Dynamics AX

Microsoft® Dynamics™ AX 4.0

Microsoft Dynamics ISV Software Solution Test Guidelines

May 23, 2007

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This test specification is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Contents

Introduction

This Document

The Microsoft® Dynamics™ AX ISV Software Solution Test Guidelines describes the technical requirements that an application must meet to integrate and operate with Microsoft Dynamics AX version 4.0. For information about testing of the requirements, see the "How to Comply" and "Test Methodology" sections in each requirement description.

Program Overview

Welcome to the ISV Software Solution Test Guidelines for Microsoft Dynamics AX 4.0. This document describes the requirements that an Independent Software Vendor (ISV) application must meet to integrate and operate with Microsoft Dynamics AX.

The goal of the test is to increase the quality of applications that run in the Microsoft Dynamics AX environment. The purpose of this test is to give the market the assurance that ISV applications built on Microsoft Dynamics meet technical requirements that ensure a high standard.

The test guidelines are designed to walk you through the test process and to help you make sure that your application meets the requirements.

This document provides the following:

- An explanation of the testing process
- Definitions of the software requirements
- Descriptions of development, test, and documentation best practices

To pass the test, an ISV must demonstrate the development quality of its product and its ability as a software company to maintain and enhance that product in the future. Therefore, the test includes a technical review and an in-lab inspection. The actual test is administered and conducted by a third-party vendor.

We welcome your comments and suggestions. Please contact *dyncert@microsoft.com* with your feedback.

Contents of the Test Guidelines

The Microsoft Dynamics AX ISV Software Solution Test Guidelines describes the test requirements, recommendations, and best practices. The guidelines are presented in individual, subject-based modules, some of which may be common to other Microsoft Dynamics tests.

This document contains the following sections:

- Introduction (this section) explains the purpose and high-level requirements of the test.
- Testing Process describes how the testing process works, from qualification through communication of test results.
- Documentation Requirements provides a summary of the documentation that you must submit with your application.
- Test Requirements defines in detail each requirement category, how these requirements are tested, and what you can do to make sure that your application meets the requirements.

Products Versions Supported

The intent of the test is to support Microsoft's latest shipping version of a product. For that reason, ISV applications submitted for testing must run on Microsoft Dynamics AX version 4.0 with the latest service pack installed.

Types of Solutions

Microsoft Dynamics solutions fall into three general categories and three setup complexity levels. The category and setup complexity of a solution determines (in part) the type and complexity of the testing requirements and the costs associated with testing the solution.

Figure 1 shows the different solution categories and setup complexity levels.

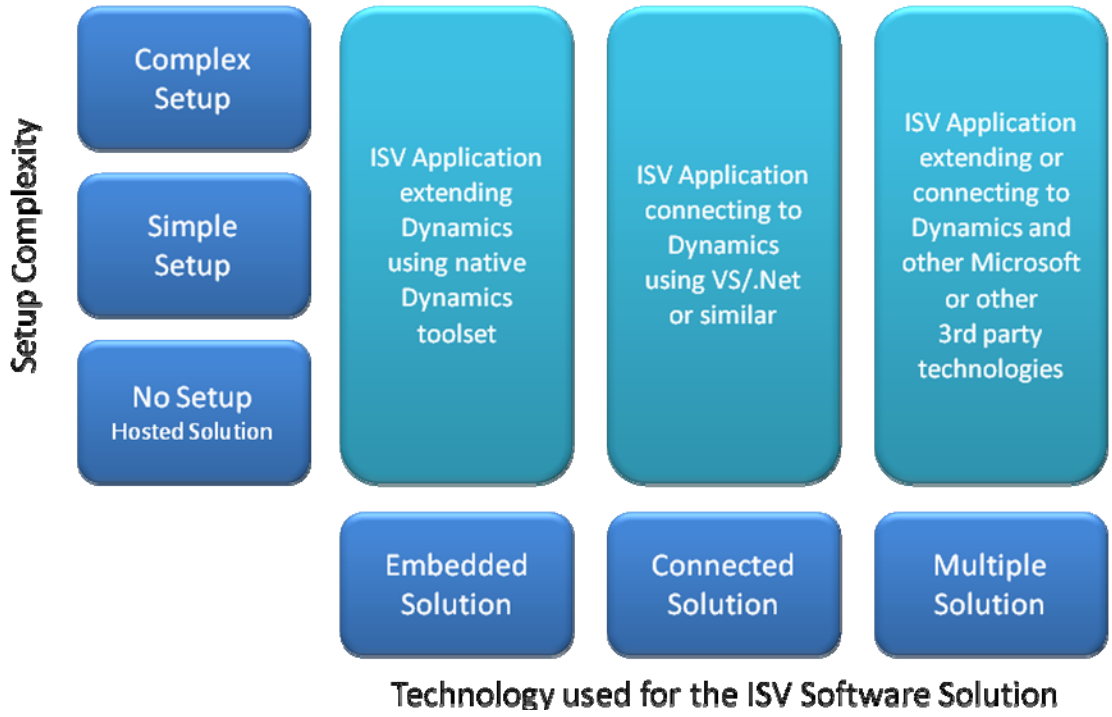


Figure 1
Solution categories and solution setup complexity levels

ISV software solution complexity falls into one of the following categories (listed from least complex to most complex):

- An embedded or *in-product* solution is an ISV application that extends Microsoft Dynamics by using only the tools provided with the Microsoft Dynamics product. For example, an embedded solution can be built in a proprietary development environment, such as the Microsoft Dynamic NAV C/SIDE, C/AL environment, or the Microsoft Dynamics AX Morph, X++ environment.
- A *connected* solution is an ISV application that uses Microsoft Visual Studio, the .NET Framework, or similar tools to connect to the Microsoft Dynamics product.

Typically, a connected solution refers to a standalone product that interoperates with the core Microsoft Dynamics product by using it as a business rules engine. The solution might establish interoperability by using Web Services, .NET Framework assemblies, COM interop, or some other means. The solution might or might not be.NET Framework–based; however, it must run on a Microsoft operating system.

- A *multiple* solution is one that extends or connects to Microsoft Dynamics and other Microsoft or other third-party technologies.

Setup complexity falls into one of the following categories (listed from least complex to most complex):

- No setup (potentially a *hosted* solution) provides services to customers, who do not have to purchase, install, or maintain the software or hardware. Hosted solutions have no setup requirements for end users; however, installing and configuring a hosted solution can be extremely complex, and the test vendor may not have the hardware, custom software, or services that the solution requires.
- A *simple* setup is one that the test vendor can install and configure without requiring a restorable backup, virtual PC (VPC), or other additional assistance.
- A *complex* setup is one that the test vendor cannot completely replicate; for example, solutions that require specific hardware, custom software, or back-end services that the vendor cannot duplicate.

Retesting

Test results are valid for 24 months. When retested, the ISV application must be updated to support the latest Microsoft Dynamics version, including the latest service pack.

More Information

For more information about the functionality of Microsoft Dynamics AX, see the Microsoft Dynamics AX Home Page at http://www.microsoft.com/dynamics/ax.

For more information about the Microsoft Partner Program, see the Microsoft Worldwide Partner Portal Home Page at http://partner.microsoft.com.

For more information about how the ISV test helps you earn partner program points, see the ISV Software Testing Framework page at http://partner.microsoft.com/global/program/competencies/40011374.

For more information about the Microsoft Dynamics ISV/Software Solutions competency, see the Microsoft Dynamics Testing for ISVs page at http://partner.microsoft.com/global/program/competencies/40013116.

Testing Process

Microsoft offers ISV software solution testing through an independent third-party test vendor. ISVs register for the test by visiting the test vendor's Web site referenced on the Microsoft Web page. The vendor site contains a description of the test, as well as an application form with a published test fee schedule.

Depending on the type of solution (embedded, connected, or multiple) and the solution setup (simple or complex), different test methods will apply for which the test fee will vary. You can make your solution available to the test vendor for testing by using any of the following methods:

- Providing the software with installation instructions for the test vendor to install
- Sending a virtual server image of a working configuration of the product to the test vendor
- Using an interactive Live Meeting session to provide access to a working configuration of the product

After you register your software solution and pay the test fee, the test vendor will contact you with detailed information about the testing process you have selected. For processes involving shipping software or virtual server images to the test vendor, you can choose to send the software on media (CD/DVD), upload to an ftp server, or have the test vendor download from your server. If you choose to use Live Meeting to provide access to your solution, the test vendor will contact you to schedule the session.

The following requirements are particularly important:

- Pre-qualification is required. You are responsible for making certain that your solution and organization meet the requirements for submitting and maintaining a Microsoft Dynamics-based solution.
- You must submit a number of documents as part of the test. These are identified in the appropriate test modules, as well as in a summary checklist. See the Documentation Requirements section of this document.
- You must upload documents and your solution to the test vendor’s servers for testing. If your setup is complex, you must be prepared to use Microsoft Live Meeting to demonstrate the solution to the test vendor.

Documentation Requirements

The checklists in this section describe the items that you must include when you submit your software solution for testing or retesting.

First-Time Software Test Requirements

The following checklist describes the documentation that you must include with your solution. Note that a single document may contain information that meets multiple requirements. For that reason, the actual number of documents might be significantly smaller than the number of items in this list.

Requirement	Check
The ISV solution (your application software) and product documentation. This can be in the form of a CD or other distributable media, or you can use a virtual server image or Live Meeting session to demonstrate your software.	
Description of the business functionality that the ISV application provides and examples of key usage scenarios (see Appendix B, Consistency Verification Test).	
Description of the ISV system development life cycle (SDLC) methodology. For more information, see http://technet2.microsoft.com/Office/en-us/library/ae466c4f-b311-4471-9ee0-2ff5641722cb1033.mspx?mfr=true.	
Copy of ISV agreements with end customers.	
Bug tracking documentation with the following elements: • Description of the ISV bug tracking system. • Sample bug report that shows the following: o Bug number o Date that the bug was entered into the system o Severity: Severity 1 = Causes the application to crash or data is lost. Severity 2 = Business process cannot be carried out according to specification. Severity 3 = Bug with workaround o Tracked by name o Name of customer who reported the bug o Procedures for reproducing the bug o File attachments, if applicable o Proposed solution for correcting the bug o State: Open; Will be corrected in the next service pack; will be corrected in next release; closed See Requirement 1.1 and Requirement 5.2.	
Description of the ISV design and development process.	
Description and justification of any exceptions to best practices rules. See Requirement 1.2.	

Partner-facing implementation guide appropriate for VARs or others who intend to deploy your application. This must include the operating system, service packs, database, browsers required and/or supported by the application, and setup/uninstall procedures. See Requirement 2.1. For more information, see the *Microsoft Dynamics AX Implementation Guide* at http://msdn2.microsoft.com/en-us/library/aa834394.aspx.	
A training log that shows how ISV staff are meeting the security education requirements for trustworthy computing. See Requirement 5.1.	
List of all resources that the ISV application adds to Microsoft Dynamics AX and complete instructions for uninstalling the ISV application. If it is not possible to uninstall the ISV application, you must state this in the documentation. See Requirement 8.1.	
List of all registry settings generated during installation. See Requirement 8.2.	
List of all components used by the application. See Requirement 8.2. For more information, see the section “Installing and Configuring Dynamics AX” in the Microsoft Dynamics AX implementation guide for more information (http://msdn2.microsoft.com/en-us/library/aa834394.aspx.	
Sample data for testing. This may or may not be part of the core application installation. See Requirement 8.6. See the standard demonstration data at https://mbs.microsoft.com/partnersource/downloads/releases/ax40demodata.htm?printpage=false.	
Description of backup and restore procedures. See Requirement 9.1.	
Customization and extensibility guide that documents how to extend your application (this is commonly known as a developers guide). See Requirement 11.1.	
Description of the ISV support and escalation process.	
Description of the ISV update process for hot fixes and service packs.	

Software Retest Requirements

The following checklist describes the documentation that you must include with your application when you submit it for retesting. Note that a single document might contain information that meets multiple requirements. For that reason, the actual number of documents could be significantly smaller than the number of items in this list.

Requirement	**Check**
The ISV application (your application software) and product documentation. This can be in the form of a CD or other distributable media, or you can use a virtual server image or Live Meeting session to demonstrate your software.	
Description of the changes to objects and components.	
Description of the changes to the data model.	

Upgrade scripts for the new release. See Requirement 11.2.	
What's New document that describes the business functionality of the new release	

ISV Software Solution Requirements and Recommendations

The Microsoft Dynamics AX ISV solution test requirements ensure that ISV applications integrate with Microsoft Dynamics AX without causing system problems or errors. Microsoft and third-party test vendors worked together to define the minimum requirements that an ISV application must meet to operate successfully in a Microsoft Dynamics AX environment.

Note: The test does not validate the correctness or relevance of ISV application functionality.

This section describes the test requirements and recommendations and the procedures for verifying that each requirement is met. In this document, the word *must* in the text of a requirement means that the item or feature is not optional. The word *should* means that the item or feature is recommended and its inclusion is a best practice, but it is not strictly required. However, these recommendations will be considered for inclusion as requirements in later versions of this test.

Some requirements are technology-specific and do not apply to all ISV applications. Therefore, each requirement indicates the type of ISV technology to which it applies. Additionally, an ISV application might include several technologies. In these situations, the vendor will test those parts of the application that use the technologies that the requirement or recommendation applies to.

If a requirement is defined as applicable to *X++*, it applies to either of the following:

- Any code written in X++ (either business logic or code that implements an integration to an external component), if the vendor in-lab test is performed directly on the code.
- Any application that includes X++ code, if an in-lab test is not performed directly on the code.

Similarly, if a test is defined as applicable to *External*, it applies to either of the following:

- Any code not written in X++ (including DLLs, ActiveX controls, services, applications that have their own user interface, and so on), if the vendor in-lab test is performed directly on the code.
- Any application that includes such code, if an in-lab test is not performed directly on the code.

Each requirement includes a table that indicates the type of technology (X++ or external) and type of solution setup (simple, complex, and host-based) that the requirement applies to.

Development Requirements

Your application must meet the following development requirements:

- Requirement 1.1: A .NET Framework–based ISV application must be compiled on .NET Framework 2.0 or later and must pass the required FxCop tests.
- Requirement 1.2: An X++-based ISV application must not produce best practice tool errors.
- Requirement 1.3: An ISV application must have an About window.
- Recommendation 1.4: An ISV application should follow Microsoft Dynamics AX architectural guidelines.
- Requirement 1.5: Managed assemblies must be strong name
- Recommendation 1.6: ActiveX Controls Should be Digitally Signed

1.1 A .NET Framework–based ISV Application Must Be Compiled on .NET Framework 2.0 or Later and Must Pass the Required FxCop Tests

Type	Test Method	Technology	Solution Category

Required	In-lab test	External	Simple	Complex	Hosted
		Managed code only	✓	✓	✓

Summary and Intent

Applications must use the latest release of the Microsoft .NET Framework and pass the required FxCop tests. FxCop is a code analysis tool that checks managed code assemblies for conformance to the Microsoft .NET Framework design guidelines.

Note: If the application has a small number of unmanaged code elements these will not have to go through the FxCop test.

Resources

For more information, see the following:

- FxCop Web site: http://www.gotdotnet.com/team/fxcop/
- .NET Framework Web site: http://msdn2.microsoft.com/en-us/netframework/default.aspx

How to Comply

An ISV can download FxCop from the FxCop Web site. FxCop uses reflection, MSIL parsing and call graph analysis to inspect assemblies for more than 200 defects.

FxCop includes the following rule libraries, based on the .NET Framework design guidelines that are loaded by default when a new project is created:

- **COM**: Rules that detect COM Interop issues.
- **Design**: Rules that detect potential design flaws. These coding errors typically do not impact the execution of your code.
- **Globalization**: Rules that detect missing or incorrect usage of information related to globalization and localization.
- **Naming**: Rules that detect incorrect casing, cross language keyword collisions, and other issues related to the names of types, members, parameters, namespaces, and assemblies.
- **Performance**: Rules that detect elements in your assemblies that will degrade performance.
- **Security**: Rules that detect programming elements that leave your assemblies vulnerable to malicious users or code.
- **Usage**: Rules that detect potential flaws in your assemblies that can impact code execution.

Rules are assigned one of five importance levels:

- **Critical error**: Issues that are highly visible, that prevent code from operating correctly in common scenarios, or both. Critical error messages should be resolved first, and should be excluded only after carefully assessing the impact of ignoring the error.
- **Error**: Issues at this level have less impact on usability and behavior than critical errors, but should not be excluded without careful assessment.
- **Critical Warning**: Issues that typically have little or no negative impact on code behavior; they are primarily concerned with code maintainability and correcting less-than-optimal choices for visible elements. However, for a minority of cases, these messages are considered errors and so they should be reviewed closely before being excluded.

- **Warning**: Issues that are typically concerned with doing things correctly to keep your code base stable, extensible, and maintainable.
- **Informational**: Messages returned by rules that report information about a target, rather than detecting errors in a target.

No errors should be reported in any rules in the security category (exceptions can be granted for warnings). And all security exceptions must be reviewed by Microsoft.

Test Methodology

The test vendor will execute the FxCop analysis on the ISV application, using the FxCop version specified by the ISV. No errors should be reported in any rules in the security category (exceptions can be granted for warnings). All security exceptions must be reviewed by Microsoft.

Criteria for Passing

This requirement is mandatory. If the ISV application does not pass this requirement, it will fail the test.

1.2 An ISV Application Must Not Produce Best Practice Tool Errors

Type	Test Method	Technology	Application Category		
Required	In-lab test	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

ISV X++ applications must use the same coding standards that the Microsoft Dynamics AX development team uses. Microsoft Dynamics AX has a built-in tool that checks whether an ISV application follows Microsoft Dynamics AX best practices. The tool performs a quantitative test on the application, and logs any errors or warnings that it identifies. Your application must not produce errors when this tool is used. And you should run through the list of warnings to make sure that best practice is implemented.

Resources

For more information, see the following topics in the "Microsoft Dynamics AX Development Best Practices" section of Microsoft Dynamics AX Help:

- Best Practices Checks
- Microsoft Dynamics AX Design Patterns
- Using the InfoLog System

How to Comply

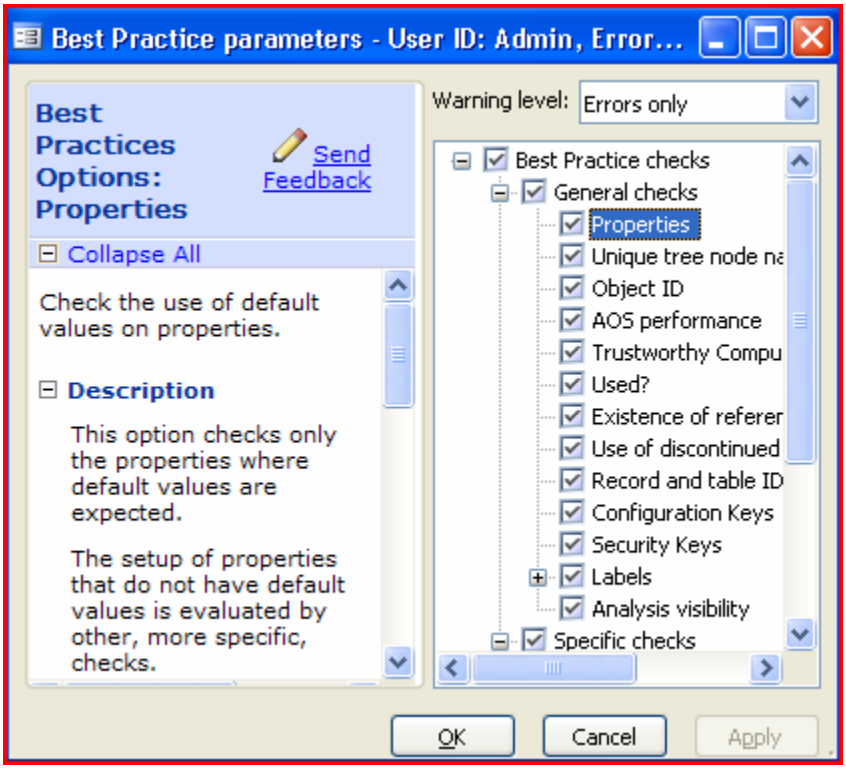
For X++ applications, refer to the Microsoft Dynamics AX Development Best Practices section of Microsoft Dynamics AX Help before and during your development and test processes. It is easier to avoid problems rather than to fix them. Periodically, run the Microsoft Dynamics AX Best Practices Tool on your application. Track and fix any errors (bugs) that the tool identifies. Be sure that the tool produces no errors before you submit your application for testing. If you receive best practices errors and you cannot fix them, provide a written justification for allowing the errors, and include it as part of your submission package. Include any justifications in your checkpoint document.

Do not use the comment function to avoid running the tool on portions of your code. The only exception to this rule is related to security errors. Refer to the Microsoft Dynamics AX Writing Secure X++ Code white paper for details.

Note: The presence of warnings in the best practice tool event log will not cause your application to fail this requirement. However, you should review all warnings carefully, examine the appropriate code, and document the results of your inspection. Use the checkpoint form to document your findings.

Test Methodology

For X++ code, the test vendor will run the Best Practices tool with the warning level "Errors only" (see picture below) and will then review the Microsoft Dynamics AX event log to determine if the application produced any errors. If the application has a large number of errors that are impractical to fix prior to the audit, the ISV must produce a plan to remedy the problems before the application is released.



Criteria for Passing

This requirement is mandatory. If the application produces errors or becomes unstable, it will fail the test.

1.3 An ISV Application Must Have an About Window

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

ISV applications must have an About window that is accessible from the Help menu. The About window must display the name of the application, contact details for the ISV, and version and build information for the application.

Resources

For more information, refer to the *Microsoft Dynamics AX Developers Guide*, located on the Microsoft Dynamics AX product CD.

How to Comply

Follow the guidelines in the *Microsoft Dynamics AX Developers Guide*.

Test Methodology

The test vendor will verify there is an About window that is accessible on the Help menu.

Criteria for Passing

This requirement is mandatory. If the ISV application does not include an About window, it will fail the test.

1.4 An ISV Application Should Follow Microsoft Dynamics AX Architectural Guidelines

Type	Test Method	Technology	Solution Category		
Recommendation	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

Microsoft Dynamics AX Help has sections that describe design concepts and standards. You should follow these guidelines to make sure that your application complies with Microsoft Dynamics AX architectural best practices and design patterns.

Resources

For more information, see the following topics in Microsoft Dynamics AX Help:

- Designing a Microsoft Dynamics AX Application
- Microsoft Dynamics AX Design Patterns
- Introduction to the Microsoft Dynamics AX Business Connector (instructions for creating custom Help).

In addition, refer to the *Microsoft Dynamics AX Developers Handbook*, available at: https://mbs.microsoft.com/partnersource/products/axapta/Documentation/

How to Comply

When you design your application, you should document your design documents in specifications. You must include these documents in your test submission package. Conduct design reviews to make sure that your application uses the design patterns and document your reviews in review reports.

Test Methodology

The test vendor will review your design documents and code for compliance. The test vendor will review a representative sample of application objects to make sure that they comply with Microsoft Dynamics AX guidelines and with the design pattern documentation you submit. The vendor will pay particular attention to tables, forms, classes, and query objects to make sure that they conform to established guidelines.

Criteria for Passing

This is a recommendation only. Failure to comply with this recommendation will not cause the application to automatically fail the test.

1.5 Managed Assemblies Must Be Strong Named

Type	Test Method	Technology	Solution Category		
Required	In-lab review	External	Simple	Complex	Hosted
		Managed Code	✓	✓	✓

Summary and Intent

This requirement is included for security purposes.

Resources

The Sn.exe tool provided with the Microsoft Visual Studio® .NET development system supports the proper use of strong names.

How to Comply

The ISV must apply strong naming to managed assemblies. The exception is that if the ISV application uses a vendor or third-party assembly, the assembly does not need to be signed. The ISV should provide a list of vendor or third-party assemblies.

Test Methodology

The test vendor will use the Sn.exe tool provided with Visual Studio .NET to verify the proper use of strong names.

Criteria for Passing

This requirement is mandatory. If the ISV does not use strong naming for managed assemblies, the application will fail the test.

1.6 ActiveX Controls Should Be Digitally Signed

Type	Test Method	Technology	Solution Category		
Recommended	In-lab review	External	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

This recommendation is included for security purposes. Digital signing helps users decide if they want to trust a control and assures users that files have not been tampered with.

Resources

Code signing certificates are available from several vendors, as described at http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsecure/html/rootcertprog.asp.

The Microsoft Windows SDK Sign Tool is available on MSDN at:
http://windowssdk.msdn.microsoft.com/library/

How to Comply

After the ISV obtains a code signing certificate, the ISV should use the Microsoft Windows SDK Sign Tool to sign the files. The exception is that if the ISV application uses a vendor or third- party assembly or ActiveX control, the control does not need to be signed. The ISV should provide a list of vendor or third-party controls.

Test Methodology

During testing, the test vendor will note any warnings about ActiveX controls without valid certificates.

Criteria for Passing

This is a recommendation only. Failure to comply with this recommendation will not cause the application to automatically fail the test.

User Assistance and Documentation Requirements

Your application must comply with the following user assistance and documentation requirements:

- Requirement 2.1: The ISV must provide an implementation guide for the application.
- Requirement 2.2: Online documentation for an ISV application must use the Microsoft Dynamics AX Help navigation structure.
- Recommendation 2.3: Online documentation for an ISV application should follow Microsoft Dynamics AX Help style guidelines.

2.1 The ISV Must Provide an Implementation Guide

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

You must include a partner-facing implementation guide in your documentation.

ISV partners and customers who use or deploy a Microsoft Dynamics AX application must be able to successfully deploy, configure, and manage the application in an existing Axapta or Microsoft Dynamics AX environment. Your documentation must provide information that allows your partners and customers to successfully install or upgrade your application in such an environment.

Resources

None

How to Comply

To meet this requirement, you must include adequate system requirements, installation, configuration, and upgrade information to allow a partner to implement your application in a new or existing Axapta or Microsoft Dynamics AX environment.

Note: Providing a security hardening guide is a trustworthy computing (security) requirement. You can meet the hardening guide requirement by including in this document a section that describes how to deploy your application in a secure manner.

Test Methodology

The test vendor will review your documentation to verify that you have included adequate implementation information.

A satisfactory guide contains the following sections:

- Description of the solution (the problem it solves)

- Hardware environment
- Operating system requirements
- Installation checklist
- Installation guide
- Operational checklist (daily, monthly, and annual procedures as well as how to perform backups, and so on)

Criteria for Passing

This requirement is mandatory. If the ISV application documentation does not include an implementation guide, the application will fail the test.

2.2 ISV Application Online Documentation Must Use the Microsoft Dynamics AX Help Navigation Structure

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

Your documentation must be easy for the user to access and to navigate. Documentation for a Microsoft Dynamics AX ISV application should provide a user experience that is consistent with the base documentation provided with Microsoft Dynamics AX. Therefore, your documentation must be in the Help .chm file format and must integrate with the Microsoft Dynamics AX Help system for X++ applications. Users must be able to see Help topics by pressing the F1 function key, which starts the Microsoft Dynamics AX Help window. For .NET Framework–based applications, please visit the MSDN HTML Help link in the Resources section, below.

Resources

To satisfy this recommendation, you could use the HTML Help SDK to create your Help system.

For more information, see the following:

- *Microsoft Dynamics AX User Assistance Best Practices Handbook*. Redmond, WA: Microsoft, 2005. AX-300-DVG-001-v02.00-ENUS. Available as a compiled HTML Help file (.chm file) and on Microsoft Dynamics AX Central, the Microsoft Dynamics AX community Web site, at http://www.AXcentral.com/viewtopic.php?t=24&highlight=handbook.
- For specific information about how user assistance and Help information should appear in your application's user interface, see the section, User Experience and Usability Requirements, in this document.
- For style guidelines and procedures to use when you create your .chm file, see the section, "Microsoft Dynamics AX Help" in the Microsoft Dynamics AX Help file.
- For information about Help for .NET Framework–based applications, see the MSDN HTML Help guidelines at http://msdn2.microsoft.com/en-us/library/aa189109(office.10).aspx

How to Comply

To satisfy this recommendation, use the documents listed in the Resources section, above, to design and create your Help system.

Test Methodology

The test vendor will review your Help documentation for compliance and usability. The test vendor will review a representative sample of application modules to make sure that Help is available by pressing F1 and that it follows the navigation structure of the core Microsoft Dynamics AX Help system.

Criteria for Passing

This requirement is mandatory. If the ISV application online documentation does not follow the structure of the core Microsoft Dynamics AX Help system, the application will fail the test.

2.3 ISV Application Online Documentation Should Follow the Microsoft Help Style Guidelines

Type	Test Method	Technology	Solution Category		
Recommended	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

The online documentation for your application should follow Microsoft Help style guidelines. It should tell the user how to perform specific tasks, and it should be easy for the user to understand. Documentation for a Microsoft Dynamics AX ISV application should provide a user experience that is consistent, both in terms of writing style and depth of information, with the base documentation provided with Microsoft Dynamics AX.

Resources

For more information, see the MSDN online Help style guidelines at http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwue/html/ch13c.asp.

How to Comply

Make sure that you have online help and that it provides meaningful information. The guidelines on MSDN will help you to create appropriate content.

Test Methodology

The test vendor will review your Help documentation for style, accuracy, and usability. The vendor will review a representative sample of application modules to make sure that Help topics are appropriate, easy to understand, correct, and adhere to style and user interface guidelines.

Criteria for Passing

This is a recommendation only. Failure to comply with this recommendation will not cause the application to automatically fail the test.

User Experience and Usability Requirements

Your application must comply with the mandatory UI guidelines and it is recommended that you comply with the recommended UI guidelines and follow the user assistance and documentation requirements:

- Requirement 3.1: An ISV application must comply with Windows and mandatory Microsoft Dynamics AX user interface (UI) guidelines.

3.1 An ISV Application Must Comply with Windows and Microsoft Dynamics AX UI Guidelines

Type	Test Method	Technology	Solution Category

Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

User Interface requirements are part of the requirements review. For X++ code, you should always try to avoid Microsoft Dynamics AX compliance problems rather than trying to correct them. To produce a compliant Microsoft Dynamics AX application, follow these general recommendations:

- Read and understand the user interface specification and checklists.
- Be careful if you need to override existing Microsoft Dynamics AX core functionality.
- Run the Best Practice tool and consider the results carefully.
- Become familiar with the Windows User Experience guidelines and the Microsoft Dynamics AX User Assistance Best Practices Handbook.
- For .NET Framework–based code, you should identify and adhere to a standard set of UI Guidelines.

Resources

See Appendix A for UI guidelines. Use the checkpoint form to make sure that your modules comply with the mandatory requirements and guidelines.

How to Comply

To determine whether your module complies with the checklists, review each checklist, item by item. If you do not have the resources to perform a complete review of your application (for example, if you must verify that there are appropriate screen tips for each text box) you should complete at least two randomly selected checks for each window.

For .NET Framework–based application testing, you must provide the user interface guidelines that you are following. The test team will sample various UI controls and forms to verify that the guidelines are being followed.

In rare cases, you might have to depart from the checklist requirements because of functional requirements or because you must adhere to specific legal requirements or match legal documents. If you believe that this is the case for your Microsoft Dynamics AX application, submit a description of the deviation and explain it carefully in the checkpoint document. Describe the following:

- Why the exception is necessary.
- What you have done to make sure that you cannot accomplish the overall user goal without violating the Microsoft Dynamics AX checklist requirements.

If your module deviates from checklist recommendations, you must provide a justification for these deviations also. However, the acceptance threshold is somewhat reduced for recommendations.

Test Methodology

The test vendor will review your application to verify that it meets the UI and UX requirements. The vendor will visually review and use a representative sample of UI elements (windows, forms, wizards, Help, and so on) to make sure that they look and function according to the guidelines.

Criteria for Passing

This requirement is mandatory. If the ISV application does not follow the mandatory UI guidelines, it will fail the test.

Reporting Requirements

Your application should meet the following reporting recommendations:

- Recommendation 4.1: An X++-based ISV application should follow the reporting guidelines in Microsoft Dynamics AX Help. A .NET Framework–based ISV application should provide documentation for the reporting guidelines used, and should adhere to them.
- Recommendation 4.2: SQL reporting services should adhere to recommended implementation for Microsoft Dynamics AX.

Additionally, Microsoft Dynamics AX–based applications should use the tools and templates provided with Microsoft Dynamics AX.

4.1 An X++ Application Should Follow Microsoft Dynamics AX Reporting Guidelines; .NET Framework–based ISV Applications Should Document the Reporting Guidelines Used

Type	Test Method	Technology	Solution Category		
Recommended	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

Microsoft Dynamics AX provides a report wizard that you must use when you design reports in Microsoft Dynamics AX. This wizard guides the user through the steps for standard reports.

Resources

A white paper that describes the SQL reporting engine in Microsoft Dynamics AX is available at: https://mbs.microsoft.com/partnersource/documentation/whitepapers/reportingandbusinessintelligence.htm?printpage=false

How to Comply

Use the report template system that is built into Microsoft Dynamics AX for any new standard report that you create. If your application is .NET Framework–based, provide a description of the reporting guidelines you used.

Test Methodology

The test vendor will review your application to verify that it meets Microsoft Dynamics AX reporting requirements. The test vendor will randomly test up to 4 reports on the list to ensure that they are deployed via the wizard, and that the user experience is consistent with other reports delivered in the core product.

Criteria for Passing

This is a recommendation only. Failure to comply with this recommendation will not cause the application to automatically fail the test.

4.2 An ISV Application Should Follow the SQL Reporting Services Implementation Guidelines

Type	Test Method	Technology	Solution Category		
Recommended	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

Microsoft Dynamics AX provides implementation guidelines for SQL reporting in Microsoft Dynamics AX applications. These guidelines ensure that all SQL-based reports are surfaced in a consistent manner and will function accurately in Microsoft Dynamics AX.

Resources

A white paper that describes the SQL reporting engine in Microsoft Dynamics AX is available at: https://mbs.microsoft.com/partnersource/documentation/whitepapers/reportingandbusinessintelligence.htm?printpage=false

Description of the SQL reporting engine working with Microsoft Dynamics AX is available at: http://www.microsoft.com/dynamics/ax/product/businessintelligencewp.mspx

How to Comply

Use the SQL reporting implementation guidelines when you design and implement SQL reports for Microsoft Dynamics AX ISV applications. Include documentation that verifies that your report covers the guidelines.

Test Methodology

The test vendor will review your application to verify that it meets SQL reporting implementation requirements.

Criteria for Passing

This is a recommendation only. Failure to comply with this recommendation will not cause the application to automatically fail the test.

Translation and Localization

To simplify globalization, your application must comply with the following requirements:

- Requirement 6.1: An ISV application must follow globalization rules so that it can run in any country without requiring translation. For example, the application must handle date and times, currency, and string formats correctly.
- Requirement 6.2: An ISV application must separate strings (labels) from source code.

Note: The .NET Framework is fully Unicode-enabled and provides extensive built-in globalization support. Microsoft Dynamics AX is Unicode-enabled as of version 4.0.

6.1 An ISV Application Must Follow Globalization Rules

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

To simplify globalization of X++-based applications, Microsoft Dynamics AX 4.0 has built-in data types for time and dates. Additionally, Microsoft Dynamics AX has a built-in label system that ensures that the labels and help text can be easily changed for use in with different languages.

The .NET Framework provides built in support for globalization.

Resources

For more information, see the following:

- Microsoft Global Development and Computing Portal
- MSDN Developer Center: Visual Studio Globalization

How to Comply

You must ensure that the application can be setup using the any local settings like time, currency etc.

Test Methodology

During the in-lab test, the test vendor will perform a qualitative review to determine whether your application follows globalization rules with respect to labels and data types for time and dates. Additionally, the vendor will review your Help to make sure that it can be used with different languages.

For .NET Framework–based applications, the FxCop tool is used to check for compliance.

Criteria for Passing

This requirement is mandatory. If the ISV application does not follow globalization rules, it will fail the test.

6.2 An ISV Application Must Separate Strings from Source Code

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

Localization of business applications requires that you implement the correct business processes and practices for a culture/locale. Differences in how cultures/locales conduct business are heavily shaped by governmental and regulatory requirements. Therefore, localization of business logic can be a massive task.

You should use specialized tools that recycle translations of repeated text and resize application UI elements to allow for localized text and graphics.

Microsoft Dynamics AX has a built-in label system that lets you identify the labels and F1 help text in the ISV application. This system simplifies translation of the application to other languages. Also, the Microsoft Dynamics AX Best Practices tool can determine whether there are any hard-coded strings in your application code. All label files must be translated into the languages where the ISV application is sold.

Resources

For more information, see MSDN Library: Localization Planning.

How to Comply

You must provide an overview of how you plan to localize your application, and you must also provide evidence that strings are separated from code.

Test Methodology

During the in-lab review, the test vendor will run the best practice tool to determine if hard-coded strings are in your source code.

Criteria for Passing

This requirement is mandatory. If the ISV application does not separate strings from source code, it will fail the test.

Technology, Configuration, and Platform Requirements

Your application must meet the following technology and platform requirement:

- **Requirement 7.1**: An ISV application must support the infrastructure that Microsoft Dynamics AX supports.

7.1 An ISV Application Must Support the Infrastructure that Microsoft Dynamics AX Supports

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

Your application must run on the specified infrastructure (browser, database, operating system, and other software) versions that Microsoft Dynamics AX 4.0 runs on, or on later versions. Additionally, you must not require any version of a technology that conflicts with those required for Microsoft Dynamics AX.

Resources

For more information see section “Preparing to install or upgrade”/”Planning hardware and software” in the *Microsoft Dynamics AX Implementation Guide* on PartnerSource at:

https://mbs.microsoft.com/partnersource/documentation/userguides/ax4sp1_help.htm?printpage=false

How to Comply

Test your Setup program on the prescribed infrastructure, and then make sure that your application runs.

Test Methodology

The test vendor will perform a qualitative review to determine whether your application runs on the prescribed architecture (browser, database, operating system, and other required software).

Criteria for Passing

This requirement is mandatory. If the ISV application does not run on the prescribed architecture, it will fail the test.

Setup Requirements

Your application must meet the following installation and removal (uninstall) requirements:

- Requirement 8.1: The ISV application installation procedure must be compatible with Microsoft Dynamics AX.

- Requirement 8.2: The ISV application installation procedure must correctly register DLLs and ActiveX controls.
- Requirement 8.3: The ISV application Setup program must verify that the correct software versions are installed.
- Requirement 8.4: The ISV application Setup program must verify all required Microsoft Dynamics AX components are installed.
- Requirement 8.5: The ISV application Setup program must inherit or create configuration settings.
- Requirement 8.6: The ISV application must include installable demonstration data.

8.1 The ISV Application Installation Procedure Must Be Compatible with Microsoft Dynamics AX

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

You must provide installation instructions, and they must be clear and easy to follow. The installation instructions must include procedures for installing and configuring Microsoft Dynamics AX so that it functions with the ISV add-on application. The instructions should be in the form of a plain text file (they can also be part of the standard user documentation) and must list all necessary steps, including working with the data import and file import, system settings, and instructions for using any automated installation executables.

You must also provide Instructions for uninstalling the ISV application, including removal of any imported code, removal of any DLL or ActiveX components, removal of registry entries, and removal of Microsoft Dynamics AX itself. If it is not possible to uninstall the ISV application, you must state this in your documentation. After the application is removed from the system, the Microsoft Dynamics AX installation on the test bed system might not be functional.

Resources

For more information see the *Microsoft Dynamics AX Implementation Guide* on PartnerSource at:
https://mbs.microsoft.com/partnersource/documentation/userguides/ax4sp1_help.htm?printpage=false

How to Comply

For .NET Framework–based applications, Microsoft recommends the use of the Windows Installer, an application installation and configuration service. The Windows Installer is an operating system component that centrally manages application installation configuration and application uninstall. The Windows Installer lets the operating system manage application setup and configuration, and provides the following benefits:

- It manages reference counting and version checking of shared components.
- It provides a reliable and complete uninstall procedure that includes correct handling of shared components.
- It automatically repairs damaged applications when they are started.
- It uses transaction-based installation: the Windows Installer can roll back to the earlier installed version of the application without error.

Test Methodology

The test vendor will confirm that the ISV has provided a complete list of all resources added to the Microsoft Dynamics AX application. This list will be used to verify the removal of the product.

- Installation: The test vendor will follow each step in the installation instructions in the order presented. The vendor should be able to complete the installation without consulting support personal or contacting the ISV.
- Uninstall: The vendor will follow each step in the removal instructions in the order presented, if un-installation is possible. The vendor should be able to remove the product without consulting the ISV. After removal of the application, the Microsoft Dynamics AX application might not be functional.

The test vendor will review the list of components that you provide to verify that the entire ISV product has been removed.

Criteria for Passing

This requirement is mandatory. If the ISV application installation program does not run on Microsoft Dynamics AX, the application will fail the test.

8.2 The ISV Application Must Correctly Register DLLs and ActiveX Controls

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

The Setup program for both Microsoft Dynamics AX–based and .NET Framework–based applications should record any DLLs and ActiveX components in the registry database of the operating system. The registry serves as a central configuration database for user, application, and computer-specific information.

Resources

For information on requirements for how to build DLLs, see: http://msdn2.microsoft.com/en-us/library/ms694470.aspxv

For information on how to register Dll files see: http://msdn2.microsoft.com/en-us/library/ms859484.aspx

For information on ActiveX controls see: http://msdn.microsoft.com/library/default.asp?url=/workshop/components/activex/activex_node_entry.asp

How to Comply

Check the registry to make sure that your Setup program functions correctly. Document the correct registry settings, and include this information with your application when you submit it for testing.

Test Methodology

During the in-lab test, the test vendor will install your application and review the registry to verify that the Setup program registers all components.

Criteria for Passing

This requirement is mandatory. If the ISV application installation program does not run on Microsoft Dynamics AX, the application will fail the test.

8.3 The ISV Application Setup Program Must Verify That the Correct Software Versions Are Installed

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

Microsoft Dynamics AX tracks software versions by adding version numbering to the **ApplicationVersion** class in the application object tree (AOT). All information about installs upgrade or service pack install are then stored in the Tables SysSetupCompanyLog and SysSetup Log.

Resources

For more information, see the Microsoft Dynamics online Help (press F1) for the ATO elements.

How to Comply

Open the **ApplicationVersion** class methods, and confirm that a method for the version number of your application has been added. Document the method and version number, and include this information with your application when you submit it for testing.

Test Methodology

The test vendor will perform a qualitative review to determine whether the correct software components are installed.

Criteria for Passing

This requirement is mandatory. If the ISV application installation program does add the correct version numbering, the application will fail the test.

8.4 The ISV Application Must Verify That Required Microsoft Dynamics AX Modules Are Installed

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

If your application requires certain Microsoft Dynamics AX modules or other applications to be installed, this should be documented and be clearly specified in your implementation or setup guide.

If you are able through your setup program to detect this prior to the install. You should perform this check early in the sequence of setup activities so that the installation process can be reversed.

Resources

For more information see section “Preparing to install or upgrade”/”Planning hardware and software” in the *Microsoft Dynamics AX Implementation Guide* on PartnerSource at:
https://mbs.microsoft.com/partnersource/documentation/userguides/ax4sp1_help.htm?printpage=false

How to Comply

You must provide the test team with a list of the Microsoft Dynamics AX components that your application requires. The test team will test different configurations with the required components missing to determine whether the ISV Setup program functions correctly.

Test Methodology

The test vendor will perform a qualitative review to determine whether the correct software components are installed.

Criteria for Passing

This requirement is mandatory. If the ISV application installation program does install the correct software components, the application will fail the test.

8.5 The ISV Setup Program Must Inherit or Create Configuration Key Settings

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

The application Setup program must have configuration settings that are compatible with the core Microsoft Dynamics AX product. The settings should inherit, not overwrite, standard configuration settings, or the application should create new settings. If the application changes configuration settings, it should display warning messages.

Microsoft Dynamics AX has a built-in system for switching functionality on and off. You use the **On** and **Off** configurations keys in the Microsoft Dynamics AX configuration form.

To use this form

1. On the **Main** menu, point to **Administration**, point to **Setup**, and then point to **System**.
2. Click **Configuration**.

Resources

See the standard Dynamics AX help by pressing F1 when you are on the Configuration Key form.

How to Comply

Before you submit your application for testing, check the Microsoft Dynamics AX Configuration form to make sure that configuration keys work as required.

Test Methodology

During the in-lab test, the test vendor will review your application to make sure that configuration settings are inherited and that warning messages display for any changed settings.

Criteria for Passing

This requirement is mandatory. If the ISV application installation program does not inherit or create configuration settings, the application will fail the test.

8.6 The ISV Application Must Include Installable Demonstration Data

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

Demonstration data is useful for many purposes, such as sales demonstrations, training, and this test. Therefore, ISVs are required to deliver at least one demonstration data with their application. Microsoft recognizes that some data centers do not permit installation of demonstration data on production servers. For this reason, you can deliver demonstration data as part of the main installation or as a separate installation (for example, VPC). If you include the demonstration data as part of the core installation, you must provide a checkbox to exclude the demonstration data from the process.

Resources

For more information see:
https://mbs.microsoft.com/partnersource/documentation/sampledata/ax40demodata.htm?printpage=false

How to Comply

Use the guidelines in the "Summary and Intent" section, above, to build your demonstration data. Deliver the data with your application as part of the test upload.

The standard Microsoft Dynamics AX database includes demonstration data. Value-added resellers (VARs) use the demonstration data during the sales process to demonstrate the application. They also use the pre-configured data later, during user training.

You should ship the data with the main objects of the application or as a separate DAT/DEF file.

Additionally, you must supply instructions that describe how to add the demonstration data to Microsoft Dynamics AX.

Test Methodology

To verify this requirement, the test vendor will follow these steps:

1. Compile the objects.
2. Follow the instructions supplied by the ISV to install the demonstration data.

Criteria for Passing

This requirement is mandatory. If the ISV application does not include demonstration data, it will fail the test.

Backup and Restore

Your application must meet the following backup and restore requirement:

- **Requirement 9.1**: The ISV must include procedures for backing up and restoring both the application and the data.

9.1 The ISV Must Include Procedures to Back Up and Restore the Application and the Data

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted

			✓	✓	✓

Summary and Intent

A customer or partner must be able to back up and restore your application and all associated data. For a Microsoft Dynamics AX–based application, new fields and tables will be treated the same as any other data and will be backed up as part of the core Microsoft Dynamics AX backup process.

For data that is stored in separate databases (this is most likely to occur in .NET Framework–based applications), you must provide a backup and restore process.

Resources

For more information, see “Upgrading to Dynamics AX” and ”Before you Begin Upgrading” in the Microsoft Dynamics AX Implementation Guide on PartnerSource at:
https://mbs.microsoft.com/partnersource/documentation/userguides/ax4sp1_help.htm?printpage=false

How to Comply

For a .NET Framework–based application, you must provide a process for backing up the application itself.

Test Methodology

During the in-lab test, the test vendor will verify that you have included backup and restore procedures. The test vendor will perform the backup and restore process to make sure that it functions correctly.

Criteria for Passing

This requirement is mandatory. If the ISV application does not include backup and restore procedures, it will fail the test.

Extensibility and Customization Requirements

Your application must meet the following extensibility and customization requirements:

- **Requirement 10.1**: (X++ applications only) The ISV must use the AOT to document how to customize the application.
- **Requirement 10.2**: (.NET Framework–based applications) The ISV must document how to customize the application.

10.1 (X++) The ISV Must Document How to Customize the Application

Type	Test Method	Technology	Solution Category		
Required	In-lab review	X++	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

Regardless of the capabilities of your application, partners or customers must be able to extend or customize it to meet their unique requirements. A customer or partner must have the required information to complete these customizations successfully so that the application continues to function as designed and can be upgraded with minimal impact.

Resources

For more information and to see examples and descriptions of design patterns for Microsoft Dynamics AX, go to: http://msdn2.microsoft.com/en-us/library/ms940503.aspx

How to Comply

Your documentation must include best practices and guidance on how to correctly customize and extend your application.

Test Methodology

During the in-lab test, the test vendor will verify that you have included customization procedures in Help text and in code comments.

Criteria for Passing

This requirement is mandatory. If the ISV does not document how to customize the application, the application will fail the test.

10.2 (.NET Framework) The ISV Must Document How to Customize the Application

Type	Test Method	Technology	Solution Category		
Required	In-lab review	External	Simple	Complex	Hosted
		Managed code only	✓	✓	✓

Summary and Intent

Regardless of the capabilities of your application, partners or customers must be able to extend or customize it to meet their unique requirements. A customer or partner must have the required information to complete these customizations successfully so that the application continues to function as designed and can be upgraded with minimal impact.

Resources

For more information and to see examples and descriptions of design patterns for Microsoft Dynamics AX, go to: http://msdn2.microsoft.com/en-us/library/ms940503.aspx

How to Comply

Your documentation must include best practices and guidance on how to correctly customize and extend your application.

Test Methodology

During the in-lab test, the test vendor will verify that you have included customization procedures in Help text and in code comments.

Criteria for Passing

This requirement is mandatory. If the ISV does not document how to customize the application, the application will fail the test.

Upgrade and Maintenance

Your application must meet the following upgrade and maintenance requirements:

- **Requirement 11.1**: The ISV must document all integration points.
- **Requirement 11.2**: The ISV must provide database upgrade scripts.
- **Requirement 11.3**: The ISV must use file versioning for DLLs and ActiveX controls**.**

11.1 The ISV Must Document All Integration Points

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

Almost every line-of-business application interoperates with components outside the core Microsoft Dynamics platform. This interoperability may be in the form of AIF, COM integration, .NET interop, Web Services, BizTalk, or even SQL Server–generated messages.

Resources

For more information, see the documentation example for AIF integration with standard Microsoft Dynamics AX, available on MSDN at: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/AxITPro/html/576efdc3-8efc-48c5-bc4a-99308190c9fa.asp

How to Comply

You should ship integration point documentation as part of your submission as well as document it during your TWC documentation.

Test Methodology

The test vendor will review documentation to determine that the ISV has documented the integration points.

Criteria for Passing

This requirement is mandatory. If the ISV does not document the integration points, the application will fail the test.

11.2 The ISV Must Provide Database Upgrade Scripts

Type	Test Method	Technology	Solution Category		
Required*	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

**Required if the ISV application has been updated.*

Summary and Intent

Upgrades are an important part of business software; the intent of this section is to make certain upgrades are relatively easier for partners by providing scripts that support database upgrades.

When new versions, service packs, and hot fixes are released, Microsoft Dynamics AX uses upgrade scripts to handle data upgrades. Similarly, you must provide upgrade scripts for your application if you change the data model structure.

Before you submit your application for testing, install your application, identify any changes to the database model, and compare the upgrade scripts with the data model changes to make sure that you have provided accurate upgrade scripts for all changes to the application.

Resources

See the white paper on how to write upgrade scripts at:
https://mbs.microsoft.com/partnersource/documentation/whitepapers/msdyax40_writing_upgrade_scripts_whitepaper.htm?printpage=false

How to Comply

Before you submit your application for certification, install the application, identify any changes to the database model, and compare the upgrade scripts with the data model changes to make sure that your have accurate upgrade scripts for all changes to the application.

When you submit your application for certification, include the upgrade scripts.

Test Methodology

During the in-lab test, the test vendor will run your upgrade scripts and then test the application against the database to verify that the scripts function correctly.

Criteria for Passing

This requirement is mandatory. If the ISV does not provide database upgrade scripts, the application will fail the test.

11.3 The ISV Must Use File Versioning for DLLs and ActiveX Controls

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

All executable files, such as DLLs and ActiveX controls, must have versioning metadata associated with them. Installation programs use this metadata to confirm that correct versions are in place before applying an upgrade, service pack, or hot fix. Without this versioning information, an installation program could corrupt the system by applying changes that cannot synchronize with the file that is currently installed. Additionally, if a shared component fails, correct file version information lets a customer identify the associated application and file producer. The file's producer is the only entity that can regress the file; therefore, the metadata must also include the company name.

Resources

All DLLs and ActiveX controls must be file versioned. See requirement 13.2 for more information on DLLs and ActiveX controls.

How to Comply

Before you submit your application for testing, examine the file information for each DLL and ActiveX control to verify that it includes the product name, company name, and file version number.

Test Methodology

Review submitted code to determine if files contain the metadata with the following elements:

- Product name
- Company Name
- File Version Number

Criteria for Passing

This requirement is mandatory. If the ISV does not use file versioning, the application will fail the test.

Sustainability

Your application must meet the following sustainability requirements:

- **Requirement 12.1**: The ISV must remove non-functioning code from the code base.

12.1 The ISV Must Remove Non-Functioning Code from Code Base

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

Non-functioning code is confusing to support engineers who are attempting to resolve a problem and to developers who may work on your application in the future. In addition, there is always the risk, however remote, that code artifacts can be started accidentally by another application or process.

Resources

See the development requirement section on how to use the Best Practices tool. This tool identifies non-functioning code.

How to Comply

It is a good practice to remove all non-functioning code from an application before the application is released. Your test plans and test scripts should check for non-functioning code.

Test Methodology

During the in-lab review, the test vendor will run the best practice tool to determine if non-functioning code has been removed.

Criteria for Passing

This requirement is mandatory. If the application contains non-functioning code, it will fail the test.

Best Practice Guidelines

The following best practices are strongly recommended for use by ISVs, but they are not part of the test process.

Trustworthy Computing Requirements

Your application should comply with the following trustworthy computing requirements:

- Requirement 5.1: Before the development begins, development staff should complete security and Security Development Life cycle (SDL) training.
- Requirement 5.2: During the implementation phase, the ISV should establish and follow security development best practices.

To comply with trustworthy computing requirements and recommendations, follow the guidelines in Microsoft Dynamics AX Help, particularly the "Development Best Practices" section.

BP 1.1 Before Development Begins, ISV Staff Should Complete Security and SDL Training

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

Microsoft has adopted a process called the *Trustworthy Computing Security Development Life Cycle (SDL)* to help make sure that software development follows security best practices. Security best practices require that developers are aware of secure coding practices including threats and countermeasures, and that they use the most recent versions of the Microsoft Visual Studio® development system, the .NET Framework, ADO.NET, and other tools and technologies. These tools provide the greatest compatibility with the platform infrastructure and provide the latest security advancements. For example, if your developers use Visual C++, they should use the **/GS** switch to compile their code and the **/RTC1** switch to debug their code.

In addition, SDL requires that developers should fix critical bugs that compromise security and perform security code reviews based on guidelines and checklists described in Appendix D of *Writing Secure Code* by Michael Howard and David C. LeBlanc. These reviews help ensure that the software design meets minimal trustworthy computing standards.

Resources

For more information, see the following:

- Microsoft Dynamics AX - Writing Secure X++ Code white paper
- Microsoft Trustworthy Computing Web site
- Howard, Michael and David C. LeBlanc. *Writing Secure Code*. Second Edition. Redmond, WA: Microsoft Press, 2002
- Microsoft Press: *Writing Secure Code Companion Content*

How to Comply

To satisfy the education requirement, all developers should complete the following:

- Read Writing Secure Code, Second Edition, by Michael Howard
- Read the Microsoft Dynamics AX - Writing Secure X++ Code white paper
- Complete the following two Microsoft eLearning security courses:
- Clinic 2806: Microsoft Security Guidance Training for Developers
- Clinic 2807: Microsoft Security Guidance Training for Developers II

To verify that your staff has satisfied this requirement, prepare a checklist or training documentation (such as a training overview, a Microsoft PowerPoint® presentation, class handouts, or syllabus), and include these items in your test submission package.

Test Methodology

The test vendor will review your training documentation to verify that your staff has completed the appropriate training. The vendor will use a score card similar to the following table.

	Company	Developer Name	*Date Completed*
Writing Secure Code			
Clinic 2806			
Clinic 2807			
Other			

Criteria for Passing

This requirement is mandatory. If the ISV application does not follow security guidelines, it will fail the test.

BP 1.2 The ISV Should Establish and Follow Secure Development Best Practices

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

Your application development team should establish, communicate, and follow specific best practices for developing secure code. This approach will help them detect and remove security defects as early as possible in the development process. There are a number of resources, tools, and processes that you can use to accomplish this goal. The time and effort you take to apply best practices early in the product development cycle can prevent a more costly response to security defects later in the development cycle, or even more painful responses after product release.

Resources

For more information, see the following:

- MSDN Library: Creating Named Shared Memory
- Microsoft Dynamics AX - Writing Secure X++ Code white paper

How to Comply

For .NET Framework–based applications, follow these guidelines:

- C# compiler: Use the latest version of the Microsoft Visual Studio® .NET development system. Use the specified compiler and linker flags for .NET Framework–based applications.
- csc.exe: Use version 8.0.50727.42.
- .NET Framework: Use version 2.0.50727.
- FxCop: Use the latest version of FxCop to locate and fix all security-related issues.
- Follow the guidelines in the SDL Shared section (that is, have no writable PE shared sections).

For X++ applications, mitigate the use of dangerous X++ APIs.

Establish the following best practices as you develop your application:

- Build tools: Use the currently required (or later) versions of compilers and compile options for your development platform.
- Code analysis tools: Use the currently required (or later) versions of code analysis tools for native (C and C++) or managed (C#) code.
- Previously released applications that were compiled with the **/GS** switch using Visual Studio version 2003 or later: You should research and resolve and **/GS** switch crashes. System crashes produced by binaries compiled with the **GS** option are marked in a specific way when uploaded through the Watson service. The data uploaded to Microsoft indicates where the buffer overflow occurred in the Microsoft code and also confirms that it happened in a customer use scenario. Because the **GS** compiler option mitigates some types of buffer overrun exploits, but not all of them, it is very important that you fix any reported buffer overruns in your code. For more information, see *Writing Secure Code, Second Edition (*Howard*)* and SDL Buffer Overflow Watson Crash Requirements, which outlines the process and requirements for servicing **GS** crashes.
- Banned APIs: New native code (C and C++) should not use banned versions of string buffer handling functions. Based on analysis of previous MSRC cases, you should avoid the use of certain APIs to reduce vulnerability. See *Writing Secure Code, Second Edition (*Howard*)* for more information.
- Executable pages: Avoid using executable pages. Unless your technology requires executable pages (for example, just-in-time [JIT] compilers), do not use them. This requirement affects how you call the **VirtualAlloc()** APIs and also prohibits compiling with **/NXCOMPAT:NO**. This requirement applies to Win32 and Win64 applications only; it does not apply to WinCE.)
- Writable shared PE sections: Your shipped binaries should not contain sections marked as **shared**. Shared binaries are a potential security threat. Use properly secured dynamically created shared memory objects instead.

Test Methodology

The test vendor will verify that your application meets the best practices requirements, as follows:

- Tools and processes: The vendor will perform a qualitative review to make sure that required processes and tools are in place and that processes were successfully executed. (The vendor will not audit the output of your tools and processes.)
- FxCop: The vendor will run FxCop and will require that you fix all reported issues.
- .NET Framework SDL Shared Section requirements (no writable PE shared sections): The vendor will run tools and require that you fix all reported issues.

- Dangerous X++ APIs: The vendor will perform an audit of a representative sample of your code to make sure that all trustworthy computing errors are fixed.

Criteria for Passing

This requirement is mandatory. If the ISV application does not follow security guidelines, it will fail the test.

Platform requirement

BP 2.1 An ISV Application Must Run on Different Language-specific Platforms

Type	Test Method	Technology	Solution Category		
Required	In-lab review	All	Simple	Complex	Hosted
			✓	✓	✓

Summary and Intent

The ISV application must be able to run on different language versions of the Microsoft Windows® operating system and on different language versions of databases.

Resources

See standard Microsoft Windows and Microsoft SQL 2005 documentation on this topic.

How to Comply

You must ensure to test that the application can run on different language versions of the operating system and different versions of the Microsoft SQL Server 2005 database.

Test Methodology

During the in-lab test, the test vendor will perform a qualitative review to determine whether your application follows this rule by running your application on top of different language versions of Microsoft Windows and Microsoft SQL 2005.

Criteria for Passing

This requirement is mandatory. If the ISV application does not run on different language platforms, it will fail the test.

Appendix A: UI Guidelines

The following best practices are divided into two; recommended and mandatory for use by ISVs. Each Best practice have a number followed by 'recommended' or 'mandatory'. An ISV must follow 'mandatory' requirements or make a descriptive documentation of reason for not complying for a given requirement. It is recommended that an ISV follow the recommended requirement as well.

Requirements for Application Windows

The windows in a Microsoft Dynamics AX application must comply with the following requirements:

- UI 1.1 – recommended: Do not change the window size when your application opens a new window. For example: the window size must not change if a user changes from table view to tree view. (A user can use the standard Microsoft Windows operating system tools and conventions to resize windows.)
- UI 1.2 - recommended: Make sure that application windows fit into an SVGA display with 800 x 600 pixels.
- UI 1.3 - recommended: By default, do not require horizontal scrolling for application windows.
- UI 1.4 - recommended: Do not use absolute coordinates to specify the size and location of a new window. When a window opens it must automatically size itself within the open application window, without exceeding preset size limits for windows.
- UI 1.5 – recommended: You should not specify window positions in absolute coordinates.

Navigation Pane Requirements

The Navigation pane in a Microsoft Dynamics AX application must comply with the following requirements:

- UI 1.6 - recommended: The Navigation pane must display in a specific percentage of the application window. If the user resizes the window, the navigation pane must be resized automatically so that the percentage remains constant.
- UI 1.7 - recommended: The Navigation pane must have the following two sections:
  - Favorites
  - Main menu

Favorites Requirements

Microsoft Dynamics AX does not include any Favorites in the Navigation pane. However, you can add them to create a set of menus that are more appropriate for the users of your application. Items that appear on the Favorites section of the Navigation pane must meet the following requirements:

- UI 1.8 - recommended: Include a maximum of 7 items in the Favorites section. While a few carefully selected Favorites can help a user to navigate effectively through an application or a set of tasks, too many Favorites can be overwhelming.
- UI 1.9 - recommended: Do not use more than two nested (indentation) levels.
- UI 1.10 - recommended: Group Favorites by type of task. For example: "Phone sales" or "Reconcile payments."
- UI 1.11 - recommended: Use clear, concise, easy-to-understand names for Favorites. Do not use cryptic abbreviations or jargon.

Main Menu Requirements

Items that appear on the Main menu section of the navigation pane must comply with the following requirements:

- UI 2.1 - recommended: At the top of the Main menu, include links to no more than six forms (that is, links to forms that are not located in a folder). For example, in Figure 3.1 the following three forms are at the top of the menu: **Chart of accounts**, **Ledger budget**, and **Fixed assets**.
- UI 2.2 - recommended: If you include links to forms at the top of the Main menu, make sure that the linked forms are the most frequently used forms in the application.
- UI 2.3 - mandatory: Use the following names for the top-level folders: **Journals**, **Inquiries**, **Reports**, **Periodic**, and **Setup**.
- UI 2.4 - mandatory: Put the top-level folder names in this order: **Journals**, **Inquiries**, **Reports**, **Periodic**, and **Setup**.
- UI 2.5 - recommended: Put no more than six forms in a folder.
- UI 2.6 - recommended: On the main menu, use no more than five nested (indentation) levels.
- UI 2.7 - mandatory: Use the following definitions for the main menu icons:

   Folder.

   Open a form.

   Produce a report.

   Perform an action.
- UI 2.8 - recommended: If a user clicks a form icon on the main menu, make sure that the same form icon appears in the upper-left corner of the new window (for an example, see below).
- UI 2.9 - recommended: If a user clicks a form icon on the main menu, use the form icon label as the prefix for the new window name. For example, if a user clicks a form icon named **Inventory to fixed assets** in the **Journals** folder, the name of the new window must be **Inventory to fixed assets** or **Inventory to fixed assets–Journal**, followed by some additional information that is useful to the user. Window names such as **Journal** or **Inventory journal** are not permitted.
- UI 2.10 - mandatory: If a user single-clicks a form icon, the application must open the corresponding form.
- UI 2.11 - mandatory: If a user double-clicks a forms icon, the application must not open two identical forms.

Figure A.1 is an example of the Microsoft Dynamics AX Main menu.

Use standard names and order for level 1 folders (rule 2.3).

Use output form icons and labels (for example, **Customer**). Single-clicking the icon opens a window with the title **Customer - Reconciliation** (rule 2.9).

Link to the six (maximum) most frequently used forms (rules 2.1 and 2.2).

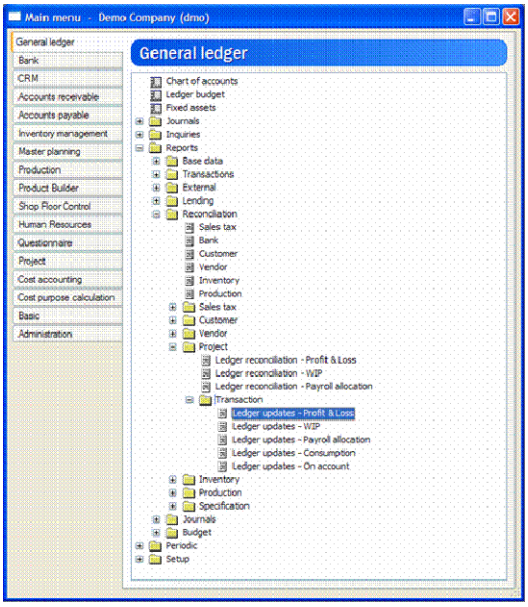


Figure A.1
Microsoft Dynamics AX Main menu

Task Pane Requirements

The Task pane appears at the right side of the Microsoft Dynamics AX application window. You cannot resize the width of the Task pane programmatically. However, users can resize the pane in the application. The Task pane reduces the space available to the Contents pane; therefore, you should use the Task pane to display information that is useful to the user but not critical to performing the task at hand. For example, you could use the Task pane to display the following types of information:

- Search results
- Help content
- Information loaded from another source, such as stock quotes, inventory status, and so on

Items that appear in the Task pane must meet the following requirements:

- UI 3.1 - recommended: Use the Task pane to present information that is useful but not required. Do not use the Task pane to display notifications or information that is critically important to the user's task.
- UI 3.2 - mandatory: Do not use the Task pane as a dialog box or for setting options.
- UI 3.3 - mandatory: Do not use the Task pane as a secondary workspace that competes with the Content pane. The Content pane is the primary workspace for displaying forms and windows associated with application tasks.
- UI 3.4 - recommended: Do not include so much information in the Task pane that you confuse or overwhelm the user.

- UI 3.5 - recommended: Do use the Task pane to display content that is specific to the currently selected form or window. The Task pane should display information that applies to the application in general or that is peripherally useful.
- UI 3.6 - recommended: Consider using a Wizard or alternative content format to present information or guidance.

Forms Requirements

The forms in a Microsoft Dynamics AX application must comply with the following requirements:

- UI 3.7: Use the following form types:
  - **Form type 1** - mandatory: a form with two tab controls. The upper tab control (the heading tab control) must contain a table that displays objects: for example, a list of sales orders. The lower tab control (the line tab control) displays lines for the currently selected object in the upper tab control. See Figure 3.2 for an example.
  - **Form type 2** - mandatory: a form with one tab control (a heading tab control), which must contain a table. See Figure 3.3 for an example.
  - **Form type 3 -** mandatory: a form with a function tab control and no table. See Figure 3.4 for an example. This form type is called a *function window*. For more information about function windows, see the topic, “Function Window Requirements,” in this chapter. You must use a tab control even if the form contains only one tab.
- UI 3.8 - recommended: If a form contains two tab controls, use a movable line (a *splitter*) to separate the controls, as shown in Figure 3.2.
- UI 3.9 - recommended: Leave the area above a table empty or use it for controls that apply frequently used filters or presentation formats to the records displayed in the table. See Figure 3.2 for an example of a blank area. See Figure 3.3 for an example that displays frequently used filters.
- UI 3.10 - mandatory: Do not use group headings in filter controls.
- UI 3.11 - mandatory: Do not provide filtering criteria for a table in a tab contained in the same tab control. For example, if you have a tab control that contains a table, do not also include a **Filter** tab that controls the filtering of the table.
- UI 3.12 - recommended: Leave inactive filter criteria blank. For example, do not use a check box to enable or disable a filter control.
- UI 3.13 - recommended: Leave the area underneath a table blank or use it to contain one of the following:
  - Row extension controls.
  - Relevant summaries of the table; for example, column totals.

  Row extension controls contain information that belongs to the currently selected row. Use row extension controls only if horizontal scrolling would otherwise be required to show all relevant fields and for fields where vertical scanning of the corresponding column is not appropriate. See Figure 3.5 for an example of row extension controls.
- UI 3.14 - recommended: Use a column total for the total of the whole column in the database. If the form uses scrolling, do not use the column total to show the total for only the currently visible part of the column.
- UI 3.15 - recommended: By default, do not require horizontal scrolling for a table.

- UI 3.16 - mandatory: Do not use **OK**, **Cancel**, or **Help** buttons in forms types 1 and 2.
- UI 3.17 - recommended: On forms, put buttons on the right side of a column.

Figure A.2 is an example of form type 1. It is a sales order form with two tab controls: a heading tab control and a line tab control. Each tab control has one table. All buttons are on the right side of the columns.

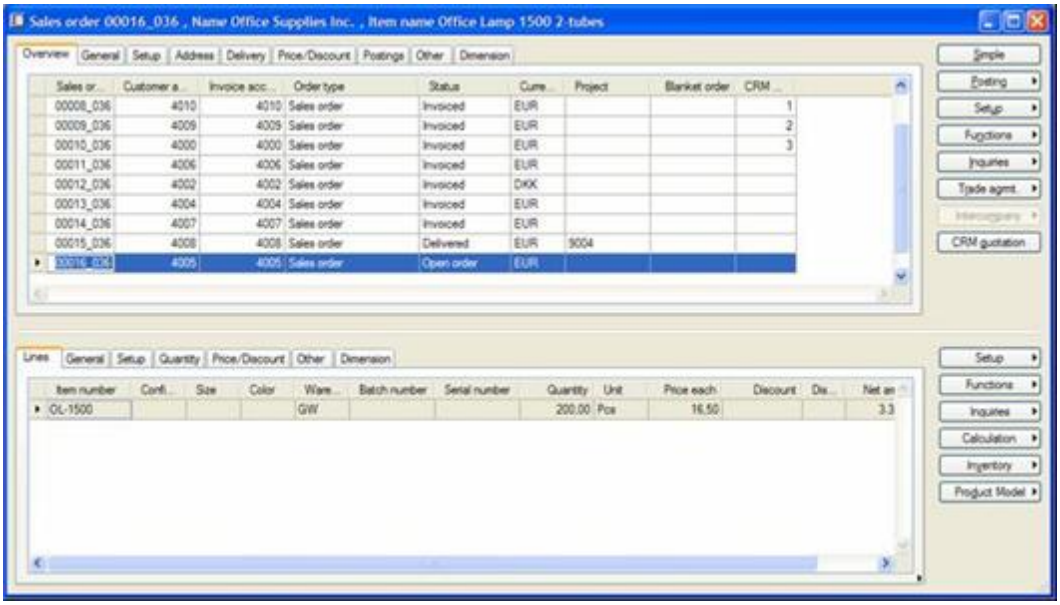
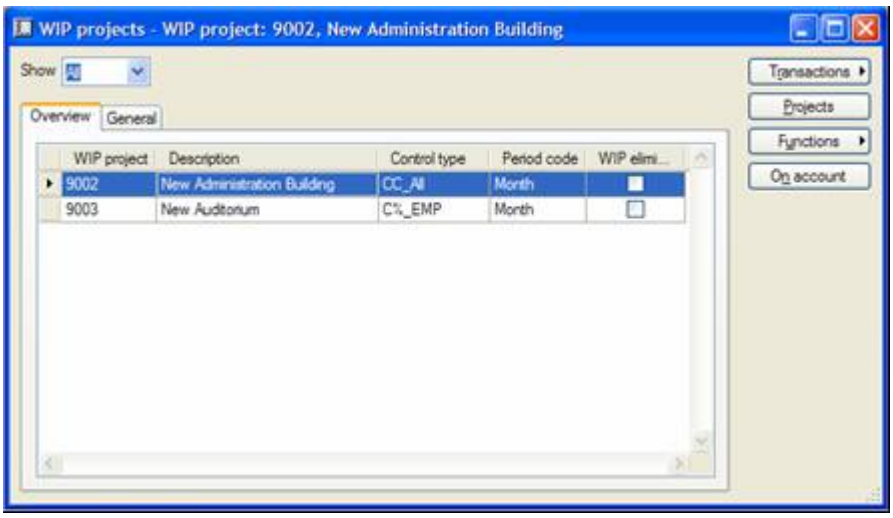


Figure A.2
Form type 1 example

Figure A.3 is an example of form type 2. It is a work-in-progress form with a heading tab control and one table.



Filter criteria for table (rule 3.3).

Figure A.3
Form type 2 example

Figure A.4 is an example of Form type 3. It is a function window that has one function tab control without a table. The example is a general ledger window used for reporting sales tax codes.

Figure A.4
Form type 3 example

Figure A.5 is an example of a general ledger account transactions form with row extension controls.

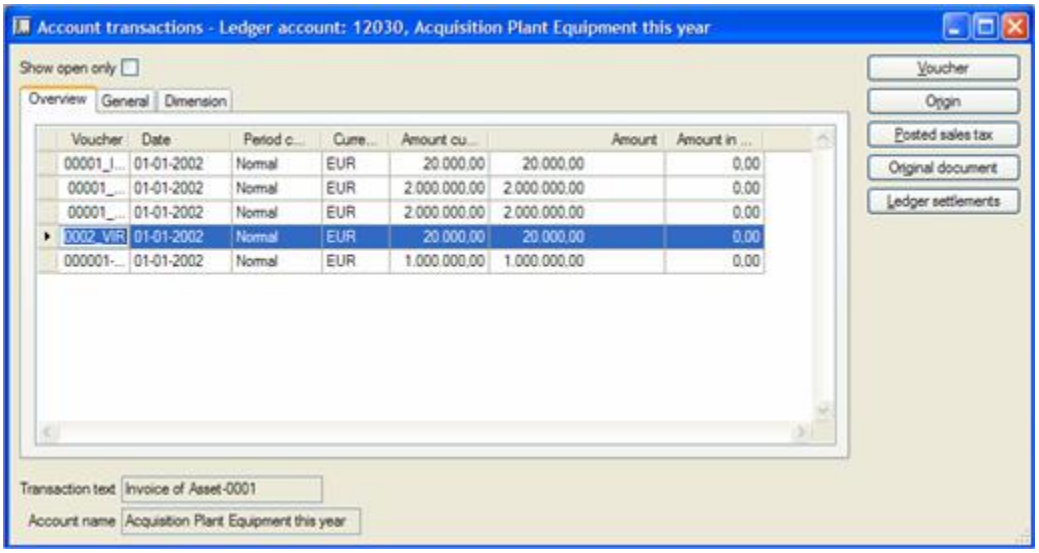


Figure A.5
Row extension controls example

Edit Control Requirements

The edit controls in a Microsoft Dynamics AX application must comply with the following requirements.

Note: The following rules for edit controls also apply to read-only fields. This is because a read-only field is an edit control where editing has been blocked.

- UI 4.1 - mandatory: Do not override the default edit control labels if the new value is identical or similar to the Microsoft Dynamics AX standard label.
- UI 4.2 - mandatory: Do not use fixed labels. To help with localization, obtain all static text in a window (headings, instructions, edit control labels, and so on) from the Microsoft Dynamics AX database.
- UI 4.3 - recommended: Do not use skip fields. The user must be able to modify all edit controls, or the fields must be read-only.
- UI 4.4 - recommended: Do not use dimmed text in edit control labels.
- UI 4.5 - recommended: Use ordinary text in text boxes. Do not use dimmed text, bold, italic, and so on.
- UI 4.6 - recommended: Use positive statements for check box labels. Do not use negative statements because some users cannot understand the implied double negative that occurs when a negatively labeled check box is unchecked. For example, do not use the label **Don't tell me again** for a check box. Instead, use the label **Display this message in the future**, and change the default value to **false** (unchecked).
- UI 4.7 - recommended: Do not use check boxes that contain mutually exclusive options. Instead, use drop-down lists or radio buttons.
- UI 4.8 - recommended: Use drop-down lists instead of radio buttons for mutually exclusive choices.

Button Requirements

The buttons in a Microsoft Dynamics AX application must comply with the following requirements.

- UI 5.1 - mandatory: Make sure that a user receives visible feedback when he or she clicks a button.
- UI 5.2 - recommended: If a button opens a new window, use the button label as the prefix for the title of the new window. For example, if a user clicks a form icon button named Inventory to fixed assets in the Journals folder, the title of the new window must be Inventory to fixed assets or Inventory to fixed assets–Journals, followed by some additional information that is useful to the user. Window names such as Journal or Journals or Inventory journal are not permitted.
- UI 5.3 - recommended: In forms, put command buttons on the right side of the window, as shown in Figure 3.6.
- UI 5.4 - mandatory: If you use buttons for Transactions, Setup, Functions, and Inquiries, use exactly these names for the buttons.
- UI 5.5 - mandatory: If you use Transactions, Setup, Functions, and Inquiries buttons, put the buttons in exactly this order.
- UI 5.6 - recommended: If you include a Transactions button, make sure that it is the first button.
- UI 5.7 - recommended: Do not use command buttons if you have only one menu item. Instead, make sure that the module performs the only possible action as soon as the user clicks the button.
- UI 5.8 - recommended: Use text for command button labels. Do not use pictures or symbols.

Exception: The Help system uses buttons that do not follow this rule.

- UI 5.9 - recommended: Make sure that users only have to click a button one time to perform the associated action.
- UI 5.10 - recommended: Use icon buttons in toolbars only. Put them immediately to the right of a text box as shown in Figure A.6.

Figure A.6 is an example of a customer contact form with command and icon buttons.

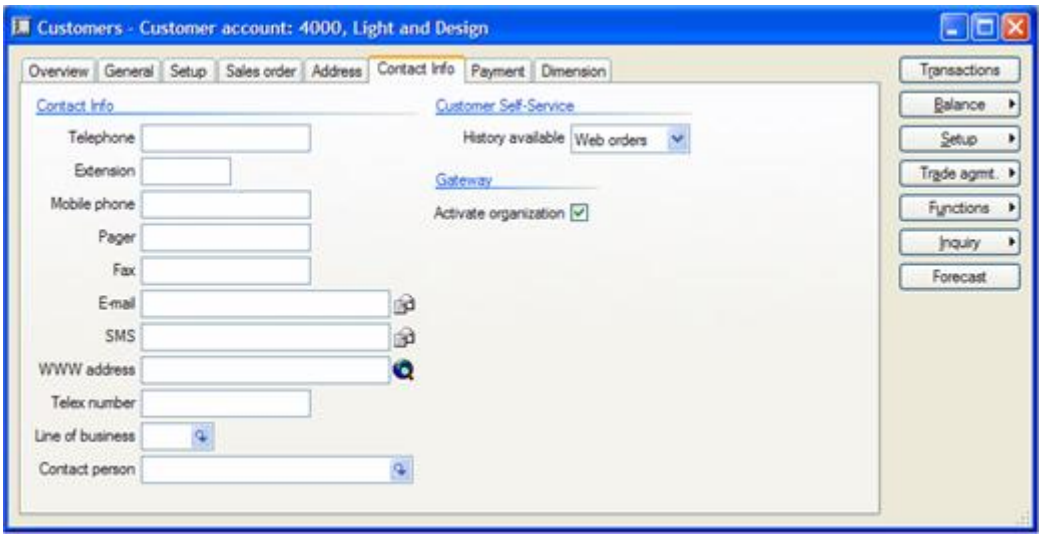


Figure A.6
Command and icon button example

Requirements for Other Controls and Toolbars

Other controls and toolbars in a Microsoft Dynamics AX application must comply with the following requirements.

- UI 6.1 - recommended: Use consistent names and organization. For example, when a user selects a category named **Sales and quotations** from a drop-down menu, the corresponding controls must be under that group heading and not under the group heading **Quotation period**.
- UI 6.2 - recommended: Use consistent names in all modules. For example, do not name a button **Inquiries** in one module and **Inquiry** in another module.
- UI 6.3 - recommended: Use TAB sequences that are natural from the user's point of view. The TAB sequence is the order in which the pointer travels from control to control when a user presses the TAB key. The natural order is usually column to column.
- UI 6.4 - recommended: Use dimmed text only to indicate unavailable buttons or menu items.
- UI 6.5 - recommended: Do not use hidden controls. Deactivate them if they are not relevant in the current context.
- UI 6. - recommended: You should not use toolbars that duplicate functions that are available in the standard Microsoft Dynamics AX toolbar.

Tab Requirements

Tabs in a Microsoft Dynamics AX application must comply with the following requirements.

- UI 7.1 - mandatory: In a heading tab control, name the first tab **Overview**.
- UI 7.2 - recommended: Display the **Overview** tab when a tab control opens.
- UI 7.3 - mandatory: Use the **Overview** tab to hold a list of records for a particular form.
- UI 7.4 - recommended: In a line tab control, name the first tab **Line**.
- UI 7.5 - mandatory: Use the **Line** tab to hold a list of lines for the record selected.
- UI 7.6 - recommended: In a line tab control, name the second tab **General**.
- UI 7.7 - recommended: In a function tab control, name the first tab **General**.
- UI 7.8 - recommended: Use the **General** tab for the most frequently used controls in the form. For a heading or a line tab control, these controls correspond to the record selected in the **Overview** tab or the line selected in the **Line** tab.
- UI 7.9 - recommended: If you use a setup tab, it must be the third tab.
- UI 7.10 - recommended: If you use a setup tab, you must name the tab **Setup**.
- UI 7.11 - recommended: If you use a dimension tab, it must be the last tab.
- UI 7.12 - recommended: If you use a dimension tab, you must name the tab **Dimension**.
- UI 7.13 - recommended: Use the **Dimension** tab for controls that let users define default financial options for the currently selected object, such as a preferred account or a cost center for the currently selected part.
- UI 7.14 - recommended: Carefully maintain the tab metaphor. For example, do not add, hide, or change tab controls and labels because of the value of certain input fields on that tab or another tab in the same tab control or because of the value of a record type. However, you can enable and disable controls depending on the record type.
- UI 7.15 - recommended: Do not put tab controls on tabs.
- UI 7.16 - recommended: Put the main key in the upper-left corner of the **General** tab.
- UI 7.17 - recommended: Do not change the number of tabs when a control starts. For example, do not use an **Advanced** check box that changes the number of visible tabs. However, users can hide tabs by using standard Microsoft Dynamics AX conventions.
- UI 7.18 – recommended: For performance reasons, a tab control should not relate to more than one or two database tables. If you need more database tables, separate the tab control into several forms.

Table Requirements

In Microsoft Dynamics AX, tables are frequently referred to as *grids*. This chapter uses the term *tables* because the Windows User Experience Requirements document uses this term.

Tables in a Microsoft Dynamics AX application must comply with the following requirements.

- UI 8.1 - recommended: Use tables on **Overview** or **Line** tabs only. Do not use tables in function windows or on any other kind of tab. This means that a tab control can contain at most one table.
- UI 8.2 - recommended: Give headings to all columns, including columns that contain only icons.
- UI 8.3 - recommended: Make sure that users can change the width of a column, including any column that contains only icons.

- UI 8.4 - recommended: Make sure that users can sort a table according to a particular column, including a column that contains only icons.

  Exception: Sorting on a display column (a column with derived data) is not required.

- UI 8.5 - mandatory: As a default, put the main key for a table in the left column.
- UI 8.6 - recommended: As a default, put the main key in clear text in the second column.
- UI 8.7 - recommended: As a default, put the fields that are most important to users in the **Overview** tab to the extent that space allows (see the previous rules about horizontal scrolling). These fields usually include the mandatory fields.
- UI 8.8 - recommended: Do not use radio controls in tables.

Tree View Requirements

Tree views in a Microsoft Dynamics AX application must comply with the following requirements.

- UI 9.1 - recommended: Provide a tree view in your application. See Figure A.7 for an example.
- UI 9.2 - recommended: In each line in the tree view, use an icon followed by relevant information about the object.
- UI 9.3 - recommended: You can provide the information about an object in a character-delimited list. If you use a character-delimited list, make sure that the first two items in the list are the object key and the object name.

  Exception: Do not use a character-delimited list in the main menu, even though it is a tree view.

- UI 9.4 - recommended: If you use a character-delimited list, use a backslash (preferred), comma, or slash to separate items in the list.
- UI 9.5 - recommended: If users can select more than two items from a character-delimited list, use a list on a tab to present the selections. Name this tab **Setup**.
- UI 9.6 - recommended: Where relevant, provide drag-and-drop functionality for repositioning, adding, and removing objects in the tree view.
- UI 9.7 - recommended: Provide an easy to find and intuitive explanation of the icons used in the tree view. Put the explanation on the **Overview** (tree) tab or on the **Setup** tab.
- UI 9.8 - recommended: Make sure that the application does not resize the form if the user selects a tree view.
- UI 9.9 - recommended: If a user resizes a window that contains a tree view, make sure that the tree view is resized also.
- UI 9.10 - mandatory: If you provide drag-and-drop functionality in a form, describe the drag-and-drop feature in the form Help.
- UI 9.11 – recommended: You should use a backslash as the separator between items in a character-delimited list.

Figure A.7 is an example of a tree view for a project tree structure.

Note: The tree view form shown in Figure A.7 does not fully comply with some of the rules in this document. If a user selects a tree view, the name of the left tab changes from **Overview** to **Tree**. The name should remain **Overview**. If the user selects or clears the **Tree control** check box, the window size changes. The window size should not change.

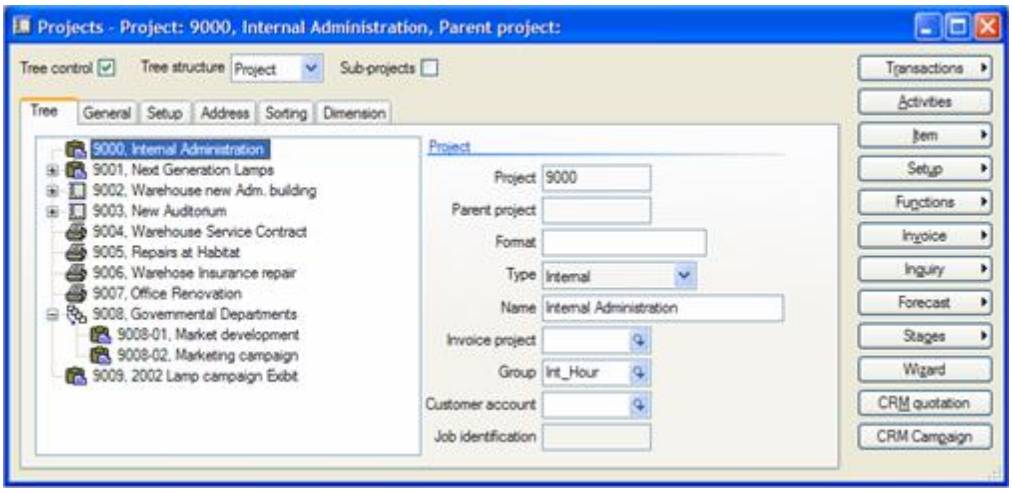


Figure A.7
Tree view example

Function Window Requirements

Function windows in a Microsoft Dynamics AX application must comply with the following requirements.

- UI 10.1 - recommended: Do not use tables in function windows.
- UI 10.2 - recommended: If a function window starts a process, make sure that it contains an **OK** button and a **Cancel** button.
- UI 10.3 - recommended: In a function window, make sure that the **OK** and **Cancel** buttons are the right-most buttons on the last line in the window.
- UI 10.4 - recommended: If you have additional command buttons to the left of the **OK** and **Cancel** buttons, separate them from the **OK** and **Cancel** buttons.
- UI 10.5 - recommended: Make sure that OK and Cancel buttons have exactly these labels. Do not use variant label forms such as Ok or OK or Cancel.

Figure A.8 is an example of a function window for a **General ledger–Periodic–Exchange adjustment** form.

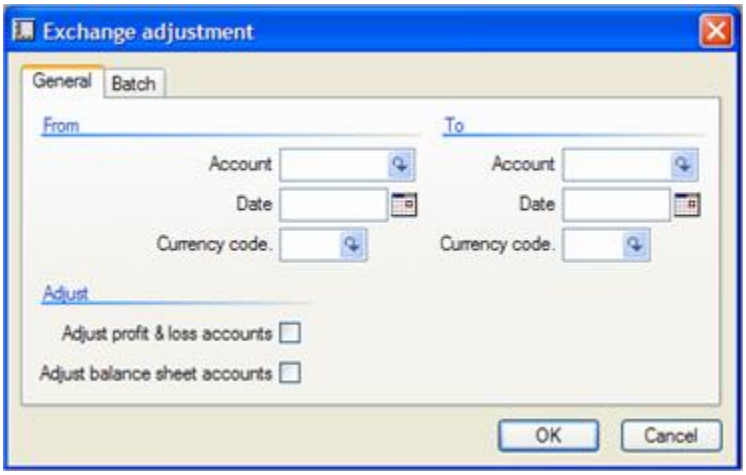


Figure A.8

Function window

Requirements for Icons and Symbols

Icons and symbols in a Microsoft Dynamics AX application must comply with the following requirements.

- UI 11.1 - recommended: Make sure that a typical user can understand all icons and symbols.
- UI 11.2 - recommended: Make sure that the icon style complies with Microsoft Office guidelines.
- UI 11.3 - recommended: For each icon, include a screen tip that explains the purpose of the icon.
- UI 11.4 - recommended: Use color in icons and symbols. Do not use icons that contain only black, gray, and white.
- UI 11.5 - recommended: Use a transparent background for the icons.
- UI 11.6 - recommended: Do not use text or characters in icons and symbols because such icons and symbols are difficult to localize.
- UI 11.7 - recommended: Do not use blinking icons or symbols.

User Assistance UI Requirements

Microsoft Dynamics AX supports the following user assistance formats:

- **Procedural Help**. Procedural Help appears in a separate Help window when users explicitly request it; for example, by pressing **F1** and then **Contents**. Procedural Help provides information about how to complete a user task. Procedural Help takes the user's point of view, and should use second person singular (for example, "You do this") or imperative (for example, "Do this…") voice. Other types of Help are informational, and can use the third person (for example, "The system does this…" or "The users did that").
- **Form Help**. Form Help appears in a separate Help window when a user explicitly requests it (for example, by pressing **F1**). Form Help provides general information about a form (window). In contrast, screen tips, tool tips, and status bar messages provide detailed information about a single screen object.
- **Screen tips**. A screen tip appears next to a text box when a user explicitly requests it; for example by clicking the text box, clicking the right mouse button, and selecting **What's This?** on the properties menu or pressing **Shift+F1**. Windows User Experience refers to screen tips as *contextual help*.
- **Tool tips**. A tool tip appears next to a command button (including a toolbar button) after a brief delay when a user uses the mouse to hover over the button.
- **Explicit Help**. Explicit Help is an explanation of the currently selected object that automatically appears above a table.
- **Status bar messages**. A status bar message automatically appears in the left side of the status bar (in the lower-left corner of the Microsoft Dynamics AX window).

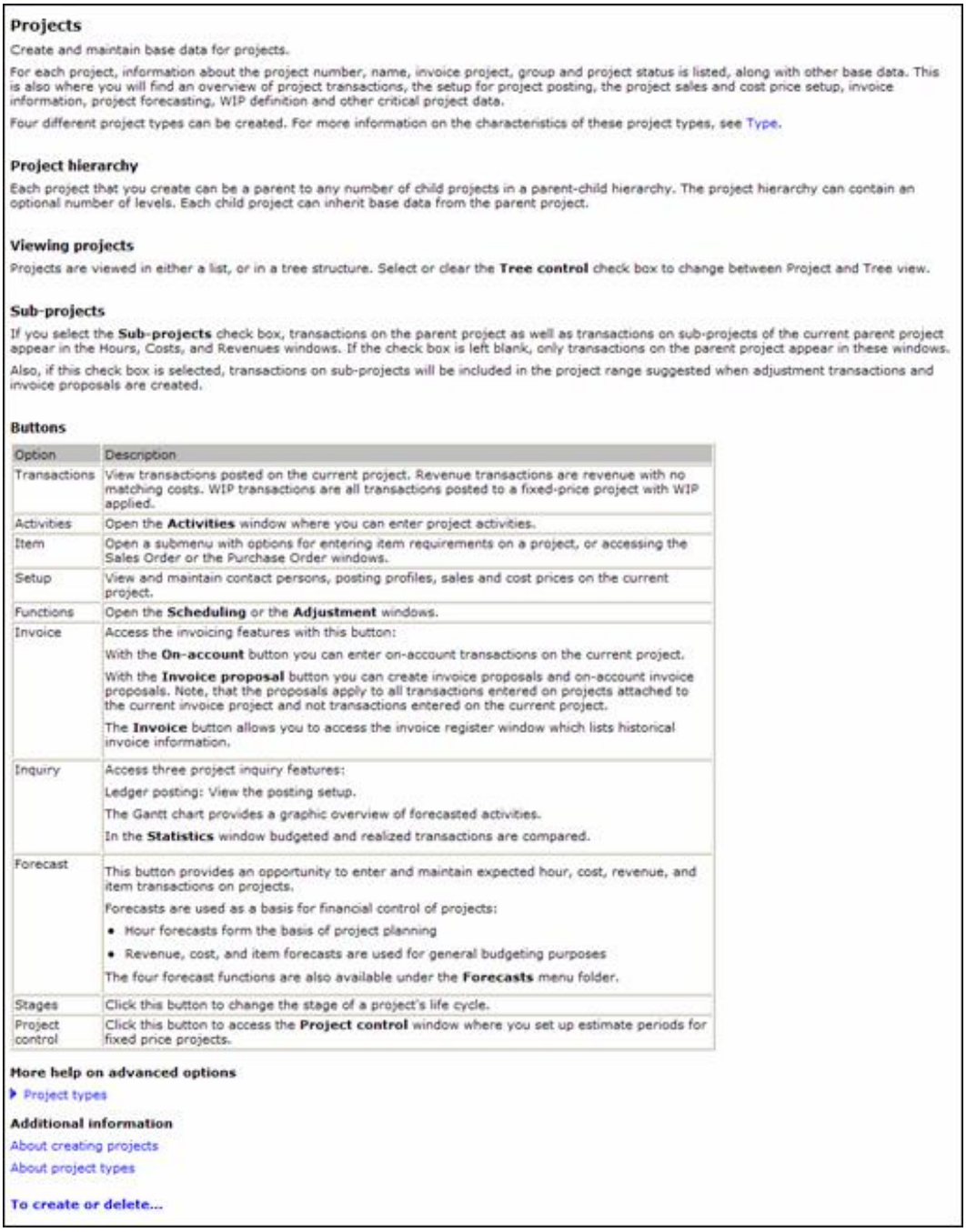
For more information about Microsoft Dynamics AX user assistance requirements, see the *Microsoft Dynamics AX User Assistance Best Practices Handbook*. The handbook is available as a compiled HTML Help file (.chm file) on Microsoft Dynamics AX Central, the Microsoft Dynamics AX community Web site.

Help Requirements

Help in a Microsoft Dynamics AX application must comply with the following requirements.

- UI 12.1 - mandatory: In Form Help, explain the overall purpose of the form, when and why the form should be used, and how its use will affect the rest of the system.
- UI 12.2 - mandatory: Provide Form Help for each form.
- UI 12.3 - mandatory: Use the Form Help format shown in Figure A.9. For more information, see UI rules 12.4 and 12.9.
- UI 12.4 - mandatory: Use a heading for each Form Help topic. Include the form name in the heading, and format the heading as shown in Figure A.9. The form name appears in the window title. In Figure A.9, the heading is Project.
- UI 12.5 - mandatory: Divide Form Help text into short sections, and give each section a subheading. Do not exceed 8 lines in each section. Format text and subheadings are shown in Figure A.9.
- UI 12.6 - mandatory: In Help for form buttons, explain the purpose of each form button. Use the format shown in Figure A.9.
- UI 12.7 - mandatory: In Form Button Help, explain how to enable and disable controls (text boxes, buttons, and so on).
- UI 12.8 - mandatory: In Form Help, provide links to other appropriate Form Help topics. If you use links, use the heading Additional information, as shown in Figure A.9.
- UI 12.9 - mandatory: Be sure to include the link **To create or delete**.

Figure A.9 is an example of a Form Help window that meets current requirements. This example is Help for Microsoft Dynamics AX projects.



Projects

Create and maintain base data for projects.

For each project, information about the project number, name, invoice project, group and project status is listed, along with other base data. This is also where you will find an overview of project transactions, the setup for project posting, the project sales and cost price setup, invoice information, project forecasting, WIP definition and other critical project data.

Four different project types can be created. For more information on the characteristics of these project types, see Type.

Project hierarchy

Each project that you create can be a parent to any number of child projects in a parent-child hierarchy. The project hierarchy can contain an optional number of levels. Each child project can inherit base data from the parent project.

Viewing projects

Projects are viewed in either a list, or in a tree structure. Select or clear the **Tree control** check box to change between Project and Tree view.

Sub-projects

If you select the **Sub-projects** check box, transactions on the parent project as well as transactions on sub-projects of the current parent project appear in the Hours, Costs, and Revenues windows. If the check box is left blank, only transactions on the parent project appear in these windows.

Also, if this check box is selected, transactions on sub-projects will be included in the project range suggested when adjustment transactions and invoice proposals are created.

Buttons

Option	Description
Transactions	View transactions posted on the current project. Revenue transactions are revenue with no matching costs. WIP transactions are all transactions posted to a fixed-price project with WIP applied.
Activities	Open the **Activities** window where you can enter project activities.
Item	Open a submenu with options for entering item requirements on a project, or accessing the Sales Order or the Purchase Order windows.
Setup	View and maintain contact persons, posting profiles, sales and cost prices on the current project.
Functions	Open the **Scheduling** or the **Adjustment** windows.
Invoice	Access the invoicing features with this button: With the **On-account** button you can enter on-account transactions on the current project. With the **Invoice proposal** button you can create invoice proposals and on-account invoice proposals. Note, that the proposals apply to all transactions entered on projects attached to the current invoice project and not transactions entered on the current project. The **Invoice** button allows you to access the invoice register window which lists historical invoice information.
Inquiry	Access three project inquiry features: Ledger posting: View the posting setup. The Gantt chart provides a graphic overview of forecasted activities. In the **Statistics** window budgeted and realized transactions are compared.
Forecast	This button provides an opportunity to enter and maintain expected hour, cost, revenue, and item transactions on projects. Forecasts are used as a basis for financial control of projects: • Hour forecasts form the basis of project planning • Revenue, cost, and item forecasts are used for general budgeting purposes The four forecast functions are also available under the **Forecasts** menu folder.
Stages	Click this button to change the stage of a project's life cycle.
Project control	Click this button to access the **Project control** window where you set up estimate periods for fixed price projects.

More help on advanced options

Project types

Additional information

About creating projects

About project types

To create or delete...

Figure A.9
Form Help window

Requirements for Screen Tips and Tool Tips

Screen tips and tool tips in a Microsoft Dynamics AX application must comply with the following requirements.

- UI 12.10: In each screen tip, answer the questions “What is this?” and “Why should I use it?”
- UI 12.11: Provide information that the object label does not convey. Duplicating a text box label, for example, even with a different word order, is not helpful.
- UI 12.12: Provide a screen tip for each text box and drop-down box and for each item in a table. You do not have to provide screen tips for check boxes or radio buttons.
- UI 12.13: Use the screen tip format shown in Figure A.10. Do not use a heading. Use bold type for important words and phrases. Provide links to relevant Microsoft Dynamics AX windows. If you use links, use the format shown in Figure A.10.
- UI 12.14: In a screen tip for a drop-down control with fixed items, explain the meaning of each item as shown in Figure A.10.
- UI 12.15: In each tool tip, answer the questions “What is this?” and “Why should I use it?” See Figure 3.10 for an example of a tool tip.
- UI 12.16: Make tool tips brief. Do not exceed two lines. You can provide a more extensive description of the control in the Form Help.
- UI 12.17: Provide a tool tip for each command button. Tool tips are not required for command buttons with submenus (such as the **Setup** button in Figure A.10) and for command button submenu items.
- UI 12.18: Use the tool tip format shown in Figure A.10.

Figure A.10 shows both a screen tip for the Invoice project text box and a tool tip for the Stages command button. Only one of these tips can appear at a time.

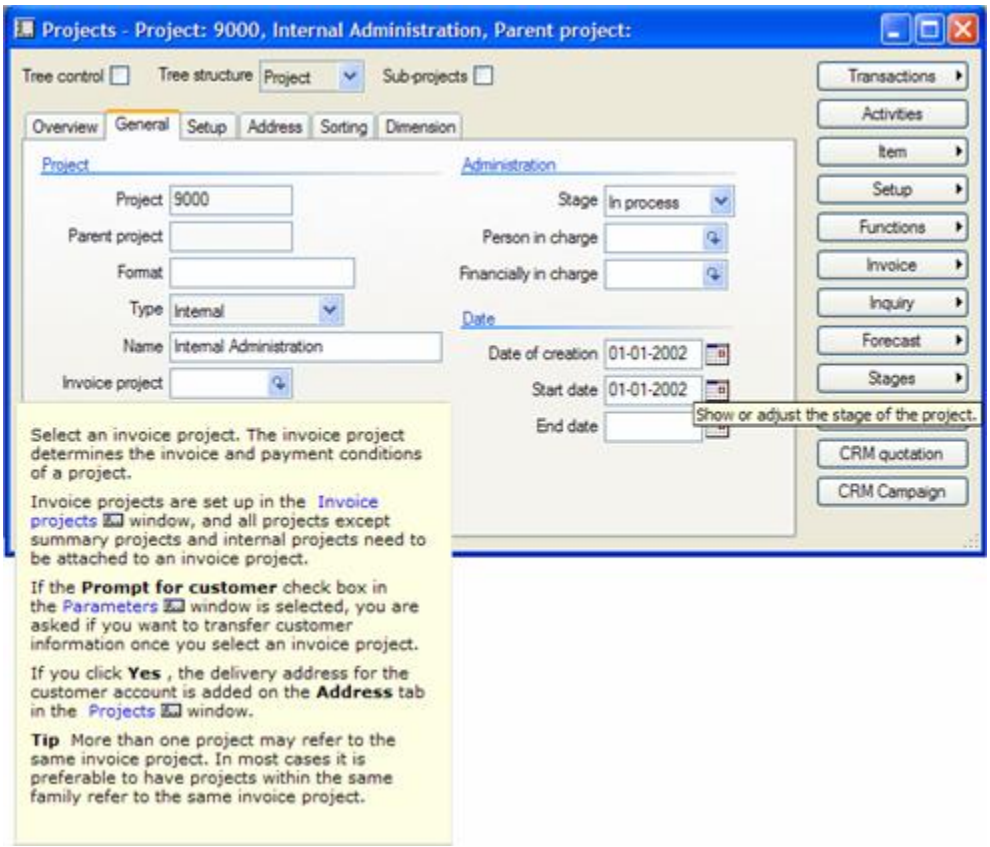


Figure A.10

Example of a screen tip and a tool tip

Requirements for Status Bar Messages

A status bar message automatically appears in the left side of the status bar (in the lower-left corner of the Microsoft Dynamics AX window). A status bar message must comply with the following requirement:

- UI 12.19: A status bar message must answer the questions: “What is this?” and “Why should I use it?”

Requirements for Explicit Help

Explicit Help is an explanation of an object that the user has selected from a table. It appears automatically above the table. Explicit Help must comply with the following requirement:

- UI 12.20: Use the format shown in Figure A.11 (see the text: “Unique key for vouchers, used when posting sales tax settlement”).

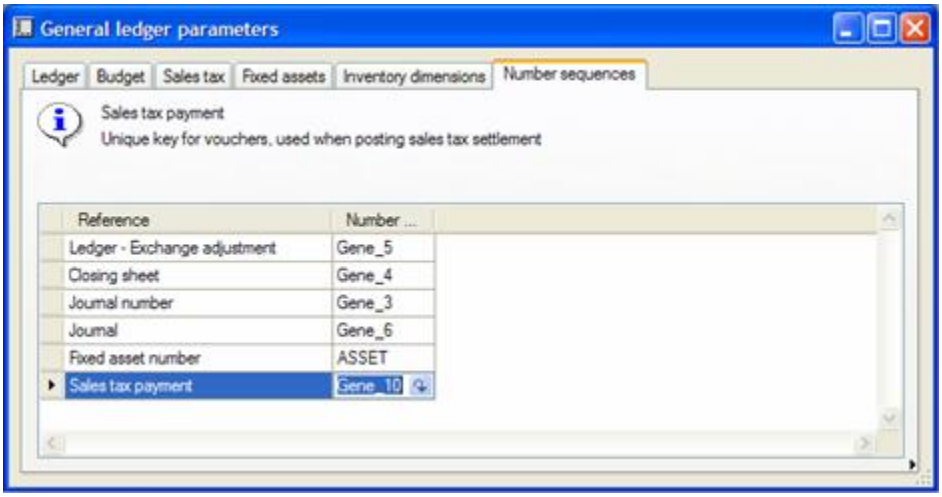


Figure A.11
Explicit Help for the sales tax payment option

Requirements for Messages

Messages in a Microsoft Dynamics AX application must comply with the following requirements.

- UI 13.1: Classify messages to the user as *critical*, *warning*, and *information* to comply with Windows User Experience rules.
  - A *critical* message informs the user of a serious problem that requires intervention or correction before work can continue.
  - A *warning* alerts the user to a condition or situation that requires a user decision and input before processing can continue, such as an impending action with potentially destructive, irreversible consequences. The message can be in the form of a question: for example, “Save changes to MyReport?”
  - An *information* message provides information about the results of a command. It offers no user choices. The user acknowledges the message by clicking the **OK** button.

  Do not confuse critical messages and warnings.

- UI 13.2: Use the following symbols to indicate the classification of a message:

   Critical message
   Warning message
   Information message

  Do not use any other symbols for messages.
- UI 13.3: Make sure that critical and warning messages are constructive. They must describe a way of solving the user's problem.
- UI 13.4: Make sure that a typical user can understand the message. Write the message clearly, without jargon or confusing terminology.
- UI 13.5: In informative messages, include an **OK** button and no other buttons.
- UI 13.6: In a message box, include **Yes** and **No** buttons only if the message contains a clearly phrased question to the user.
- UI 13.7: Do not use routine confirmation messages. You may provide confirmation messages for transactions that users consider critical; for example, you could use them for password changes or account deletions.

Wizard Requirements

Wizards in a Microsoft Dynamics AX application must comply with the following requirements.

- UI 14.1: In a wizard, include a welcome page, one or more interior pages, and a successful completion page.
- UI 14.2: On the welcome page, explain the purpose of the wizard. Do not include any controls.
- UI 14.3: Divide wizard pages into three parts as shown in Figures A.12 through A.14. You can format the interior pages as shown in Figure A.13 or you can format them to resemble the welcome and successful completion pages.
- UI 14.4: Make the wizard pages uniform in appearance. Use the same graphic on the left side throughout the wizard.
- UI 14.5: Make sure that each wizard page has a Back button, a **Next** button, and a **Cancel** button. Position these buttons as shown in Figures A.12 through A.14. Disable **the** Back button on the welcome page. Replace the **Next** button with a **Finish** button on the successful completion page, as shown in Figure A.14.
- UI 14.6: Do not use any additional buttons other than those listed in UI rule 14.5.
- UI 14.7: Disable the Next button on interior pages until the user has entered the information that is required to continue.
- UI 14.8: Make sure that the wizard pages are easy to understand. Include a limited amount of text, and ask only for information that is required for a simple version of the task.
- UI 14.9: Make sure that the wizard does not open new windows.
- UI 14.10: Make sure that interior pages do not contain more than three user controls, in addition to the three buttons at the bottom of the wizard. If you require more controls, divide the interior page into two or more separate interior pages.
- UI 14.11 (Recommendation): Make sure that users can start the wizard from a visible control. This could be a button labeled Wizard in the right side of a form or a form icon on the main menu. Do not require the user to press **CTRL+N**.

- UI 14.12 (Recommendation): You should include default values or settings when possible.

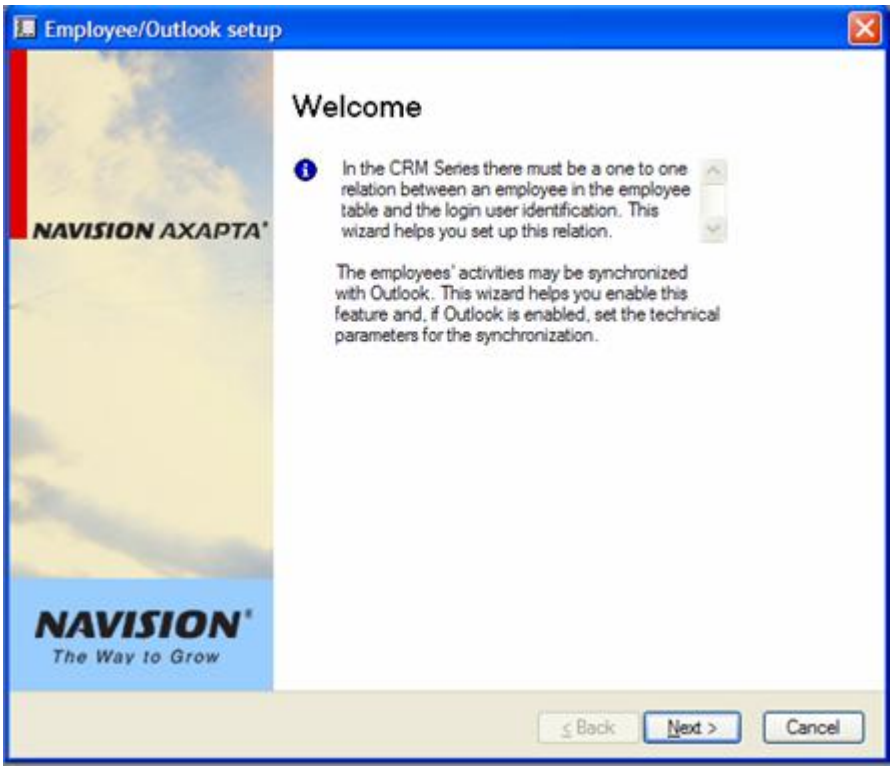


Figure A.12
Wizard welcome page

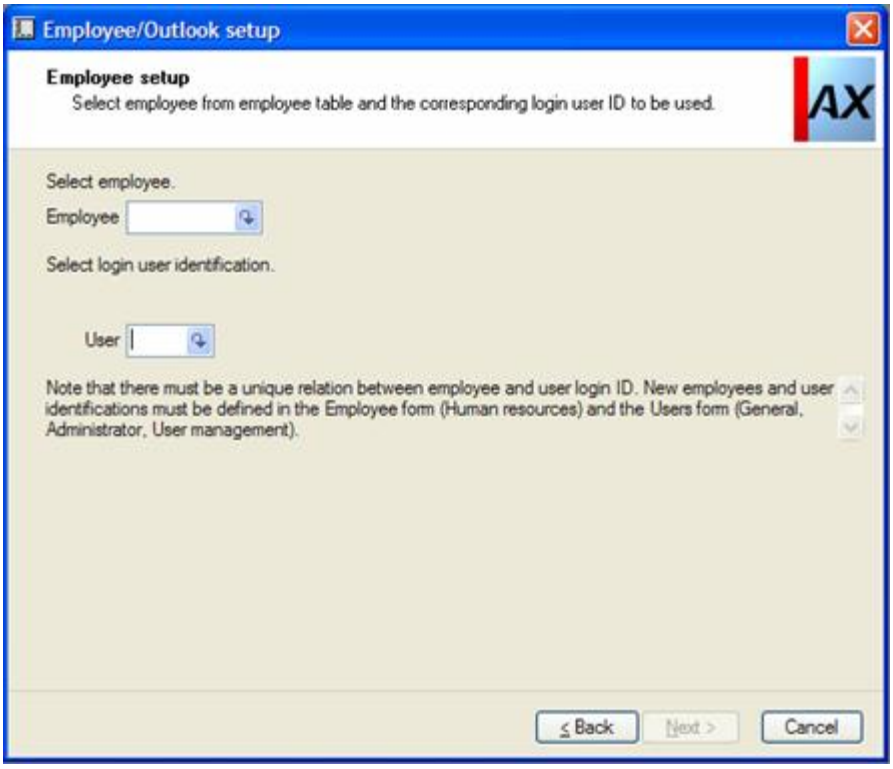


Figure A.13
Wizard interior page

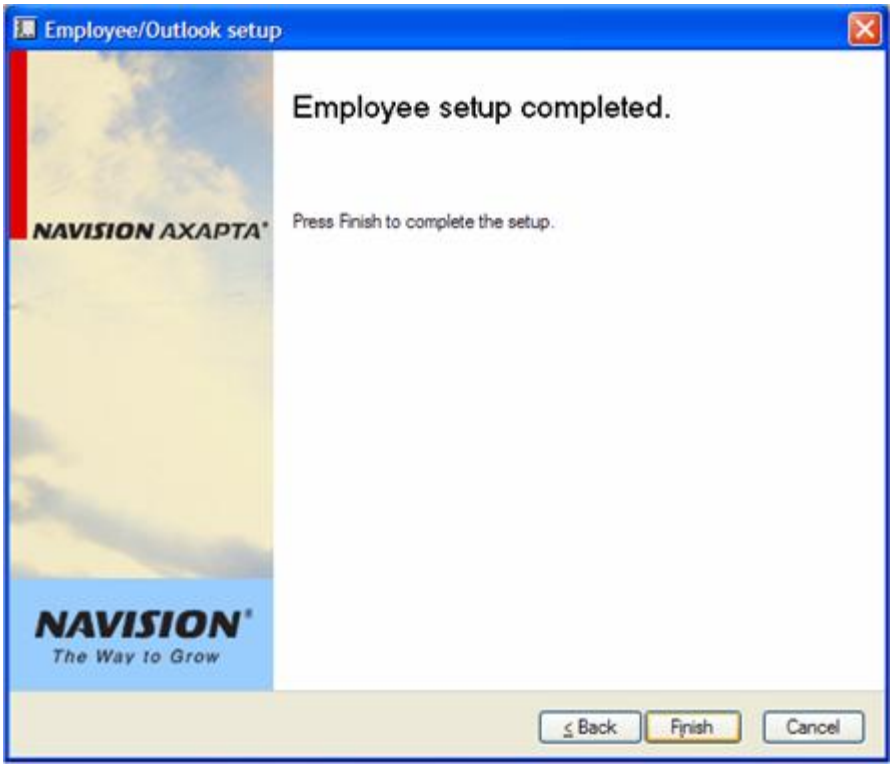


Figure A.14

Wizard successful completion page

Enterprise Portal Requirements

The Enterprise Portal in a Microsoft Dynamics AX application must comply with the following requirements.

- UI 15.1: Make sure that the Web site is clearly visible in a browser with a screen size of 1024 x 768 pixels with the standard button and address toolbars enabled.
- UI 15.2: Make sure that all top-level Web pages, such as Home pages and Activity Centers, have an icon and a title.
- UI 15.3: On an Activity Center page, the title must match the corresponding name on the top menu. For example, the title of the Purchase Activity Center page must be Purchase on both the page and the menu.
- UI 15.4: Do not use icons or titles on the top of Web pages that are not top-level pages. The title on these pages must be in blue.
- UI 15.5: Except on Home pages or Activity Centers, use only one form per Web page.
- UI 15.6: Activity Center pages must contain a form, usually a list, and an instruction to pick a task from the list. For example, on a Sales Activity Page, you should have an instruction to Pick a Sales task (or similar instruction) and a list of sales tasks.
- UI 15.7: Make sure that the instructions on the Activity Page are helpful and instructive. For example, an instruction to "Enter name" is not helpful, while the instruction to "Select a customer and enter the item number and quantity to calculate prices" is helpful.
- UI 15.8: Organize all fields into groups.
- UI 15.9: If the Web site visibility requirement (rule UI 15.1) allows, make sure that users can employ standard controls to add and remove grid lines.
- UI 15.10: Set the maximum number of grid lines to **Auto**.
- UI 15.11: Use a Tunnel to display Help.
- UI 15.12: Do not display Help above lists.

Appendix B: Microsoft Dynamics AX Consistency Verification Test

Company:

Application:

Version:

Prepared by:

Date Prepared:

☐ *Identify the intended purpose of the product.*

Specify what fundamental service the product is supposed to provide. To the extent possible, define the audience for the product. Write a paragraph that briefly explains the purpose of the product and describes its intended audience.

Example:

Test Application is an application for managing data warehouse that enables the user to define the data warehouse strategy in a multi-facility and multi-user environment.

Describe the product purpose:

☐ *Identify primary functions.*

Term	Definition	Notes
Primary function	Any function so important that, in the estimation of a typical user, its inoperability or impairment would make the product unfit for its purpose.	A function is *primary* if you can associate it with the purpose of the product and it is essential to that purpose. Primary functions define the product. For example, the function of adding text to a document in Microsoft Word is so important that the product would be useless without it. Groups of functions, taken together, could be a primary function. For example, although no single function on the drawing toolbar of Word is primary, the complete toolbar might be primary. If the toolbar is primary, most of the functions on that toolbar should be operable for the product to pass the test.

Examples of primary functions:

- Manage cross-docking operations.
- Manage stock environment.
- Manage IT interoperability with a fast carrier company.

List all primary functions:

☐ *Identify contributing functions.*

Term	Definition	Notes
Contributing function	Any function that contributes to the utility of the product, but is not a primary function.	Although contributing functions are not primary, their inoperability could be grounds for test failure. For example, users may be able to do useful things with a product, even if it has an Undo function that never works, but most users will find that unacceptable. Such a failure would violate fundamental expectations about how products should work.

Example of a contributing function:

- Generate a 3-D report.

List all contributing functions:

☐ *Specify potential instabilities and challenging data.*

1. List five to 10 functions or groups of functions (preferably primary functions) for focused instability testing.
2. Specify challenging data for each selected function. Think of large, complex, or otherwise challenging input.

Examples:

- Functions that interoperate with other products (for example, object linking and embedding, file conversion).
- Functions that handle events external to the application (for example, wake up a sleeping computer when a fax arrives).
- Functions that make intensive use of memory.
- Functions that interact extensively with the operating system.
- Functions of unusual complexity.
- Functions that change operating parameters (for example, preference settings).
- Functions that manipulate operating system configuration.

☐ *Design and record a consistency verification test.*

Prerequisites:

List any action that must be performed before the consistency verification test can be executed.

Examples:

- Install Microsoft SQL Server 2005.
- Use radio frequency identification (RFID) to identify hardware.

Required Information:

List any information that a user must know to perform the consistency verification test.

Examples:

- User must log on as User x.
- User must know the product serial number.
- User must know the account passwords.

Test Procedure:

Complete the following procedure to test each primary function of the application. You must describe each step that is required to test a primary function. You can combine similar functions, if appropriate.

Example:

- Manage cross-docking operations.
  1. Define the goods involved.
  2. Determine the delivery locations.
  3. Determine the transport company.
- Manage the stock environment.
  1. On the menu, select **File** and then select **Save**.
  2. Type the file name in the field, select the location for the file, and then click **OK**.
- Manage IT interoperability with a fast carrier company.
  1. Configure database access.
  2. Define user rights and circuit approval.