu options and functions key descriptions.

In Visual LANSA a partition is automatically defined as multilingual with a default language created for the partition.

In an IBM i the partition must modified to be multilingual and an appropriate default language (and any other languages) must be defined before the partition can be used.

If you select multilingual support, you must also select the SAA/CUA standard apply option.

It is recommended to always define partitions as multilingual.

Rules

- You must use Multilingual support if any of the following conditions apply:
- The application uses a bi-directional language such as Hebrew or Arabic.
- The application uses ideographic characters (or double byte character set characters) such as Chinese, Japanese or Korean.
- The application executes in more than one language i.e. the same application will appear in more than one language.

Tips & Techniques

There are some important things that you should know about multilingual support before you attempt to turn it on (or off):

- Providing (or not providing) multilingual support in your applications is an important application design decision that you should make before starting development. Changing your mind late in the development cycle may lead to unnecessary and avoidable maintenance and deployment issues.
- There are important procedures, considerations and guidelines that you should understand and then follow when converting a partition from monolingual to multilingual (or vice versa). Please refer to the Multilingual Application Design Guide BEFORE changing a partition to be multilingual.
- 1 7.3.2 Partition Definition Tab

Web Enabled

This option is read only and indicates if the partition is already web enabled. If not, you can change it. Refer to Enable for the Web for more information.

SAA/CUA standard apply

Mandatory.

Specify that SAA/CUA (System Application Architecture/Common User Access) standards should applied in this partition.

Rules

• Visual LANSA partitions must be defined as SAA/CUA.

Also See

7.3.2 Partition Definition Tab

SAA/CUA Implementation in the Application Design Guide.

Enable Documentor

Specify if the LANSA Documentor is to be enabled in this partition.

LANSA Documentor is an IBM i based documentation tool to enable you to generate technical and user documentation for LANSA based applications.

Rules

• This option may only be used with LANSA for iSeries Systems.

Also See

Documentor Services in the *iSeries User Guide*.

User access enforced

Specify that LANSA user security is to be enabled in this partition. User authority will be checked to confirm that users have rights to perform specific development actions as described in LANSA User and Security.

Tips & Techniques

- This option is used to control user security in a similar fashion used in the LANSA for iSeries development environment.
- If you are using a LANSA for iSeries Master System, you must update your Visual LANSA System definition after making changes to security settings on IBM i. Refer to Visual LANSA Slave System Administration.

Perform checks before propagated

Specify that database integrity checks will be performed before Repository Synchronization propagates changes to the other databases. For more details, refer to Repository Synchronization.

Rules

• This option may only be used with LANSA for iSeries acting as the master repository.

Also See

Ignore propagated deletes ↑ 7.3.2 Partition Definition Tab

Ignore propagated deletes

Specify that any deleted objects notifications propagated to this workstation by Repository Synchronization are to be ignored by this repository. For more details, refer to Repository Synchronization.

Rules

• This option may only be used with LANSA for iSeries Systems.

Also See

Perform checks before propagated

*ENDWHERESQL

This option is used to enhance the performance of the SELECT commands on different tables (that is, SQL and native I/O) because SQL, using ODBC, doesn't handle table operations the same way as native I/O on System i. This option supersedes the SELECT command's option: *ENDWHERE.

This option allows developers to handle specific SELECT commands in the most appropriate manner according to the WHERE condition. It signals the LANSA system to interpret ALL *ENDWHERE options in SELECT commands as though *ENDWHERESQL had been coded. It is recommended that, as code is updated or new code is written, that the SELECT commands are changed to use this new option where it is appropriate.

7.3.3 Task Tracking Partition Settings

Enable task tracking Task is required Confirm Task required Allow Task to be Changed Disable Task "work with" security Active Task Tracking on Imports 1 7.3 Partitions

Enable task tracking

If a partition is defined on an IBM i Master then Task Tracking must be enabled if required. Task Tracking is automatically enabled when a partition is defined in Visual LANSA.

This option indicates that task tracking is active in this partition. Object and task authority checks will be performed and all events that have taken place for work performed on objects will be recorded.

Also See

Using Task Tracking

↑ 7.3.3 Task Tracking Partition Settings

Task is required

This setting indicates that the user requires a task identifier before any work can be performed on a selected object.

Rules

- Task Tracking must be enabled.
- This option may only be used with LANSA for iSeries Systems.

Also See

Create, Review, Change or Delete Partition Definitions in *iSeries User Guide*.↑ 7.3.3 Task Tracking Partition Settings

Confirm Task required

This settings indicates that a confirm task identifier pop-up is required when work has completed on a selected object. The user will be prompted to confirm or change (if CHANGE function key is enabled) the task identifier to be allocated for work performed on the selected object.

Rules

- Task Tracking must be enabled.
- This option may only be used with LANSA for iSeries Systems.

Also See

Create, Review, Change or Delete Partition Definitions in *iSeries User Guide*. ↑ 7.3.3 Task Tracking Partition Settings

Allow Task to be Changed

This setting indicates that the user is allowed to change the task identifier that is allocated to the selected object on which work was performed.

Rules

- Task Tracking must be enabled.
- This option may only be used with LANSA for iSeries Systems.
- The user must be authorized to the task identifier.
- ↑ 7.3.3 Task Tracking Partition Settings

Disable Task "work with" security

This setting indicates that the security checks should be disabled within task administration.

Rules

- Task Tracking must be enabled.
- This option may only be used with LANSA for iSeries Systems.

Also See

Partition Definitions - Create, Change or Delete in *iSeries User Guide*.

1 7.3.3 Task Tracking Partition Settings

Active Task Tracking on Imports

This setting indicates that task tracking is active for import jobs.

Rules

- Task Tracking must be enabled.
- This option may only be used with LANSA for iSeries Systems.

Also See

Partition Definitions - Create, Change or Delete in *iSeries User Guide*.
↑ 7.3.3 Task Tracking Partition Settings

7.3.4 RDMLX Partition Settings

WARNING: If you have an existing LANSA partition containing significant development, you must FULLY understand the implications of setting this value.

Before enabling any RDML objects to become full RDMLX object types, you must have a very clear understanding of the rules and impacts of the RDMLX types.

The decision to enable objects should be supported by very clear business benefits in order to avoid unnecessary impacts to your existing applications.

Changing to RDMLX may change the performance characteristics of your applications, so you should properly evaluate this once the object changes have been made.

Also See

Enable partition for full RDMLX Create Component as RDMLX Create Fields as RDMLX Create Function as RDMLX

Create File as RDMLX Enable Short Char

↑ 7.3 Partitions

Enable partition for full RDMLX

This setting indicates the partition will be allowed to use all RDMLX features including all RDMLX object types (such as WAMs) and all RDMLX language enhancements.

If the partition on the server is RDMLX-enabled and you are going to use RDMLX server side components, functions or files within your application, you should select this option.

Select this option, for example, when your application might be using CALL_SERVER_FUNCTION to run an RDMLX function or to access an RDMLX file on the server.

If this options is selected, the connection to the server uses the Built-In Function DEFINE_ANY_SERVER. For the restrictions in using this Built-In Function, refer to DEFINE_ANY_SERVER in the *Technical Reference Guide*.

Tips & Techniques

• If you are creating a new LANSA partition and you wish to determine if this setting is appropriate, refer to *When to use* ... in the 7.3.1 RDML and RDMLX Partitions Concepts.

Rules

- Once a partition has been enabled for RDMLX:
 - it cannot revert back to being RDML.
 - all LANSA objects must be developed using the Visual LANSA development environment.

Also See

Create Fields as RDMLX Create File as RDMLX Create Component as RDMLX Create Function as RDMLX 1 7.3.4 RDMLX Partition Settings

Create Fields as RDMLX

Specify that the default for all newly created fields in the partition will be *Enabled for Full RDMLX*.

To use RDMLX with an existing field, open the field in the Visual LANSA Editor and choose the *Definition* tab. Select the *Enabled for RDMLX* option.

Rules

• This option can only be selected if the Partition is enabled for full RDMLX.

Tips & Techniques

• This option only affects newly created fields. When you edit an existing field in the partition, it will not be changed to be *Enabled for RDMLX*. You can change a Field's RDMLX settings using the Field Definition.

Create File as RDMLX

Specify that the default for all newly created files in the partition will be *Enabled for Full RDMLX*.

Rules

• This option can only be selected if the partition has been enabled for full RDMLX.

Tips & Techniques

• This option only affects newly created files. When you edit an existing file in the partition, it will not be changed to *Enabled to RDMLX*. You can change the File RDMLX settings using the File Attributes.

Create Component as RDMLX

Specify that the default for all newly created components in the partition will be *Enabled for Full RDMLX*.

Rules

• This option can only be selected if the partition has been enabled for full RDMLX.

Tips & Techniques

• This option only affects newly created components. When you edit an existing component in the partition, it will not be changed to be *Fully RDMLX Enabled*.

Create Function as RDMLX

Specify that the default for all newly created functions in the partition will be *Enabled for Full RDMLX*.

Rules

• This option can only be selected if the partition has been enabled for full RDMLX.

Tips & Techniques

- If you are creating a large number of functions for IBM i based online applications (i.e. 5250 workstations), you may not want to set this value.
- This option only affects newly created functions. When you edit an existing function in the partition, it will not be changed to be Fully RDMLX Enabled.

Enable Short Char

Select *Enable Level One* through to *Nine* as required.

Default is Disable.

This setting indicates whether special Short Char handling should be generated to improve the runtime performance of Visual LANSA applications using the Full RDMLX types of String and Char.

Changing a partition's *Short Char Level* can alter the structure of LANSA working lists that contain one or more fields of type string and/or char. This altered structure is not compatible between LANSA objects that share the working list unless all these objects are rebuilt. If you do change the *Short Char* level, you must rebuild all the files, functions, forms and reusable parts in the partition in order to avoid unpredictable behavior at runtime.

By default the Visual LANSA runtime maintains the current value of a String/Char field by dynamically allocating a piece of memory long enough to store the field's current value. This is an efficient memory management mechanism when the length of a field's current value is generally much less than the field's defined length. On the down side, this mechanism does incur a performance overhead in order to manage the allocation and de-allocation of the piece of memory. This overhead can impact the performance of large working lists that include String/Char fields.

In order to minimize this overhead Visual LANSA has the facility to generate Short Char handling for String/Char fields. Short Char support is implemented by a single allocation of memory that can store the largest value allowed by the field's length. This feature saves the overhead of per value memory management, but this improved performance comes at the expense of a slightly larger memory allocation.

When the disabled setting is selected, Visual LANSA will treat all fields of type String and Char the same, irrespective of length.

To enable this setting, you select a *Short Char* level from one to nine. Each level corresponds to a multiple of 32 such that the level multiplied by 32 derives a Short Char length. All Visual LANSA fields of type String or Char whose length is less than or equal to the Short Char length will be implemented as a Short Char.

The most appropriate setting requires a judgment call that balances improved performance against increased memory usage. Further, the longer the String/Char field the greater the probability that much of the piece of memory allocated for the field's value will never be used. A reasonable balance can be achieved using a level around 2 to 4.

7.3.5 Enable Existing Partitions for Full RDMLX

All LANSA partitions prior to Version 11 are RDML Partitions. Once a LANSA System has been upgraded to Version 11 or greater, a partition can be enabled for RDMLX and then it becomes an RDMLX Partition.

Before you enable an existing partition for RDMLX, you should fully understand the potential development and business implications of this change. Specifically, you should review:

- When to use an RDMLX Partition in 7.3.1 RDML and RDMLX Partitions Concepts
- LANSA RDMLX Object Types

If you have a LANSA for iSeries Master System, the procedure for enabling an existing RDML Partition for Full RDMLX may involve steps on both the LANSA for iSeries Master System and steps on each workstation or server that contains a Visual LANSA Repository. To enable an existing partition, complete the following tasks as required:

- Update LANSA for iSeries Partition Definition
- Refresh Visual LANSA Systems
- Initialize LANSA for iSeries Partition

Once the partition has been enabled, remember that all development must be done from the Visual LANSA development environment.

If you have an Independent Visual LANSA System, the partition can be enabled for RDMLX in Visual LANSA as required.

If you enable a partition for RDMLX, and if you have existing SuperServer applications that are currently using Built-In Function DEFINE_OS_400_SERVER, you should change to Built-In Function DEFINE_ANY_SERVER.

↑ 7.3 Partitions

Update LANSA for iSeries Partition Definition

To begin, you must change the existing partition definition and then define the RDMLX features enabled for the partition.

Using the LANSA for iSeries Master System, perform the following steps:

- 1. Logon as the System Administrator or Partition Owner for the LANSA for iSeries partition to be RDMLX enabled.
- 2. Go to the LANSA Housekeeping Menu and select the Partition Maintenance option.
- 3. Select the existing partition to be RDMLX enabled.
- 4. Set the "Enable partition for full RDMLX" flag to YES.
- 5. Specify the RDMLX features to be enabled. (Refer to 7.3.4 RDMLX Partition Settings.)
 - a. It is recommended that **Create functions as RDMLX** be set to **NO** because you have many existing objects that are RDML objects.
 - b. It is recommended that **Create components as RDMLX** be set to **YES** to allow all new features to be enabled.
 - c. It is recommended that Alpha, Packed and Signed field types always be allowed.
- 6. Save the partition definition.

Next, complete the Refresh Visual LANSA Systems task.

 \uparrow 7.3.5 Enable Existing Partitions for Full RDMLX

Refresh Visual LANSA Systems

On each Visual LANSA Workstation or Server that contains a repository with the newly enabled RDMLX partition, you must update the partition definition. Perform the following steps:

- 1. Start Visual LANSA.
- 2. When the Logon Parameters dialog appears, enter a user profile and password and select newly enabled RDMLX partition.
- 3. Press the System Initialization button.
- 4. When the System Initialization dialog appears, make the following selections:
 - a. Check connect to Master.
 - b. In the Partition Definition field, enter the 3 character partition identifier.
 - c. It is recommended that you also select the PC Users, PC Workstations and Current Task Lists.
- 5. Press Ok to initialize the system.

Next, complete the Initialize LANSA for iSeries Partition task.

↑ 7.3.5 Enable Existing Partitions for Full RDMLX

Initialize LANSA for iSeries Partition

Finally, the original LANSA for iSeries partition must be initialized in order to support the execution of RDMLX objects on the IBM i. You will use a Visual LANSA System that is connected to the LANSA for iSeries Master System to perform the initialization.

Perform the following steps:

- 1. Start Visual LANSA and logon to the newly enabled RDMLX Partition. Ensure that the host monitor has been started.
- 2. Check out all files in the RDMLX Partition for read only (assuming all file changes have previously been checked into the Master Repository). If there are a large number of files, it is recommended that you use a LANSA for iSeries export and then import to Visual LANSA.
- 3. Check in all files and specify that the I/O Module is to be recompiled.
- ↑ 7.3.5 Enable Existing Partitions for Full RDMLX

Deploying RDMLX Applications to an IBM i

If you actually deploying an RDMLX Visual LANSA application using LANSA SuperServer to an IBM i, please note:

- If you already use an automatic connection to an IBM i using X_RUN, you should replace DBID=AS400 with DBID=*ANY.
- If you have used the Built-In Function DEFINE_OS_400_SERVER, in your source code you should replace it with DEFINE_ANY_SERVER.
- ↑ 7.3.5 Enable Existing Partitions for Full RDMLX

7.3.6 Available Field Types in Partition

Available Field Types can only be modified if the Enable partition for full RDMLX option has been selected.

Common and Specialized Field Types Field Type Enabled in Partition

↑ 7.3 Partitions

Common and Specialized Field Types

Follow these links for detailed field type information from the *Technical Reference Guide*:

Common types

The common field types include:

Alpha

Packed

Signed

DateTime

String

Alpha, Packed and Signed field types are always enabled.

When a partition is enabled for RDMLX all the Common Field Types are automatically enabled in the partition.

Specialized types

The specialized field types include:

Char

Binary

VarBinary

Date

Time

Integer

Float

CLOB

BLOB

Boolean

The Specialized Field Types are only available when the partition is RDMLX enabled. These field types must be manually enabled as required.

↑ 7.3.6 Available Field Types in Partition

Field Type Enabled in Partition

This setting indicates that the selected field type is available to be manually created by developers in this partition.

If a specific field type is not enabled, fields of this type may still be automatically created when an OTHER file is loaded.

Once you have enabled a field type to be created in the partition, you will be allowed to define the default field characteristics.

Refer to these field definitions in the *Technical Reference Guide*:

- Field Length
- Decimal Places
- Default Value
- Edit Code
- Keyboard Shift
- Input Attributes
- Output Attributes

Alpha, Packed and Signed field types are always enabled.

It is recommended that you do not change these defaults unless you are an expert user with knowledge of field types and databases.

Also review the detailed field type information for your selected Common and Specialized Field Types.

Tips & Techniques

- All fields types may be automatically created in the LANSA Repository when an OTHER file has been loaded. Refer to Load OTHER File in the *Developer Guide*.
- To determine if you should enable a field type, review all of the Field Type Considerations in the *Technical Reference Guide*.

↑ 7.3.6 Available Field Types in Partition

7.3.7 Universal interface options

The following options are used to control how RDML Function applications created using the Universal Model will be generated in the partition.

Function key lines
Message line number
Border fill - top corner
Border fill – top and bottom
Border fill - sides
Border fill - bottom corner

Also See

Creating Applications using Functions in the *Developer Guide*. **1** 7.3 Partitions

Process Menu style

This setting indicates whether the selection of menu items on a process menu should be by number or cursor. The default setting is Cursor.

Rules

Allowable values are:

Cursor The cursor position will be used to select an option from a process menu.

Number A number will be entered to select an option from a process menu.

Uppercase process title

This setting indicates that the process title should be in uppercase characters. **1** 7.3.7 Universal interface options

Uppercase function title

This setting indicates the function title should be in uppercase characters. **1** 7.3.7 Universal interface options

Display process date and time

This setting indicates that the date and time the process was created is to be displayed.

Display function date and time

This setting indicates that the date and time the function was created is to be displayed.

Display panel ID

This setting indicates that the panel identifiers are to be displayed in all functions.

Function key lines

Function key line 1 and Function key line 2 indicate the lines on the screen where the function keys should be displayed. The default settings are 23 and 24 respectively.

IBM SAA/CUA defaults specify that line 23 and 24 should be used.

Message line number

Specify the line on the screen where messages should be displayed. The default settings is 22.

IBM SAA/CUA defaults specify that line 22 should be used.

Border fill - top corner

This setting indicates the character to be used as the border at the top corner of the screen.

- Compiled pop-up window applications will use these fill characters at compile time, so a change to any value may mean that applications need to be recompiled before the change will be completely visible to end users.
- \uparrow 7.3.7 Universal interface options

Border fill – top and bottom

This setting indicates the character to be used as the border at the bottom corner of the screen.

- Compiled pop-up window applications will use these fill characters at compile time, so a change to any value may mean that applications need to be recompiled before the change will be completely visible to end users.
- ↑ 7.3.7 Universal interface options

Border fill - sides

This setting indicates the character to be used as the border at the left and right sides of the screen.

- Compiled pop-up window applications will use these fill characters at compile time, so a change to any value may mean that applications need to be recompiled before the change will be visible to end users.
- \uparrow 7.3.7 Universal interface options

Border fill - bottom corner

This setting indicates the character to be used as the border at the bottom corner of the screen.

- Compiled pop-up window applications will use these fill characters at compile time, so a change to any value may mean that applications need to be recompiled before the change will be visible to end users.
- ↑ 7.3.7 Universal interface options

7.3.8 Supported Database Products

Supported database products include:

Adaptive Server Anywhere

Microsoft SQL Server

ORACLE

IBM DB2 for IBM i

The set of Supported Database products is predetermined and cannot be modified.

The database products supported in a partition determine the limits and restrictions applied to database objects. For example, the field lengths supported vary by database type. Consequently, if you create a field any warning messages relating to the maximum allowable length of a field will relate directly to the supported database products.

For each supported database in a partition, the following information is supplied:

Database ID Database Description Database Enabled 17.3 Partitions

Database ID

This setting identifies the supported database. The Database ID is predetermined and cannot be modified.

↑ 7.3.8 Supported Database Products

Database Description

This setting describes the supported database. The Database Description is predetermined and cannot be modified.

1 7.3.8 Supported Database Products

Database Enabled

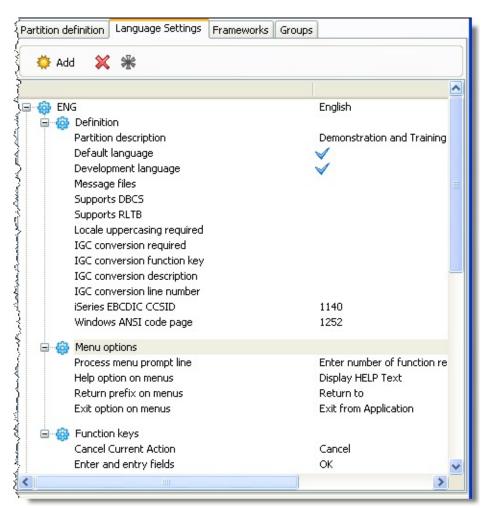
This setting indicates if the selected database is one of the databases supported in this partition. This is the only Supported Database Type setting which can be modified.

↑ 7.3.8 Supported Database Products

7.4 Language Settings

For each language defined in a partition, the following groups of information must be specified:

7.4.1 Language Definition 7.4.2 Menu Option Text 7.4.3 Function Keys



↑ 7. System Information

7.4.1 Language Definition

Partition Language Identifier

Partition language code Partition language description Partition description Character set code page IBM i EBCDIC CCSID Windows ANSI code page ISO language code

Language Settings

Default language Development language DBCS support required RLTB support required Locale uppercasing required

Ideographic Character Conversion

IGC conversion required IGC Function key IGC Conversion Description IGC Conversion Line number

Message Files

User Message files ↑ 7.4 Language Settings

Partition description

Mandatory.

Specify a description for this partition using the language being defined. This value defaults to the Partition Description and should be modified for the appropriate language based on this default description.

Rules

• Maximum length 40 characters.

Partition language code

Mandatory.

Specify a character code that uniquely identifies the language within the partition. It should be the same code used for the same language in other partitions, allowing language matching on import and export runs.

Rules

- Must be a 1 to 4 character code that uniquely identifies the language within the partition.
- If this is a base language, then it should be the same code as detailed in Specifying a Partition's Multilingual Attributes in the Multilingual Application Design Guide.

Partition language description

Mandatory.

Specify a full description of the language. This description is used on some reports as a description of the language being used.

Rules

- Maximum length 20 characters.
- 17.4.1 Language Definition

Character set code page

Specify the character set code page for this language. Character set code page is important for the correct translation of characters.

Default language

Indicates if this language is to be used as default language for the partition.

Rules

One language in the partition must be nominated as the default language. The default language is an important language and has the following implications:

- When an application is invoked without nominating the required language, the default language will be used.
- Initial data dictionary, database and application development must be done in the default language. For instance, if you specify a field description via the data modeling extended file definition facilities, it is assumed that the description is in the default language.

Development language

Specify if this language is a development language for this partition. Applications will be developed in LANSA with all development panels being displayed in the nominated language.

Warnings

• The ENG (English) language is a development language by default.

Tips & Techniques

• You should contact your product vendor before indicating if any language is to be used as a development language to confirm its availability for use in development.

DBCS support required

Specify whether or not the language requires DBCS (double byte character set or ideographic) support to successfully implement applications using this language.

For information about DBCS, search for "DBCS" in the Multilingual Application Design Guide.

RLTB support required

Specify whether or not the language requires right to left support (or bidirectional) to successfully implement applications using this language.

Bi-directional languages are supported in 5250 applications using server-side RDML but are not supported in Visual LANSA.

For information about RLTB, search for "RLTB" in the Multilingual Application Design Guide.

Locale uppercasing required

In a client/server configuration, this option is obtained from an IBM i server and is set during system initialisation on the PC.

On a Visual LANSA stand alone master system, it is set in the *Details* tab of the LANSA Editor while viewing the *Language Settings* for a partition.

The option is only used with fields that do not have the attribute LC (Lowercase entry allowed) specified. Unless LC is specified, a field's input values in the range **a** to **z** will be converted to upper case (that is **A** to **Z**).

This option is set to:

YES if characters outside the **a** to **z** range must also be converted to upper case, for example, from **à** to \hat{A}

NO, if special characters are to remain as lower case while those in the **a** to **z** range are converted to upper case. This could be the result: RESUMé.

For information regarding field attributes, refer to Input Attributes in the *Technical Reference Guide*.

IGC conversion required

Specify that ideographic character conversion (IGC) is required for applications generated under this language in this partition.

IGC Function key

Mandatory if IGC is required.

Specify a function key to be assigned to the IGC conversion task.

Rules

• Choose a function key number in the range of 1 to 24. The number should not conflict with any other CUA assigned function key in this partition.

Also See

IGC Conversion Description

IGC Conversion Line number

IGC Conversion Description

Mandatory if IGC is required.

Specify the short form description to be used for the IGC function key.

Tips & Techniques

If you are using a DBCS language you may find that you cannot specify the function key description in the DBCS language initially. To solve this problem:

- Define the DBCS language in full, but specify NO for the IGC conversion option initially.
- Exit from LANSA and sign off. Sign on again and then invoke LANSA. Proceed to this function again.
- Review the multilingual attributes of the DBCS language again. Use the change function key. This time you should be able to specify the function key description, and other details on this screen panel, correctly.

Also See

IGC Conversion Line number

IGC Conversion Line number

Mandatory if IGC is required.

Specify the line on the screen panel that the conversion area is to be placed.

Rules

• This option is only required if IGC conversion required has been selected and a IGC Function key has been specified.

Tips & Techniques

• It is recommended that the conversion area be placed on the same line as the message area. If your system is set up to show messages on line 22, it is advisable to also place the IGC conversion area on line 22.

Also See

IGC Conversion Description

IBM i EBCDIC CCSID

When text is moved from the client to the server and vice versa it must be translated from one system's code page to the other. Specify the CCSID to be used to translate text to the correct IBM I format for the selected language. If the code page is not entered then the default CCSID for the language will be used by the LANSA system.

Windows ANSI code page

When text is moved from the client to the server and vice versa it must be translated from one system's code page to the other. Specify the code page to be used to translate text to the correct Windows format for the selected language. If the code page is not entered then the default code page for the language will be used by the LANSA system.

ISO language code

The ISO 639-1 two-letter language code. It can be optionally qualified by an ISO 3166-1 two-letter country code. The language code or qualified languagecountry code must be a valid registered code, for example: en (English), en-US (English-United States) and en-GB (English-United Kingdom). This field is case sensitive. Values must be entered in the correct case (for example: **en-US**).

The ISO language code is commonly used in Web applications and when interfacing with external libraries that support localization.

User Message files

Specify up to 10 user defined message files for the partition language.

These files are used with override message file commands when entering applications defined within this partition. For more details, refer to Specifying a Partition's Multilingual Attributes in the *Multilingual Application Design Guide*.

Rules

• Message file "pairing" is mandatory. There must be the same number of User Message Files specified for each language as there is specified for the default language.

7.4.2 Menu Option Text

The menu option text allows you to control how LANSA builds and displays the default menus when a process is executed. You may control the following:

Process Menu Process Menu Prompt Line
Help Option on Menus
Return Prefix on Menus
Exit Option on Menus
↑ 7.4 Language Settings

Process Menu Prompt Line

The default value is 'Enter number of function required or place cursor on same line:'

Specify whether the prompt line option is to appear on process menus and what text should be displayed.

Rules

- If a Menu Prompt Line is required, enter the text that is to be displayed.
- Specify *NONE (in uppercase characters) to indicate that the prompt line is not required on process menus.

Tips & Techniques

- Change this text as required, particularly if you are running a system in a language other than English.
- You must recompile process menus after changing this option.
- 17.4.2 Menu Option Text

Help Option

The default value is 'Display process or function HELP text'.

Specify whether a 'Help' option is to appear on process menus.

Rules

- If a Help Option is required, enter the text that is to be displayed.
- Specify *NONE (in uppercase characters) to indicate that the help option is not required on process menus.

Tips & Techniques

- Change this text as required, particularly if you are running a system in a language other than English.
- You must recompile process menus after changing this option.
- If this option does not appear as a menu option, it does not prevent the user from using the help function key(s).

↑ 7.4.2 Menu Option Text

Return Prefix

The default value is 'Return to'.

Specify whether a 'Return to' option is to appear on process menus.

Rules

- If a Return To option is required, enter the text that is to be displayed.
- Specify *NONE (in uppercase characters) to indicate that the return option is not required on process menus.
- The Return To option will only appear on lower level menus as a top level menu has nowhere to return to.

Tips & Techniques

- Change the prefix as required, particularly if you are running a system in a language other than English.
- You must recompile process menus after changing this option.
- If this option does not appear as a menu option, it does not prevent the user from using the cancel function key.
- ↑ 7.4.2 Menu Option Text

Exit Option

The default value is 'Exit from system'.

Specify whether an 'Exit' option is to appear on process menus.

Rules

- If an Exit option is required, enter the text that is to be displayed.
- Specify *NONE (in uppercase characters) to indicate that the exit option is not required on process menus.

Tips & Techniques

- Change the text as required, particularly if you are running a system in a language other than English.
- You must recompile process menus after changing this option.
- If this option does not appear as a menu option, it does not prevent the user from using the exit function key.

↑ 7.4.2 Menu Option Text

7.4.3 Function Keys

The table below indicates the standard function key assignments used by the LANSA implementation under SAA/CUA, the suggested short form descriptions and the suggested function key that should be assigned.

You are not forced to follow these recommended values and can change them to your site standards as required.

Notes	Description Of Function Key	Short Form Description	Function Key No
Re	Cancel current action	Cancel	12
Re	Enter and entry fields	Enter / OK	RA
Re	Request help for the current function	Help	01
Ор	Display choices from a list of entries	Prompt	04
Op *	Restore or refresh panel	Refresh	05
***	Application defined messages key	Messages / Msgs	14
Op *	Scroll panel backward	Bkwd	07
Op *	Allow entry of a command	Command	09
Op *	Display cycle of function key area forms	Keys	13
Re	Exit to the highest level	Exit	03
Re	Exit to the next level above	Exit	03
Op *	Move cursor to first field on the screen	Home	HM
Op *	Scroll panel forward	Forward	08
Ор	Display table of contents for help	Contents	23
Op	Provide information about	Ex Help /	02

	entire panel	ExtHelp	
Op *	Provide index of help information *	Index	11
Op *	Provide names and functions of keys *	Keys help	09
Op *	Scroll panel left *	Left	19
Op *	Re-display last command that was issued *	Retrieve	09
Op *	Scroll panel right *	Right	20
Op *	Move cursor backward *	Switchbkwd	18
Op *	Move cursor forward *	Switch fwd / SwitchFwd	06
Op	Move cursor to action bar	Actions	10
***	Application defined add key	Add/Create / Create	06
***	Application defined change key	Change	21
***	Application defined delete key	Delete	22

Notes about this Table

- Re Indicates a SAA/CUA 'reserved' function key. In such cases the function key cannot be reassigned to another function key, even if the specified functions don't apply to a particular panel.
- Op Indicates a 'non-reserved' SAA/CUA function key. In such cases the function key may be reassigned to other functions, but only if the application does not support the functions proscribed for the function key by this table.
- * Indicates an SAA/CUA function key assignment that is proscribed for complete SAA/CUA compatibility, but is not currently implemented

directly within LANSA. The use of such keys is controlled by the USER_KEYS parameter of DISPLAY and REQUEST commands. It is the responsibility of the user to implement such key assignments in RDML programs as per the SAA/CUA guidelines described by the table above.

- *** Indicates a non-SAA/CUA proscribed function key that is required in this table because it is implemented by LANSA.
- ↑ 7.4 Language Settings

7.5 Frameworks

7.5.1 Identifier 1	7.5.4 Short name
7 E D Identifier D	

- 7.5.2 Identifier 27.5.5 Long name
- 7.5.3 System framework

Frameworks are a business-oriented grouping of items. For example all the components used to create the LANSA Personnel Demo application are in the Human Resources framework. Frameworks are used to establish a link between related components.

Some generic frameworks such as Manufacturing and Executive Information are automatically defined in a partition and contain components typically needed in these kinds of applications.

You can create your own frameworks.

When you create a component you can specify the framework in which you want to store it. You can also change the framework in the editor using the Framework property of the component.

You can view a list of objects in the repository using a framework name.

Frameworks do no have any programming or development impact. Frameworks are simply a method of associating objects to assist developers in managing their development environment.

There is no structural relationship between groups and frameworks, i.e. a group is not contained in a framework or owned by a framework.

1 7.5 Frameworks

7.5.1 Identifier 1

Mandatory.

This value will default to the serial number of the server on which the framework was created, with any letters removed.

Rules

- Value is numeric.
- Maximum of 5 digits.
- ↑ 7.5 Frameworks

7.5.2 Identifier 2

Mandatory.

Specifies a user assigned number to make the identity unique within the partition when combined with 7.5.1 Identifier 1.

Rules

- Value is numeric.
- Maximum of 2 digits.
- ↑ 7.5 Frameworks

7.5.3 System framework

Indicates this framework is a system framework.

Rules

• System frameworks cannot be deleted.

Tips & Techniques

• If you are creating a partition and copying system fields from a nominated partition, the system frameworks and groups in that partition will also be copied.

↑ 7.5 Frameworks

7.5.4 Short name

Mandatory.

Specify a short name that will be used to identify this framework for the developers. This name is used in the Repository tab and will be displayed in lists to identify the business framework.

Rules

- Must be entered in uppercase characters.
- No embedded blanks are allowed.

Also See

7.5.5 Long name

7.5.5 Long name

Mandatory.

Specify a long name that will be used to identify this framework for the developers. This is a more detailed description of the business framework.

Rules

- May be entered in mixed case.
- Descriptions may include embedded blanks.

Also See

7.5.4 Short name↑ 7.5 Frameworks

7.6 Groups

```
7.6.1 Identifier 1 7.6.3 System group7.6.2 Identifier 2 7.6.4 Short name7.6.5 Long name
```

Groups are a development-oriented means of grouping similar items together in the LANSA Repository. They do not have any programming or development impact but are simply a method of associating objects to assist you in managing your development environment.

Groups are similar to the frameworks, although frameworks are a businessoriented grouping of items. You can view a list of objects in the repository using a Group name.

- An object does not have to be associated with a Group when it is created.
- An object may be associated with a Group and with a Framework.

There is no structural relationship between groups and frameworks, i.e. a group is not contained in a framework or owned by a framework. Refer to 7.5 Frameworks.

1 7. System Information

7.6.1 Identifier 1

Mandatory.

This value will default to the serial number of the server on which the group was created, with any letters removed.

Rules

- Value is numeric.
- Maximum of 5 digits.

Also See

7.6.2 Identifier 2

↑ 7.6 Groups

7.6.2 Identifier 2

Mandatory.

Specify user assigned number to make the identity unique within the partition when combined with 7.6.1 Identifier 1.

Rules

- Value is numeric.
- Maximum of 2 digits.

Also See

7.6.1 Identifier 1

↑ 7.6 Groups

7.6.3 System group

Indicates that this is a system group.

Rules

• System groups cannot be deleted.

Tips & Techniques

- If you are creating a partition and copying system fields from a nominated partition, the system frameworks and groups in that partition will also be copied.
- ↑ 7.6 Groups

7.6.4 Short name

Mandatory.

Specify a short name that will be used to identify this group for the developers. This name is used in the Repository tab and will be displayed in lists to identify the group of developer objects.

Rules

- Must be entered in uppercase characters.
- No embedded blanks are allowed.

Also See

7.6.5 Long name ↑ 7.6 Groups

7.6.5 Long name

Mandatory.

Specify a long name that will be used to identify this group for the developers. This is a more detailed description of the group of developer objects.

Rules

- May be entered in mixed case.
- Description may include embedded blanks.

Also See

7.6.4 Short name

↑ 7.6 Groups

7.7 Primitives

All primitives that have been installed with the LANSA system can be viewed from the Visual LANSA Repository under System Information. This information is only available in Visual LANSA.

↑ 7. System Information

7.8 Remote Systems

This displays defined Remote Systems, including the associated master system if this is a Slave installation.

The Visual LANSA slave system automatically creates a master Remote System during the Visual LANSA installation. This master system definition cannot be changed.

Deployment Systems are also displayed. Before connecting to a Deployment System using Deliver To, the Remote System must be correctly set up. In the case of a Linux Remote System this includes using the Partition Initialization command to set up a corresponding partition on the Linux environment. IBM i Remote Systems must use the Refresh option to get the workstation names from the Remote System.

The *System Definition* tab displays the following details of the remote system:

Remote System Name	The name of the remote system. The remote system name should correspond with a Partner LU Name in the lroute.dat file.
Remote System Description	A description of the system.
Remote System Type	Indicates the type of system (Master or Deployment). This cannot be changed.
Build environment	IBM i or Linux.

Also see

Remote Systems

↑ 7. System Information

7.9 User and Security Maintenance

7.9.1 LANSA User Profiles
and Security7.9.3 Create a User
7.9.4 Maintain a User's7.9.5 Maintain a User's
Privileges7.9.2 Types of User ProfilesDefinition7.9.5 Maintain a User's

The LANSA Development Environment supports the concepts of user profiles, object owners and authorities to control the actions that can be performed by developers.

If you are using an Independent Visual LANSA System, user profiles and user access are maintained directly in Visual LANSA. Otherwise all user and security maintenance is performed on a LANSA Master System and is displayed as read only in the Visual LANSA slave system.

If you are using Visual LANSA Slave System connected to a LANSA for iSeries Master System, refer to User Access to the LANSA System in *the iSeries User Guide*. Also refer to Visual LANSA Slave Systems Administration.

Also see

- 7.11 Object Maintenance
- 1 7. System Information

7.9.1 LANSA User Profiles and Security

User profiles control security in the LANSA System. Each developer should have their own unique user profile. Each user profile is password protected and is used in the following ways:

- A developer logs on to LANSA (refer to Logon dialog) using their user profile.
- The user profile is authorized to Task IDs for Using Task Tracking in LANSA.
- The user profile has 7. System Information or privileges to determine actions that can be performed in the development environment.
- The user profile is used to identify ownership of new and existing Repository objects.

Refer also to the differences between IBM i and Visual LANSA authority, in Initial User Access to Objects in the LANSA for iSeries User Guide.

Developer Access

Each user profile is assigned access to objects, described in 7.11 Object Maintenance or privileges to determine the set of actions that can be performed in the development environment. For example, a user profile must be authorized to create or change a field in the repository before they can perform this action.

Object Maintenance

Authority to a specific object and its data is controlled using the 7.11 Object Maintenance. Each object has a list of authorities and users who can use or maintain the object.

Partition Settings

A partition level setting, User access enforced, determines if the security settings are used. The default is off, i.e. the user access security in Visual LANSA is not enforced. Once this setting is turned on, the security setting listed are enforced.

↑ 7.9 User and Security Maintenance

7.9.2 Types of User Profiles

There are many different user profiles required by the different software layers. For example, the Windows operating system, database management system, and Web Servers, may all requires user profiles that are registered as part of different software layers.

Windows Profiles

Visual LANSA software requires at least one Windows user profile (by default PCXUSER) with proper administrative authorities to set up the Windows environment to use the Visual LANSA software. The rules for Windows user profiles are dependent upon the operating system software and version.

Database Manager Profiles

By default the Visual LANSA installation includes an Adaptive Server Anywhere (ASA) database. The default database user is DBA with password SQL (uppercase). This profile is required to start the database service or connection to the installed Visual LANSA database.

Developer Profiles

As an Administrator, you are responsible for creating LANSA user profiles to allow developers to log on to the Visual LANSA development environment.

Visual LANSA user profiles can only be created using a Master LANSA System. Visual LANSA user profiles control the development operations available (add/change/delete operations) and user access rights to specific LANSA objects. Changes to a user's access rights do not take effect until the next time the user logs on to LANSA. If the user is currently using Visual LANSA, they should log off and then log back on to Visual LANSA to ensure that the changed object access rights take effect.

Note: Some options displayed in the editor are based on user authority. For example, system maintenance options are only displayed to authorized users; all users may view partition definition details, but only authorized users may change these details.

Existing Profiles

In an existing system where many user profiles may already be defined, LANSA provides an easy way to control the user access rights in Visual LANSA. The administrator may set the standard developer security access in the Master System.

If using a LANSA for iSeries Master system, you can Work with Administration Tasks and select the Authorize Visual LANSA developer option. Once updated on IBM i, the information must be refreshed or propagated to Visual LANSA. Refer to Maintaining Users and Tasks on Slave Systems.

If using a Visual LANSA Master System, you can create a User Profile to match the existing user profile and then define the privileges as described in 7.9.5 Maintain a User's Privileges for the user profile. Using this approach, a developer will not require multiple profiles.

↑ 7.9 User and Security Maintenance

7.9.3 Create a User

If using a Visual LANSA Master System, you can create a LANSA user as follows.

For a list of existing users, in the *Repository* tab, display the *System Information* contents and select *Users* from the list.

To create a user in Visual LANSA, either select *New* from the *File* menu and then choose User from the dropdown list or, alternatively, press the *New* button on the toolbar and choose *User* from the drop down list. The *New User* dialog will be displayed.

🔏 New User		×
User ID	NEWUSER	⊆reate
Password		Cancel
Confirm password		
Group profile	<u> </u>	Open in editor
		Close

For each new user you must enter:

User ID

Password

Confirm Password

and optionally:

Group Profile

Once all validation has been passed, the Create button will be enabled. Click it to create the user.

Note that if the Open in editor check box is left selected, the user will remain open for further editing. If the Close check box is left selected, the New User dialog will not remain open once the user has been created.

1 7.9 User and Security Maintenance

User ID

Mandatory.

Specify the name of the user who is to be allowed access to LANSA, or whose existing access rights to LANSA are to be reviewed or changed.

Rules

- Maximum of 10 characters.
- Must follow normal object naming conventions.

Tips & Techniques

- This user profile is being registered to Visual LANSA to allow a user to access the Visual LANSA development environment. This user profile does not have to exist anywhere else in the operating system.
- If you have an existing user profile, you can register it with Visual LANSA. Refer to 7.9 User and Security Maintenance.
- A valid user profile must be specified to start Visual LANSA. Refer to Logon Parameters.
- 17.9.3 Create a User

Password

Mandatory.

Set or reset the password for the user profile.

Rules

- Maximum 10 characters in length.
- Passwords are not case sensitive.

Tips & Techniques

• The password specified will be entered when starting Visual LANSA. Refer to Logon Parameters.

17.9.3 Create a User

Confirm Password

Mandatory.

Set or reset the password a second time to confirm the Password entered. Password characters are not displayed.

Rules

- Must match the value entered for the Password.
- Can be up to 128 characters long.
- Passwords are case sensitive.

↑ 7.9.3 Create a User

Group Profile

A Group profile is optional. Enter an existing user's name or select from a list of Users by clicking on the Ellipsis button. The group profile you select is the one from which you wish this user to inherit privileges.

Rules

- The group profile must already exist.
- *NONE is allowed.
- Can be left blank if desired.
- Users specified as being part of a Group cannot be used as a group to which to attach other other users.

1 7.9 User and Security Maintenance

7.9.4 Maintain a User's Definition

If you have an Independent Visual LANSA System installed, you can maintain LANSA user profiles within Visual LANSA.

For a list of existing users, in the *Repository* tab, display the *System Information* contents and select *Users* from the list.

If you are using a LANSA for iSeries Master System, user profiles will be created using the iSeries Master System and must be refreshed in the Visual LANSA System. Refer to Maintaining Users and Tasks on Slave Systems. The IBM i maintained profiles are displayed as read only in Visual LANSA.

📴 LANSA Editor - USER	4 (LANSA User)			1
File Edit View Options	Verify Web Debug	Tools Wir	ndow Help	1
📗 🔁 New 🔻 🛛 🤌 🖺	6 8 6 6	9 4	0 🛛 🖬 🖑 🔍	▷ 😪 🗉
_	·····································	Definitio	on Privileges	
USER4	-			
	517 - 57	Ξ 🍐	Details	-
User ID	USER4		Name	USER4
Password			Group profile	QSECOFR
Confirm a converse			Supplementary Groups	i i
Confirm password		- S	Kerberos Principal Names	1
Group profile	QSECOFR			
				1
👸 Find		×		1
🔏 User				
Like name	QSECOFR	_		
	🙌 Find			
Outl				
Name				
SECOFR				
•				
		el		
and the second s	and the second	J. Constant	Strength Str	

For existing users, the following details can be changed:

- Password
- Group Profile

- Supplementary Groups
- Kerberos Principal Names

The user must log off and on again, to be able to use the changes.

1 7.9 User and Security Maintenance

Supplementary Groups

Specify a supplementary group profile to which the User ID is to be attached. Enter the supplementary group by selecting the Supplementary Groups in the Definition tab and then entering the User profile of the secondary group in the *Details* tab. You can press the ellipsis button to obtain a list of current users to select from.

📕 LANSA Editor - USER2 (LANSA User*)				
File Edit View Options Verify Web Debug Tools Window Help				
🔁 New 👻 🔗 🔚 😂 🗶 🗈 🖻 🗳 🦃 🥙 🖓 🖼 🖉 🖉 🛸 🍕 🔍 🔍 🔍 🛸 🦂				
×	Definition Privileges			
USER2				
	🖃 🥈 Details			
Group profile	Name	USER2		
USER3	Group profile	QOTHPRDOWN		
	🖻 🄏 Supplementary Groups			
	S USER3	1		
	🔏 Kerberos Principal Names	3		
		1		
		1		
		1		
🖸 Details 🙀 Favorites 🖯 Repository 🕢 🕨				
and a set of the set of		and and		

Any profile used as a group cannot have groups of its own. Similarly, any profile having a group or groups cannot be a group to another user.

Rules

- The Group Profile must be specified.
- You can define up to 15 supplementary groups.
- ↑ 7.9.4 Maintain a User's Definition

Kerberos Principal Names

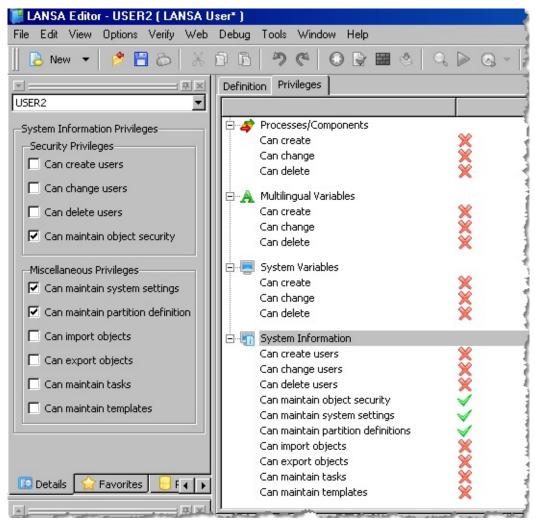
Displays the Kerberos principal names from the *Kerberos Network Authentication Protocol*. For information, refer to Single Sign-On (SSO).

↑ 7.9.4 Maintain a User's Definition

7.9.5 Maintain a User's Privileges

If using a Visual LANSA Master System, you can maintain the user's privileges from within Visual LANSA.

To view or maintain a user's privileges, choose the *User* from the list in the *Repository* tab, and then select the User's *Privileges* tab to see the editing functions allowed for the user. Pre-set privileges, identified by a gray tick, are inherited from the group, if any, to which the user has been assigned. To add privileges, select the object to be changed in the list in the *Privileges* tab and the settings to amend will be displayed in the *Details* tab. In the following example, the *Details* tab reflects that the user has selected the *System Information* entry:



The partition level setting User access enforced determines if the security settings are used. The default is off, i.e. the user access security in Visual LANSA is not enforced. Once this setting is turned on, the security settings

listed are enforced.

For each LANSA user profile, you may specify the following developer privileges within the LANSA environment:

Field Privileges

File Privileges

Function Privileges

Process and Component Privileges

Multilingual Variable Privileges

System Variable Privileges

System Information Privileges

Visual LANSA Specific Privileges

As you change settings, they will be immediately reflected in the Privileges tab.

1 7.9 User and Security Maintenance

Field Privileges

- Can create fields
- Can change fields
- Can delete fields
- 1 7.9.4 Maintain a User's Definition

File Privileges

- Can create files
- Can change files
- Can delete files
- Can load other file
- 1 7.9.4 Maintain a User's Definition

Function Privileges

- Can create functions
- Can change functions
- Can delete functions
- 1 7.9.4 Maintain a User's Definition

Process and Component Privileges

Specify whether the nominated user can:

- Can create processes / component
- Can change processes / component
- Can delete processes / component

1 7.9.4 Maintain a User's Definition

Multilingual Variable Privileges

Specify whether the nominated user can:

- Can create multilingual variables
- Can change multilingual variables
- Can delete multilingual variables

1 7.9.4 Maintain a User's Definition

System Variable Privileges

Specify whether the nominated user can:

- Can create system variables
- Can change system variables
- Can delete system variables

1 7.9.4 Maintain a User's Definition

System Information Privileges

- Can create users
- Can change users
- Can delete users
- Can maintain object security
- Can maintain system settings
- Can maintain partition definitions
- Can import objects
- Can export objects
- Can maintain tasks
- Can maintain templates
- 1 7.9.4 Maintain a User's Definition

Visual LANSA Specific Privileges

Specify whether the nominated user can:

• Can maintain remote systems.

This controls a user's access to all options associated with Delivering To a Deployment System – create, change and delete of Remote Systems, Refresh of a IBM i Deployment System, Partition Initialization on a Linux Deployment System and the availability of the Deliver To command against repository objects.

This set of privileges relates to functionality specific to the Visual LANSA development environment.

7.10 Task Maintenance

7.10.1 Task Identifier 7.10.3 Task Status7.10.2 Task Description 7.10.4 Authorized Users

Task maintenance is completed from a LANSA Master System only. If you are using an Independent Visual LANSA System, tasks are maintained directly in Visual LANSA.

You can review and edit the Tasks Definition from the *New* tab that creates a new Task or you can click on a Task to edit or add details.

If you are using Visual LANSA Slave System connected to a LANSA for iSeries Master System, refer to Working with Tasks in *the iSeries User Guide*. Also refer to Task Tracking in Master/Slave Systems and Visual LANSA Slave Systems Administration.

Also See

Change ManagementUsing Task Tracking in LANSA↑ 7. System Information

7.10.1 Task Identifier

A unique Task ID must be entered. Tasks are defined at the LANSA System level.

If Task Tracking is enabled in the partition, the Task ID is required by the Logon Parameters. The current Task ID is displayed on all the relevant LANSA windows in both Visual LANSA and on an IBM i Master system.

Rules

- Task ID is a maximum of 8 characters.
- The value entered is 'right blank' adjusted.
- If using Special Task IDs, the first two (left most) characters of the Task Identifier should conform to the codes in the Task Tracking System Settings (stored in positions 640-645 of data area DC@A07). Refer to Set Special Task ID.

Tips & Techniques

- It is strongly recommended that site naming standards are implemented for the naming of Task IDs.
- Task IDs can be used in any partition within the LANSA system, but it is recommended that a Task ID should be allocated for use in one partition only.
- If you have more than one LANSA System, it is very important to have a strategy for managing Task IDs between systems. For example, you may decide on a naming standard to identify the LANSA System or you may decide to create uniform Task IDs that are used for the same purpose in all LANSA Systems.

Also See

Tracked Objects ↑ 7.10 Task Maintenance

7.10.2 Task Description

A Task Description must be entered to clarify the Tasks usage. This description will appear on all lists and reports of Task IDs.

Rules

• A Task Description must be entered.

Tips & Techniques

• Inclusion of meaningful information in the Task Description makes the process of selecting the appropriate task to use much easier. For example, include details about the type of tasks created and any partition specific details.

Also See

Tracked Objects

↑ 7.10 Task Maintenance

7.10.3 Task Status

The Task Status is automatically set to Open (OPN) when a Task is created. When any work is logged against the Task the status is automatically updated from Open to Work (WRK).

The Task Status must be manually changed to Closed (CLS) when this Task ID no longer available to be used by a developer to modify objects. Closed Tasks retain a lock on any objects modified using this Task ID.

The final stage for a Task ID is to set it to Finished (FIN). This status must be manually assigned and effectively releases all objects which were locked to this Task ID.

Rules

Valid task status values are:

OPN	Task is Open. Authorized developer can start to use the task.
	•
	Task status can be manually changed.
	Task cannot be deleted.
WRK	Work has been performed using the Task ID.
	Objects are locked to this Task ID.
	Authorized developers can continue to use this Task Id to modify objects already locked to the Task ID or additional objects can be locked to the Task ID.
	Task status can be manually changed.
	Task cannot be deleted.
CLS	Task is Closed.
	Objects modified using this Task ID are still locked to the Task ID.
	Developers can no longer make changes to objects that are locked by this Task ID.
	Task and associated objects can be exported.
	Task status can be manually changed.
	Task can be deleted.
FIN	Task is Finished.
1	

The following is a summary table of the rules for tasks based on their status:

Status	Description	Change Status	Change Object	Object Locked	Export Task	Delete Task
OPN	Open	Y	Y	-	Ν	Ν
WRK	Work	Y	Y	Y	Ν	Ν
CLS	Closed	Y	Ν	Y	Y	Y
FIN	Finished	Ν	Ν	Ν	Ν	Y

Task status is commonly used as follows:

- A new task is created and its status is automatically set to OPN.
- As objects are locked to this task, the task status is automatically set to WRK.
- When work is deemed completed, the task status is manually set to CLS so no more changes can be performed. The locked objects can be included in an export.
- If any problems are identified during testing, the status can be manually reset to WRK to allow additional changes to be made to the objects.
- After the objects locked to the Task ID are released into production, the task status can be manually set to FIN. The objects are now unlocked and can be used by another task for another unit of work.

Also See

Tracked Objects Transferring Object Locks ↑ 7.10 Task Maintenance

7.10.4 Authorized Users

Specify one or more User Ids or Group User Profiles who are authorized to perform work using this Task ID. Refer to Choosing Your Task Tracking Approach.

Rules

- Up to ten valid user or group profiles can be assigned to a Task ID.
- If using Task Tracking by Developer, only a single user should be specified. Refer to Configuring Special Task IDs.

Tips & Techniques

- Only the Partition security officer and user or group profiles specifically associated with the Task ID can use the Task ID for development. Refer to Special Authorities and Task Tracking.
- Any changes to the list of authorized users on a Task ID will require the Task definitions to be refreshed on any slave systems. Refer to Maintaining Users and Tasks on Slave Systems.

Also See

Sharing Task IDs Special Authorities and Task Tracking

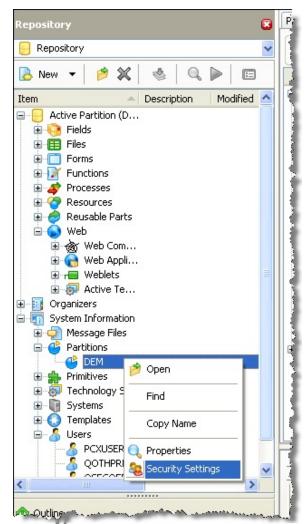
↑ 7.10 Task Maintenance

7.11 Object Maintenance

Object maintenance is used to control access to LANSA defined objects.

In Visual LANSA select an object, right click with your mouse, and choose *Security Settings* from the context menu.

The security settings will only be available in read-only mode if you are on a Slave system.



Security Settings - DEM	(Partition) - I	Demonstr 🔀
🔅 Add 🛛 💥		
Authorized Users	User	*PUBLIC
S *PUBLIC QOTHPRDOWN PCXUSER	Options Definition Use Modify Delete	
ŌK		Cancel

In an iSeries Master System use the *Review Access to Objects Defined within LANSA* option on the *Housekeeping Menu* to review and edit the object security details.



See also

7.11.1 Object Maintenance Concepts	7.11.2 Object name	7.11.4 Authority to Object Definition
	7.11.3 User Id	7.11.5 Authority to Object Data

For details about user access to the LANSA development environment, refer to

7.9 User and Security Maintenance.

1 7. System Information

7.11.1 Object Maintenance Concepts

The LANSA object security system can be used to control access to specific LANSA objects:

- Fields (and Forms, Reusable Parts and WAMs when relating to them on an IBM i)
- Files
- Processes
- Functions
- Partitions
- Application Templates
- Multilingual Text
- Weblets
- System Variables

Implementing security at an object level is optional and can be a significant administrative overhead.

Function level security should be used with care as this involves more run time security checking and consequently uses more system resources.

If using an IBM i Master, the *Use Function level security* option in the **Execution and Security Settings** window must be set to Y, indicate that function level security is required.

If using a Visual LANSA Master, right click on an object in the Repository to select the *Security Settings* for that object.

Access to all of objects is controlled by the *LANSA Object Maintenance* system.



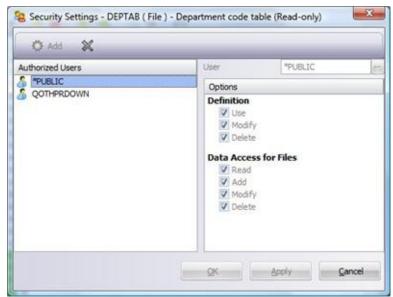
Within the LANSA security system there are 2 "classes" of access associated with most object types. These are:

DEFINITION

This class of access is applicable to fields, files, processes, functions, partitions, templates, system variables, weblets and multilingual variables. This access controls a users' right to USE, MODIFY and DELETE the definition of an object.

DATA Access for Files

This class of access is only



applicable to files. This access controls a users right to READ, ADD, MODIFY or DELETE information (records) contained in the file.

Refer to 7.1.5 Execution and Security for security setting information.

The two object classes (DEFINITION and DATA) and the way they affect the object types is summarized in the following table.

Object Type	Access Class	Description Of Access Allowable
FIELD COMPONENT WAM	DEF	USE: User can use the field definition. MODIFY: User can modify the field definition. DELETE: User can delete the field definition.
	DATA	Data rights are not applicable.
FILE	DEF	USE: User can use the file definition. MODIFY: User can modify the file definition. DELETE: User can delete the file definition.
	DATA	READ: User can read records from the file. ADD: User can add records to the file. CHANGE: User can change records in the file DELETE: User can delete records from the file.
PROCESS or FUNCTION	DEF	USE: User can use (run) the process/function. MODIFY: User can change the definition.

		DELETE: User can delete the definition.
	DATA	Data rights are not applicable.
PARTITION	DEF	USE: User can access the partition. MODIFY: User can change the partition definition. DELETE: User can delete the partition definition.
	DATA	Data rights are not applicable.
TEMPLATE	DEF	USE: User can use the template. MODIFY: User can change the template definition. DELETE: User can delete the template definition.
	DATA	Data rights are not applicable.
SYSTEM VARIABLE	DEF	USE: User can use the system variable. MODIFY: User can change the system variable. DELETE: User can delete the system variable.
	DATA	Data rights are not applicable.
MULTILINGUAL VARIABLE	DEF	USE: User can use the multilingual variable. MODIFY: User can change the multilingual variable. DELETE: User can delete the multilingual variable.
	DATA	Data rights are not applicable.
WEBLET	DEF	USE: User can use the weblet. MODIFY: User can change the

	weblet. DELETE: User can delete the weblet.
DATA	Data rights are not applicable.

Also See

7.9 User and Security Maintenance↑ 7.11 Object Maintenance

7.11.2 Object name

Display only.This is the name of the current object to be maintained.↑ 7.11 Object Maintenance

7.11.3 User Id

In Visual LANSA select the User ID to modify their rights to the current object. Select the New option to add a User ID to the object.

In an IBM i Master System maintain the list of User Ids against the selected object.

Rules

• User ID must be defined to LANSA.

↑ 7.11 Object Maintenance

7.11.4 Authority to Object Definition

Specify the access rights that the nominated 7.11.3 User Id has to the objects' definition.

Rules

• Allowable values are:

USE User can use the file definition.

MODIFY User can modify the file definition.

DELETE User can delete the file definition.

Tips & Techniques

• Changes to a user's access rights to a LANSA object do not take effect until the next time the user starts to use LANSA.

↑ 7.11 Object Maintenance

7.11.5 Authority to Object Data

Specify the access rights that the nominated 7.11.3 User Id has to the objects' data.

This option is only available when working with Files.

Rules

- Allowable values are:
 - READ User can read records from the file.
 - ADD User can add records to the file.

CHANGE User can change records in the file

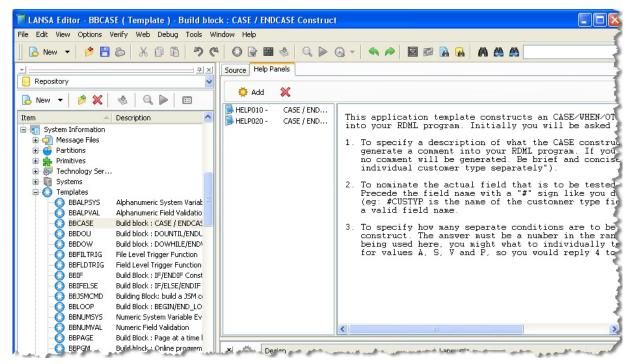
DELETE User can delete records from the file.

↑ 7.11 Object Maintenance

7.12 Application Templates

7.12.1 Template Name	7.12.3 Display sequence	7.12.5 Initial public
7.12.2 Template	7.12.4 Extended	access
Description	description	

In Visual LANSA you can review a list of all LANSA Application Templates in the Repository tab. To modify or review an Application Template definition, find the template from the Repository tab.



In an IBM i Master System use the Work With Application Templates Definitions from the Housekeeping Menu.

Also See

Application Templates in the Developer Guide.

1 7. System Information

7.12.1 Template Name

Mandatory.

Specify a unique identifier for this application template.

Application templates are shared across all partitions.

Rules

- Maximum of 10 characters.
- Identifier must be unique in the LANSA System.

Tips & Techniques

- Templates shipped with Visual LANSA are given the prefix "VL_" for their template identifier. This prefix allows the editor to display Visual LANSA specific templates. It is recommended not to use this prefix if you create you own templates.
- ↑ 7.12 Application Templates

7.12.2 Template Description

Mandatory.

Specify a short description of the application template. This will appear on all lists of templates.

Rules

- Maximum 40 characters.
- ↑ 7.12 Application Templates

7.12.3 Display sequence

Mandatory.

Specify the sequence number of the template. This number determines the sequence of templates displayed in the Template Maintenance dialog.

Rules

- Display Sequence must be greater than 0.
- ↑ 7.12 Application Templates

7.12.4 Extended description

Specify a more detailed description of the application template. Used for documentation purposes only.

Rules

- Maximum 40 characters.
- ↑ 7.12 Application Templates

7.12.5 Initial public access

Mandatory. The default value is ALL. Specify the access the public is to have to the template. This option is only displayed when a new template is created.

Rules

- Allowable values are:
 - All User has all rights to use, modify and delete the template definition.

Normal User has rights to use the template.

None User has no rights to the template definition.

↑ 7.12 Application Templates

8. Environment Settings

Visual LANSA allows you to control the behavior of the editor as well as development environment settings that are used when building applications. Review the following:

8.1 LANSA Editor Settings

8.2 Design View Language

8.3 Development Language

8.4 Object Locking

8.5 Workstation Locks

8.6 Language Options

8.7 Message File Maintenance

8.8 Export and Import

Also See

System Maintenance

8.1 LANSA Editor Settings

By using the LANSA *Settings*, developers can control what information is displayed and how it is displayed.

To set the LANSA Settings, open the *File* menu and choose *Editor Options* button at the bottom of the window.

IANSA Setti	ngs	×
General General Source Design Compile Compile Style	General Show Details Show Hint Show In Key Sequence Show Internal Files Show ReadOnly Message Show Short Names Assistant Auto Expand Parameters Details Show Advanced Features Outline Layout Items	QK Gancel Apply Reset Reset All Reset Editor

The Setting's options are described in details in LANSA Settings in the User Guide.

Font and color changes are selected from the status bar at the bottom of the LANSA Editor window:



Style Settings

General Source	QK Cancel Apply Reset
	Apply
Source	
Source	
Jource	ineset inteset
	D 1 41
	Reset All
Design	
	Reset Editor
WAM	
Debug	
🙏 🛛	
Compile	
Style	
Style	

↑ 8. Environment Settings

8.2 Design View Language

The Design View Language is set to the Current Language you used to log on to Visual LANSA.

The Design View Language determines what language you will see multilingual values displayed in (such as field labels) when you view the Design tab in a form or reusable part. Design View Language can be changed as required by using the LANSA Editor *Options* menu and selecting the Design View Language option.

Note: The Design View Language does not change the Language used to display descriptions and other information in the Repository tab. This information is based on the Current Language used to log on to Visual LANSA.

↑ 8. Environment Settings

8.3 Development Language

By using the LANSA Editor *Options* menu and selecting the Development Language, you can select the language used for all Visual LANSA dialogs, menus and messages the next time you log on.

You may select the development language from the list of available languages provided by the software vendor. The development language can only be one of the languages Visual LANSA is enabled for.

Note: The development language and the partition's design view language are not the same. The development language determines the language of the software development environment whereas the design view language specified for the partition determines the default language used for components in the Design tab.

Also See

- 8.6 Language Options
- 1 8. Environment Settings

8.4 Object Locking

By using the LANSA Editor *Options* menu and the Object Locking options, you can specify whether workstation locks will be applied as developers edit objects in the LANSA Repository. Object locking is defined at the repository level and not at the individual workstation level.

The object locking option is used to enable or disable Database-wide and Workstation-wide locks. For further information, refer to Object Locking Concepts.

Object locking can also be used to set all objects in the partition to a READ ONLY status so that no changes can be made by developers. Developers will be allowed to open objects in the editor in read only mode.

You can see a list of all objects locked by a specific workstation as described in 8.5 Workstation Locks.

If you have more than one developer sharing a single LANSA repository, it is strongly recommended that object locking is turned on.

Object locking is part of a complete Change Management strategy.

Also See

8.5 Workstation LocksObject Locking Concepts↑ 8. Environment Settings

8.5 Workstation Locks

Execute the Workstation Locks form, XVFLOCK, to manage any workstation locks on the Repository. You must have administrator authority to display and clear all workstation locks.

Node	TWA07		4					
User	Lock Type	Object Type	Process ID	Object Name	Extension	Partiti	Date	Time
TWA01	Shared No Update	Execute	4044	LVFLOCK		IDE	09/07/2012	16:37:57
TWA01	Shared For Read	Component	4044	LVFLOCK		IDE	09/07/2012	16:37:57
TWA01	Exclusive Allow Read	Field	5576	LEBIF010E		IDE	09/07/2012	16:19:06
TWA01	Exclusive Allow Read	Component	5576	LVFLOCK		IDE	09/07/2012	16:17:53
TWA01	Shared For Read	User	5576	KATE1		IDE	09/07/2012	16:17:04
TWA01	Shared For Read	Task	5576	*UKATE01		IDE	09/07/2012	16:17:04
TWA01	Shared For Read	Partition	5576	IDE		IDE	09/07/2012	16:17:04
WA01	Shared No Update	System	5576	IDETRUNK		IDE	09/07/2012	16:17:04
								Clear Locks

Select the desired workstation from the dropdown list to display the details of any locked objects for this workstation. Use the refresh button beside the dropdown to ensure you are viewing the latest details.

If you do not have administrator authority you will only see a list of the workstation locks related to your user profile. You cannot clear the workstation locks.

If you do have administrator authority you will see the details of all workstations and their related locks and will be able to clear the locks if the associated process is not active.

↑ 8. Environment Settings

8.6 Language Options

By using the LANSA Editor *View* menu and selecting Language Options, you can specify the options associated with translation from a particular language.

📜 Configure Language Options	×
Message File	
Message file name to be used in place of 'DC@M01':	
DC@M01	
Field Prefix Character	
Character to be used instead of the '#' character. Note that this character has to be set up to be translated to and from HEX '7B' on the iSeries	
#	
Character Character to be used instead of the '@' character.	
'\$' Character Character to be used instead of the '\$' character: \$	
OK Messages Cancel Help	

These options include:

- 8.6.1 Message File Name
- 8.6.2 Field Prefix Character
- 8.6.3 Substitution Character for the @ Character
- 8.6.4 Substitution Character for the \$ Character

These values should only be changed when absolutely necessary as the result of character translation issues.

1 8. Environment Settings

8.6.1 Message File Name

Mandatory.

Specify the message file to use in place of DC@M01. This file will then be used when searching for messages normally residing in the message file, DC@M01.

Note: Do not change this message file name unless you are absolutely sure that the message file name needs to be changed.

Also See

- 8.7 Message File Maintenance
- ↑ 8.6 Language Options

8.6.2 Field Prefix Character

Mandatory.

Specify the character to be used as the field prefix character in LANSA. This character is normally the # character.

For example, the Standard Name field in the repository is identified as #STD_NAME. The # symbol allows LANSA to identify this as a field name in RDML commands.

Note: Do not change this character unless you are absolutely sure that the field prefix character needs to be changed as the result of character translation issues. On the IBM i, this character must be set up to be translated to and from the HEX 7B character in the translation tables.

Also See

8.6.3 Substitution Character for the @ Character

8.6.4 Substitution Character for the \$ Character

↑ 8.6 Language Options

8.6.3 Substitution Character for the @ Character

Mandatory.

Specify the character that will be used as the substitution character for the *@* character in LANSA. This will be used when validating object names.

The @ character is often used in LANSA internal file and library naming standards.

Note: Do not change this character unless you are absolutely sure that it needs to be changed as the result of character translation issues.

Also See

8.6.2 Field Prefix Character

8.6.4 Substitution Character for the \$ Character

1 8.6 Language Options

8.6.4 Substitution Character for the \$ Character

Mandatory.

Specify the character that will be used as the substitution character for the \$ character. This will be used when validating object names.

The \$ character is used to identify special substitution values in LANSA.

Note: Do not change this character unless you are absolutely sure that it needs to be changed as the result of character translation issues.

Also See

8.6.2 Field Prefix Character8.6.3 Substitution Character for the @ Character

1 8.6 Language Options

8.7 Message File Maintenance

If you have proper authority, you can maintain the LANSA internal message file using the LANSA Editor *View* menu and selecting the *Message File* option. This message file can be used by both the development environment and your LANSA created applications. This dialog allows messages for the development language to be read from one message file at a time from the database. Only the Message identifier and a portion of the First Level Text is displayed.

📜 Message File Maintenance	×
Message Search Criteria 1st level text contains: Message file 2nd level text contains: DC@M01 Language: ID from: dcm to: Find English	
Message ID/Text No. of Messages: 4867	
DCM0001 Place cursor on same line as function required or en DCM0002 No data dictionary fields found with full or partial DCM0003 Choose required dictionary field(s) from the list or DCM0004 No file definitions found with full or partial name &1 DCM0005 Choose required file definition(s) from the list or DCM0006 No process found with full or partial name &1 DCM0007 Choose required data access module(s) from the list DCM0008 Field &1 could not be found in file DC@F03V2	
Close Add Change Delete Messages Help	

Search criteria based on following (all criteria are combined using AND logic):

- 8.7.1 1st level text contains
- 8.7.2 2nd level text contains
- 8.7.3 ID from
- 8.7.4 ID to

A new message can be added by selecting the *Add* button. (Refer to 8.7.5 Add/Change Message Text.)

An existing message can be changed by selecting the message in the list and then selecting the *Change* button.(Refer to 8.7.5 Add/Change Message Text.)

An existing message can be deleted by first selecting the message in the list and then selecting the *Delete* button.

Note: The changes made in the Message File Maintenance dialog are committed

to the database and the host as a result of the Change, Add and Delete buttons. Refer to the Application Design Guide before editing the message file.

Also See

8.7.5 Add/Change Message Text

↑ 8. Environment Settings

8.7.1 1st level text contains

Specify the first level text to be used in the searched. This field is case sensitive. Leaving this field blank will give a match for all messages.

All search criteria are combined using AND logic.

Also See

- 8.7.2 2nd level text contains
- 8.7.3 ID from
- 8.7.4 ID to
- 1 8.7 Message File Maintenance

8.7.2 2nd level text contains

Specify the second level text to be used in the searched. This field is case sensitive. Leaving this field blank will give a match for all messages. All search criteria are combined using AND logic.

Also See

- 8.7.1 1st level text contains8.7.3 ID from
- 8.7.4 ID to
- 1 8.7 Message File Maintenance

8.7.3 ID from

Specify the lowest alphanumeric value that will be searched when matching message IDs. This field is case sensitive. Leaving this field blank will start the search at an ID of AAAAAAA.

This field is used in combination with the 8.7.4 ID to field to define a range of message IDs.

All search criteria are combined using AND logic.

Also See

8.7.1 1st level text contains

8.7.2 2nd level text contains

8.7.4 ID to

1 8.7 Message File Maintenance

8.7.4 ID to

Specify the highest alphanumeric value that will be searched when matching message IDs. This field is case sensitive. Leaving this field blank will end the search at an ID of zzzzzz .

This field is used in combination with the 8.7.3 ID from field to define a range of message IDs.

All search criteria are combined using AND logic.

Also See

8.7.1 1st level text contains

8.7.2 2nd level text contains

8.7.3 ID from

1 8.7 Message File Maintenance

8.7.5 Add/Change Message Text

Msg file: Message ID:	DC@M01 PRR0077	Language: English
1st. level text:	Communication	is error on receive from server with symbo
2nd. level text:	A receive operation	ation from a server with &1 detected an e
No./Type/Le		
- Substitution v No./Type/Le 001 A		Type:
		Length: Change Decimals: Delete

This dialog allows a message to be added or an existing message to be changed. Substitution variables can be added, changed or deleted. Refer to the User Defined Messages in the Application Design Guide before editing the message file.

The following details may be specified:

Message ID

First Level Text

Second Level Text

Substitution Variables

Substitution Variable Type

Substitution Variable Length

Substitution Variable Decimals

1 8.7 Message File Maintenance

Message ID

Mandatory. Specify the ID of the message.

Rules

- The Message ID must not be the same as any other message ID in the message file.
- Maximum length is 7 characters.

Tips & Techniques

• Standards should be defined for all message IDs. For example, message IDs are generally in the format or AAA9999 where AAA identifies a group of messages and 9999 is a sequence number within the group.

1 8.7.5 Add/Change Message Text

First Level Text

Mandatory.

Specify the first level text message text associated with the Message ID. Substitution Variables can be embedded in this field.

Rules

- Cannot be blank.
- Maximum length is 132 characters.
- Substitution variables must use & as the first character and must be followed by a numeric value. For example, &1 or &2.

Tips & Techniques

- Remember that the maximum length of the entire message including any substitution values is 132 characters. Do not exceed this length.
- 1 8.7.5 Add/Change Message Text

Second Level Text

Specify the second level text message text associated with the Message ID. Second level text is optional.

Tips & Techniques

- Always enter second level text. Use the second level text to provide more detailed information for the user. For example, you can include a corrective action if this is an error message.
- 1 8.7.5 Add/Change Message Text

Substitution Variables

Specify the substitution variable number for the variable that has been defined for the First Level Text. Variables are number &1, &2, etc.

A substitution variable is used to pass a value to a message. For example, a message might have the following text: "File &1 in library &2 cannot be compiled." The value of substitution variable &1 can be passed to the message so that the text becomes: "File PSLMST in library DEMOLIB cannot be compiled".

Rules

• A definition must be created for each variable used in the first level text.

Also See

Substitution Variable Type Substitution Variable Length Substitution Variable Decimals ↑ 8.7.5 Add/Change Message Text

Substitution Variable Type

Specify the type of the substitution variable.

Rules

Allowable Values are:

Alpha Substitution variable will be an alpha value.

Numeric Substitution variable will be a numeric value.

Also See

Substitution Variable Length

Substitution Variable Decimals

1 8.7.5 Add/Change Message Text

Substitution Variable Length

Mandatory.

Specify length of the new substitution variable.

Rules

• The maximum length is 132.

Tips & Techniques

• The maximum length of the entire message including the substitution values is 132 characters. Do not exceed this length.

Also See

Substitution Variable Type Substitution Variable Decimals 1 8.7.5 Add/Change Message Text

Substitution Variable Decimals

Specify the number of decimals if the new Substitution Variable Type is numeric.

Rules

• Cannot exceed the Substitution Variable Length.

Also See

Substitution Variables 1 8.7.5 Add/Change Message Text

8.8 Export and Import

LANSA supports the concept of exporting and importing application definitions and data to move information between different partitions or different LANSA Systems.

If you need to create an export where only a small number of objects is required, use the 8.8.1 Quick Export facility. To move large volumes of objects or a LANSA application and data use the LANSA Deployment Tool. (For details, refer to the LANSA Deployment Tool Guide.)

An import is performed to a specific partition.

To execute an import, you must have first used LANSA's export feature which creates files that can then be imported into Visual LANSA. Refer to Exporting and Importing in the *iSeries User Guide*.

A LANSA import can be executed using the following methods:

- Use the *Tools* menu, and select the Quick Export provides a simple and easily accessible way for users to create exports of Visual LANSA object definitions. option within Visual LANSA to import from a directory on a local drive
- Use the 8.8.4 LIMPORT icon in the LANSA, Settings and Administration folder, to import from a directory on a local drive
- Use the 8.8.5 LIMPORT (from IBM i) icon in the LANSA, Settings and Administration folder, to import using shared folder support on an IBM i Server.

Before you start to import data into the repository, your repository must be correctly initialized.

Also See

8.8.3 Import ConsiderationsSystem InitializationPartition Initialization↑ 8. Environment Settings

8.8.1 Quick Export

Quick Export provides a simple and easily accessible way for users to create exports of Visual LANSA object definitions.

xport Objects		
lame	 Description 	Туре
e@@rrno @	Relative record number	Field
📒 @@UPID	Field update / access identifier	Field
📒 ADDRESS1	Street No and Name	Field
📒 ADDRESS2	Suburb or Town	Field
📒 ADDRESS3	State and Country	Field
🔁 DEPTAB (DC@DEMOLIB)	Department code table	File
📒 DEPTMENT	Department Code	Field
📒 EMPNO	Employee Number	Field
📒 GIVENAME	Employee Given Name(s)	Field
📒 MNTHSAL	Monthly Salary	Field
📒 PHONEBUS	Business Phone Number	Field
😬 PHONEHME	Home Phone Number	Field
📒 POSTCODE	Post / Zip Code	Field
🔁 PSLMST (DC@DEMOLIB)	Personnel	File
PSLSKL (DC@DEMOLIB)	Personnel skills	File
📒 SALARY	Employee Salary	Field
🔁 SECTAB (DC@DEMOLIB)	Section code table	File
ECTION SECTION	Section Code	Field
📒 SKILCODE	Skill Code	Field
🔚 STARTDTE	Start Date (DDMMYY)	Field
📒 STARTDTER	Start date (YYMMDD)	Field
📒 SURNAME	Employee Surname	Field
📒 TERMDATE	Termination Date (DDMMYY)	Field
E TERMDATER	Termination Date (YYMMDD)	Field

Accessing Quick Export

Quick Export can be accessed from the main LANSA toolbar or from any repository object context menu, provided that it is available for export. The dialog is modeless, allowing the LANSA Editor to be accessed to add further objects to the export. When the *Quick Export* dialog is closed, all items in the export are removed.

Adding Objects to the Quick Export

There are three ways to add objects to a Quick Export:

- Execute from a context menu. The currently selected items will be automatically added to the export.
- Drag and drop from any of the repository browsers.
- Use the *Cross References* toolbar icon to get cross references for the selected objects in the *Quick Export* dialog, for example Fields in a File, Functions in a Process.

Running the Export

When executed, the definitions of the objects are exported to the partition's *QuickExport* folder. This is a subfolder of the partition folder. The resulting export is then zipped as QuickExport.zip. Any existing objects in the QuickExport folder will be lost when the export is executed.

This is a modal process and will lock the LANSA Editor while the export is running. When complete, the export log will be shown.

As the name suggests, *QuickExport* is not intended for large exports. If you need to export large volumes of objects, you should use the *Deployment Tool*.

🞯 Quick Export	- 0 ×					
E 🗶 🕨 🗖						
Export Objects C:\Users\Stewart\AppData\Local\Temp\Export.Log						
Object Export Facility Started 29/05/12 14:52:11 (14:52:11)						
This procedure is being logged into file C:\Users\Stewart\AppData\Local\Temp\export.log (14						
Export of table lxxf60 (to C:\LANSA\IDE13DEV\X_WIN95\X_LANSA\x_ide\QuickExport\lxxf60.as						
Export of table lxxf60 (to C:\LANSA\IDE13DEV\X_WIN95\X_LANSA\x_ide\QuickExport\lxxf60.as						
Export of table lxxfof (to C:\LANSA\IDE13DEV\X_WIN95\X_LANSA\x_ide\QuickExport\lxxfof.asc						
Export of table lxxfof (to C:\LANSA\IDE13DEV\X_WIN95\X_LANSA\x_ide\QuickExport\lxxfof.asc) completed no						
Definition of file PSLMST in library DC@DEMOLIB successfully exported. (14:52:12)						
Export of table lxxfof (to C:\LANSA\IDE13DEV\X_WIN95\X_LANSA\x_ide\QuickExport\lxxfof.asc) in progress. PI Export of table lxxfof (to C:\LANSA\IDE13DEV\X_WIN95\X_LANSA\x_ide\QuickExport\lxxfof.asc) completed no						
Definition of file PSLSKL in library DC@DEMOLIB successfully exported. (14:52:12)	.) completed no					
Definition of field @@RRNO successfully exported. (14:52:12)						
Definition of field @@UPID successfully exported. (14:52:12)						
Definition of field/component ADDRESS1 successfully exported. (14:52:13)						
Definition of field ADDRESS2 successfully exported. (14:52:13)						
Definition of field/component ADDRESS3 successfully exported. (14:52:13)						
Export of table lxxfof (to C:\LANSA\IDE13DEV\X_WIN95\X_LANSA\x_ide\QuickExport\lxxfof.asc) in progress. PI						
Export of table lxxfof (to C:\LANSA\IDE13DEV\X_WIN95\X_LANSA\x_ide\QuickExport\lxxfof.asc) completed no						
Definition of file DEPTAB in library DC@DEMOLIB successfully exported. (14:52:13)						
Definition of field/component DEPTMENT successfully exported. (14:52:13)						
Definition of field/component EMPNO successfully exported. (14:52:13)						
Definition of field/component GIVENAME successfully exported. (14:52:13) Definition of field/component MNTHSAL successfully exported. (14:52:14)						
Definition of field PHONEBUS successfully exported. (14:52:14)						
Definition of field PHONEHME successfully exported. (14:52:14)						
Definition of field POSTCODE successfully exported. (14:52:14)						
Definition of field/component SALARY successfully exported. (14:52:14)						
Export of table lxxfof (to C:\LANSA\IDE13DEV\X_WIN95\X_LANSA\x_ide\QuickExport\lxxfof.asc) in progress. PI						
Export of table lxxfof (to C:\LANSA\IDE13DEV\X_WIN95\X_LANSA\x_ide\QuickExport\lxxfof.asc) completed no						
Definition of file SECTAB in library DC@DEMOLIB successfully exported. (14:52:14)						
Definition of field/component SECTION successfully exported. (14:52:14)						
Definition of field/component SKILCODE successfully exported. (14:52:14)						
Definition of field/component STARTDTE successfully exported. (14:52:14)						
	*					

8.8.2 LANSA Import

By using the LANSA Editor *Tools* menu and selecting the Import option, you can execute an import while within the LANSA development environment.

LANSA Import				? 🛛
Look in:	C IMPORTS	v G	1 10	
My Recent Documents Desktop My Documents	COMMANDS COMPINIT DEF_DEM DEF_SYS DEMO GRAFVARS GROUPS INDEPEND LWIZ MESSAGES MESSAGES newpart OV_BIFS STD_BIFS	C Std_weblets StS_INIT SYSVARS TEMPLATE UPGRADE VLFramework WebPage WEBUTIL		
My Computer	File name:		*	Open
	Files of type:	LANSA Import (*.del)	*	Cancel
My Network		Open as read-only		

You must specify the directory that contains the files to be imported into the LANSA Repository.

The files in this directory should have been created by LANSA export. A lxxdir.del file should be included in the file set to identify the import.

Look in

Specify the directory that contains the files to be imported.

File Name

Specify the file name containing the import listing. By default, LANSA imports create a **lxxdir.del** file that contains import listing.

Files of Type

Specify the type of file to locate. The default for the import listing file is a ***.del** file.

↑ 8.8 Export and Import

8.8.3 Import Considerations

Following are some considerations when executing imports:

- The exported data should have been formatted with a Target System Type of PC.
- Exports are performed at a partition level. If you have exported data from a different partition than the selected import partition, a warning will be displayed. For example, an export from the DEM partition that is imported into the TST partition will receive a warning message.
- Always display and review the import log after the import has completed. It is recommended that you review any warning messages in the log. If there are any fatal messages, a message will be displayed that the import has failed.
- It is important to identify if you are using an import from a multilingual partition. If your export was created from a multilingual partition, you must import into a multilingual partition.

Also See

Exporting Objects - Tips and Techniques in the *iSeries User Guide*.

1 8.8 Export and Import

8.8.4 LIMPORT

By using the *LIMPORT* icon in the LANSA, Settings and Administration folder, you can execute an import where the import data is from a directory on a local drive. The Standard LIMPORT dialog will be displayed.

Standard LIMPORT 💦 🔲 🔀					
Language	ENG				
Partition Identifier	DEM				
LANSA and OS User	PCXUSER				
Database User	DBA				
Database Password					
Database Name	LX_LANSA				
Database Type	SQLANYWHERE				
Import From Directory x:\xxxxxxx					
OK Cancel	Help Parameter Help				

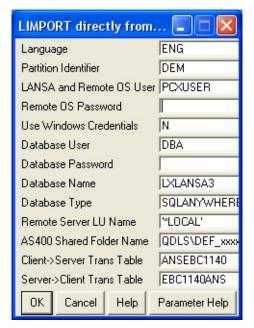
The Visual LANSA development environment does not have to be active to perform an import, but the Visual LANSA database services must be active so that the import can establish a database connection.

Use the *Parameter Help* button for detail about any of the required input fields.

↑ 8.8 Export and Import

8.8.5 LIMPORT (from IBM i)

By using the *LIMPORT (From* IBM i) item in the LANSA, Settings and Administration folder, you can execute an import where the imported data is stored in shared folder on a IBM i server. The LIMPORT directly from IBM i dialog will be displayed.



You must have an active connection to the IBM i server with a valid IBM i profile and password. The profile must have all of the proper IBM i authorities to access the shared folder containing import data. The import data in the shared folder should have been formatted as a PC export.

The Visual LANSA development environment does not have to be active to perform an import, but the Visual LANSA database services must be active so that the import can establish a database connection.

Use the *Parameter Help* button for detail about any of the required input fields.

↑ 8.8 Export and Import

9. Compiler Settings

The compiler settings used by Visual LANSA are not usually a significant part of the decisions required to install Visual LANSA. Visual LANSA generates portable C and C++ code that just requires a supported compiler to compile it correctly. Which compiler the Visual LANSA install chooses to use does not normally effect the LANSA application.

But compilers are software and software has defects, so it may be important. Its particularly important when considering the Windows Build Machine. It is also recommended to use the same compiler on every developer PC.

The Visual LANSA install searches the machine for a compiler that is supported by the current version of Visual LANSA. If one is not found, it will install the Microsoft compiler shipped with Visual LANSA (LANSA-shipped Compiler). If one is found, it does nothing .

When you logon to Visual LANSA, if the LANSA-shipped compiler is enabled, that is used. Otherwise the same search for a supported compiler is performed as occurs with the Visual LANSA install.

The details of the algorithm are as follows:

1. Check the LANSA-shipped compiler registry setting. The keys used for each version of LANSA are listed:

V11.xHKEY_LOCAL_MACHINE\SOFTWARE\LANSA\MicrosoftCompiler V12.xHKEY_LOCAL_MACHINE\SOFTWARE\LANSA\MicrosoftCompiler V13.xHKEY_LOCAL_MACHINE\SOFTWARE\LANSA\MicrosoftCompiler

In these keys the value Enabled = 1 indicates that the LANSA-shipped compiler is to be used and the InstallDir value is where the compiler is installed. So installing a supported Microsoft compiler after the install will not change the compiler used. The LANSA-shipped compiler needs to be disabled by setting Enabled = 0.

- 2. Search for the latest supported compiler. Microsoft's compilers are designed to be installed side-by-side. Each release has a unique identification. The latest compiler found that is supported by the current version of Visual LANSA is the one used. So, if the LANSA-shipped compiler is not enabled, and a later supported Microsoft compiler is installed, it will be used.
- 3. Use whatever values have been set in the windows environment for PATH, INCLUDE and LIB.

For example, If Microsoft Visual Studio 2005 was the latest compiler installed on the PC when Visual LANSA V12 was first installed, then Microsoft Visual Studio 2005 will be used by the Development Environment. If Microsoft Visual Studio 2008 was then installed, then that would be used by Visual LANSA V12. But, if Microsoft Visual Studio 2012 was then installed, it would not be used because it is not supported by Visual LANSA V12.

Likewise, if Microsoft Visual Studio 2012 is the only compiler installed when you first install Visual LANSA V12 on a machine, it will be treated as if a compiler is not installed and the LANSA-shipped compiler will be installed.

Your local LANSA Agent will give you details about supported compiler versions or you can refer to the LANSA Supported Platform document.

10. Troubleshooting

This section of the Administrator's guide will assist in the troubleshooting process for your LANSA applications. Please review the following:

10.1 Reporting Problems to LANSA Support

10.2 Gathering Information About LANSA

If you are using a specific LANSA product or feature, please refer to the troubleshooting section of the appropriate product guide:

LANSA Integrator Troubleshooting

LANSA Open Troubleshooting

LANSA Client Troubleshooting

Visual LANSA Framework Troubleshooting

Logical Modeler Troubleshooting

Installing LANSA on Windows Troubleshooting

Installing LANSA on iSeries Troubleshooting

Deploying LANSA Applications on Linux Troubleshooting

Also See

Refer to the support pages on the LANSA Web site at www.lansa.com/support

10.1 Reporting Problems to LANSA Support

Refer to the LANSA Help Desk area of the web site for information on how to contact your local LANSA support group and the information that may be requested to investigate your LANSA issue:

Technical Resources at LANSA Help Desk.

Please review the following:

10.1.1 Problem Analysis

- 10.1.2 Problem Details Checklist
- 10.1.3 Procedure for Reporting Application Failures and Exceptions

10.1.4 Requesting new Product Features and Enhancements

Also See

10.2 Gathering Information About LANSA

10.1 Reporting Problems to LANSA Support

10.1.1 Problem Analysis

Refer to the information regarding improving issue resolution times on the Technical Resources section of the LANSA Web site for the types of questions that LANSA Support will ask regarding your LANSA issue. It will improve the call resolution time if you have this information available when reporting your problem.

Also See

10.1.2 Problem Details Checklist

↑ 10.1 Reporting Problems to LANSA Support

10.1.2 Problem Details Checklist

When submitting a problem to LANSA Support, please refer to the following Technical Note for the types of information that will be requested. Some of the required information can be obtained by running the *About LANSA*. Refer to 10.2 Gathering Information About LANSA

- LANSA software versions details (build number)
- list of all EPCs applied to LANSA software (refer to 10.2.3 EPC Listings)
- list of licensed products for this application environment
- operating system version details (including service pack details)
- user profile details (if appropriate)
- hardware details (make, model, memory, free disk space, total disk space, etc.)
- job logs (if appropriate)
- error logs (refer to 10.2.4 X_ERR.LOG Files)
- dump file (if appropriate)
- trace files (refer to 10.2.7 x_tracennn.txt files)
- full error message numbers and details
- print screens of the application
- details about the circumstance of the error.

Also See

- 10.1.1 Problem Analysis
- 10.1.3 Procedure for Reporting Application Failures and Exceptions
- ↑ 10.1 Reporting Problems to LANSA Support

10.1.3 Procedure for Reporting Application Failures and Exceptions

Refer to the following area of the web site for information: Technical Resources at LANSA Help Desk.

↑ 10.1 Reporting Problems to LANSA Support

10.1.4 Requesting new Product Features and Enhancements

In addition to reporting problems with the software, you may use LANSA Support to request enhancements to the software or to request new features to be added.

Your comments and input are important to the ongoing development of the LANSA software.

When providing a request for a new feature or enhancement, please provide details as to why the enhancement is required or what the benefits of the feature would be. This type of information helps us to better understand the objective of the change or the limitation you are trying to overcome.

10.1 Reporting Problems to LANSA Support

10.2 Gathering Information About LANSA

There are many sources of information about your LANSA system including error logs and tracing files. *About LANSA* is also an important feature that will help LANSA Support identify information about your specific software installation.

- 10.2.1 About LANSA for iSeries
- 10.2.2 About Visual LANSA
- 10.2.3 EPC Listings
- 10.2.4 X_ERR.LOG Files
- 10.2.5 Dump Files
- 10.2.6 Producing Debug Symbols for Your LANSA Application
- 10.2.7 x_tracennn.txt files
- 10. Troubleshooting

10.2.1 About LANSA for iSeries

To display details of this LANSA system, such as LANSA version number, OS version number, libraries and IFS, you can execute the following command from an OS/400 command line:

<pgm lib>/LANSA REQUEST(ABOUT)

(where **<pgm lib>** is the name of the LANSA program library)

or

you can use the *LANSA About* in the General System Information on the Administration Menu.

An option is available to save this data for sending to LANSA Support.

Also See

ABOUT parameter in the LANSA Command Parameters in the *iSeries User Guide*.

10.2 Gathering Information About LANSA

10.2.2 About Visual LANSA

Using the *Help* menu in the Visual LANSA Editor and selecting the *Product Information* option provides you with information about the current release of Visual LANSA. The information displayed in the product dialog can be saved and forwarded to LANSA Support.

Refer to Product Information in the User Guide.

10.2 Gathering Information About LANSA

10.2.3 EPC Listings

In order to diagnose a problem, it is important that you have identified ALL of the EPCs that have been applied to the LANSA software being executed.

Refer to the EPC Information in the Technical Resources section of the LANSA Web site for:

- EPCs required by product and version
- Common questions and Answers regarding EPCs
- EPCs available for download for registered customers on Maintenance.
- 10.2 Gathering Information About LANSA

10.2.4 X_ERR.LOG Files

LANSA logs error messages to an X_ERR.LOG file. This file may contain important information for troubleshooting your application.

Information is available at the LANSA Web site support pages at www.lansa.com/support. Review the LANSA Help Desk information or Technical notes for more details about using the x_err.log file.

10.2 Gathering Information About LANSA

10.2.5 Dump Files

When an **unhandled exception** occurs, a dump file is written to the runtime temporary directory, TPTH if it is known, otherwise it is written to %TEMP%. There is only one dump file kept so each time an unhandled exception of the same name occurs, the dump file is overwritten.

If a user interface is available then a message like the following is displayed:



If there is not a user interface, for example a web job, then there may be very little feedback to the user interface that an unhandled exception has occurred. Either one or both of X_ERR.LOG or the Windows Event Viewer may contain details of the unhandled exception.

The X_ERR.LOG message will be the last one and may look similar to this:

```
Mon May 28 15:56:06 2012
Release 13.0.0 Build 4055 Windows 7 Ultimate Edition Service Pack 1
---LANSA Installation/Upgrade 13.0.0 Build: 4xxx (23/05/2012 10:26:25
AM)
Message : (0885) - Access violation (exception code c0000005) at address
06345604 should be reported to your product vendor as soon as possible by
supplying the files X_ERR.LOG and X_DLL.DMP in directory c:\Temp\.
Routine : X_DLL
Job Number: 005400 OS User : robert
ITRM=9999999 INST=MSI TPTH=c:\temp\ DBII=TRUNK2
DBUT=MSSQLS GUSR=QOTHPRDOWN PROC=*WAM
WAML=LANSA:XHTML WVAR=0 W3ST=1 WXSL=YY LANG=ENG
PART=DEX USER=PCXUSER CMTH=T CDLL=LCOMGR32.DLL
WSTY=IISB ITHP=N
ROOT=C:\program files\lansa\X_WIN95
```

The Event Viewer message will be in the *Application Log* and may look similar to this:

Event Properties - Event 0, LANSA		×
General Details Image: String of the strin		
+ System - EventData	*	
Fatal Error Message : (0885) - Access violation (exception code c0000005) at address 06345604 should be reported to your product vendor as soon as possible by supplying the files X_ERR.LOG and X_DLL.DMP in directory c:\Temp\. Routine : X_DLL	Ŧ	
Сору		ose

In the case of a server job crashing, the easiest way to locate the dump file and X_ERR.LOG file is to look in the *Event Viewer*. This is because the location of these files may be different for each user, whereas *the Event Viewer* is in one place. It may be useful to configure a global LANSA temporary directory by specifying TPTH in the X_RUN environment variable, although all users will write to the same dump file and therefore overwrite each others state. It is set like this:

X_RUN=TPTH:C:\TEMP\

When you locate the dump file, **C:\TEMP\X_DLL.DMP** in this example, check that the date and time of the dump file match the timestamp in **X_ERR.LOG** or the *Event Viewer*. The time may not match exactly when a message has been displayed to the user, as the dump file is written out before the message is displayed. The x_err.log and the *Event Viewer* entries are written afterwards.

The dump file will be critical to resolving your issue, so if there is a matching file ensure that it is sent to your LANSA supplier as part of the package of information. A dump file can be quite large, so it can be worthwhile compressing it before sending or uploading it to an ftp site.

Note that in rare circumstances the dump file may be called **LANSA_UNNAMED.DMP** and it will be in **%TEMP%**, not TPTH.

Also see

10.2.6 Producing Debug Symbols for Your LANSA Application↑ 10.2 Gathering Information About LANSA

10.2.6 Producing Debug Symbols for Your LANSA Application

In order to debug using a 3GL debugger (e.g. Microsoft Visual Studio), the environment variable X_DEV needs to be defined. This has two purposes. The first is a full debug in a development environment. The second is release debug for use in deployed environments so that process dumps can be fully interpreted by support staff.

Full debug is selected by defining the X_DEV environment variable and choosing Debug Enabled. This is useful when 3rd party programming languages are used with LANSA.

Release debug is selected by defining the X_DEV environment variable and NOT choosing Debug Enabled. The resultant DLL will be the same size as if X_DEV did not exist. An extra PDB file is produced which must be kept as a pair with the DLL.

A typical situation that requires release debug is when an access violation occurs that is hard to reproduce. The end-user takes a process dump and sends it to their application's support service. The release debug files are then used to determine exactly what the program state was when the access violation occurred. This is very useful for LANSA support, even if the application developer has no intention of using these files themselves. For further information, refer to the Microsoft MSDN site: http://msdn2.microsoft.com/en-us/default.aspx and search for "Symbol Server".

LANSA strongly recommends that release debug is chosen before shipping an application to end users. It is important that the DLL and PDB files produced are kept as a set as the PDB files cannot be regenerated later, even if the source file is exactly the same. This is a Microsoft Windows restriction. Do not ship the PDB files with the application, just keep them in the event that a difficult to diagnose defect occurs.

The generated source code is also required in order to fully resolve dump files. It is possible to produce this when needed, provided the original Visual LANSA development environment is retained.

10.2 Gathering Information About LANSA

10.2.7 x_tracennn.txt files

When application tracing has been turned on, x_tracennn.txt files are created. These files may contain important information for troubleshooting your application. Tracing is a developer tool that will impact application performance and should only be enabled when requested by LANSA Support.

Information is available at the LANSA Web site support pages at www.lansa.com/support. Review the LANSA Help Desk information or Technical notes for more details about using the x_tracennn.txt files.

10.2 Gathering Information About LANSA

Appendix A. LANSA's Customer Services

Your local LANSA agent can provide a range of services including: Education Services Conferences and User Groups LANSA Customer Support Consulting Services Refer to the www.lansa.com and select *Partners* for information about your local LANSA agent.

Education Services



LANSA's education offerings provide the knowledge, skills, and experience to help your company maximize the return on your investment. Through these educational programs, you will learn to design and implement new systems efficiently so you can take full advantage of the product architecture and avail yourself of evolving trends in technology.

Because the training needs of each individual are very different, it is essential to choose training courses correctly tailored to the needs of each person.

LANSA operates high quality skills development courses for programmers, analysts, system designers, administrators, and end users. Given the openness of the LANSA product family, any of the courses available can be tailored or combined with other courses to target specific integration needs.

At LANSA, we believe that education is far more than teaching students how to use our toolset. To be successful with our products, students must fully understand our tools, how best to use them and how the tools interact with their own company's existing environment.

Our courses are developed and delivered by experienced professionals.

A Variety of Choices

LANSA offers a variety of educational choices to suit your needs.

Tutorials

Online tutorials are supplied with the LANSA products as part of the core documentation. For example, to learn how to build Internet applications with Web Application Modules, refer to the WAM Tutorials in the LANSA for the Web Guide. Tutorials can be accessed using the Tutorial icon on the toolbar, or from the Online Directory page.

Public Courses

LANSA courses are offered publicly at multiple locations. Spacious, comfortable classrooms incorporate state of the art technology to provide students the best environment for learning. Instructors provide immediate answers to questions and offer valuable insights into using the LANSA products. Some public courses may also offered over the Internet. Refer to LANSA LANSA Education pages at the LANSA Web site for public courses in your region.

On-Site Courses

Courses can be conducted at your company's facility or a location of your choice. You will receive the appropriate education at the appropriate time, and your staff will learn in the comfort and convenience of their own environment. LANSA can customize any course to meet the unique needs of your organization. Refer to the LANSA Web site at LANSA Education for information about LANSA courses.

Supplementary Education Solutions

Your LANSA education does not end when you leave the classroom. Utilize LANSA's experienced consultants to help apply what is learned in class. By working side by side with your staff on your first project, we can provide skills transfer, answer any questions, and help apply what was learned in class to your unique situation.

↑ Appendix A. LANSA's Customer Services

Why use LANSA Education?

With LANSA education, you will learn from the experts. Our courses are developed and taught by experienced professionals who have years of application development experience plus an intimate knowledge of the LANSA technology. They bring their real world experience into the classroom for your benefit.

Our courses are modular, allowing flexibility to leverage your existing skill set with the new technology you are facing. Using modularization, we can focus on the education needs of your particular environment. This provides a reduced learning curve, allowing you to quickly become more productive.

Our courses are designed for hands-on learning. Classes are very interactive and students are encouraged to participate through questions, comments or by following along with the instructor from their own workstation.

We feature comprehensive exercises that reinforce key topics. These exercises make up a large percentage of our course material. We believe students learn best by doing, and we apply this philosophy to our course material.

Our courses educate and encourage students to work independently. By utilizing reference material and interactive help facilities, we help students to manage on their own once the education is completed.

Need More Information?

You will find the most up to date information about the courses available and

course dates, at the LANSA Web site at www.lansa.com.

Conferences and User Groups

The people who are using LANSA are a great source of information. LANSA user group meetings and the LANSA International Conferences allow information exchange between LANSA developers and keep you up-to-date on the latest developments.

User groups are typically organized by the LANSA users in a specific region and may be hosted by the local LANSA agent. User groups create a community for sharing ideas among LANSA developers. In most cases, the LANSA user groups operate independently.

LANSA Conferences are held annually at different locations around the world. These conference assembles a diverse range of LANSA developers and business partners. It offers an excellent opportunity for building a network of contacts and for showcasing new LANSA-based products. Conferences include a variety of training streams for different products, technologies and professions. Speakers often include members of the LANSA Product Centre software development team.

For information about LANSA Conferences and User Groups, contact your local LANSA agent or visit the LANSA Web site at www.lansa.com.

↑ Appendix A. LANSA's Customer Services

LANSA Customer Support

There are several methods of contacting our Technical Support team including phone, email and fax. Please use whichever method is most convenient for you. Refer to the LANSA Help Desk at the LANSA Web site for details.

Whichever method you use to log your request for assistance, please provide as much applicable information as possible. For example:

- IBM i Job Logs
- RDML Listings
- Product Version Number
- Operating System Environment
- Error Message Screen Prints and etc.

It is recommended that one person be nominated at each site to look after support line requests. If your site has a problem that cannot be resolved, the nominated person can call/e-mail the support for help. The support staff will work with you in resolving the problem.

You may also find the answer to your support and technical questions at LANSA Support on the LANSA Web site.

↑ Appendix A. LANSA's Customer Services

Consulting Services

Your LANSA distributor offers a Consulting Services Division that will assist you to build and adapt mission-critical applications using the LANSA family of products and tools. This includes a variety of host based IBM i systems, client/server systems, Windows systems, Internet applications and much more.

LANSA's professional consultants provide expertise in the areas of systems integration, project management, joint application development, data and process modeling, programming and education. Highly skilled experts provide industry knowledge, skills transfer and product competency.

While LANSA products offer immediate productivity gains, the most successful strategy is to leverage LANSA's inherent architecture throughout the organization, thereby optimizing the business benefits achieved through sensible system design.

This can be achieved by supplementing the LANSA products with a broad spectrum of professional services. By offering technical and strategic consulting services to our clients, it ensures that they will understand the benefits of employing LANSA as part of a major technology investment plan. This plan will not just support your business, but will stimulate opportunity and growth by empowering the users of corporate information to do whatever is necessary to keep ahead of the competition, as well as achieving significant cost savings.

As a premier provider of repository architecture, LANSA has demonstrated proven commitment to the open and distributed environments that are becoming a requirement in today's businesses. With a worldwide network of companies using LANSA as a data repository, we can draw from a wealth of experience across all industries.

LANSA consultants are specialists in developing information solutions to address business issues in various types of industries. When you work with LANSA consultants, you work with a team committed to excellence in information systems.

↑ Appendix A. LANSA's Customer Services