Deployment of Enterprise Architect

By Sparx Systems

Table of Contents

Introduction

Sparx Systems' Enterprise Architect is designed for use within large corporate environments. As a scalable modeling platform, Enterprise Architect provides a range of deployment options to accommodate the variety of modern enterprises.

This whitepaper discusses the options available for enterprise-wide sharing of model information.

Typical Enterprise Structures

These are typical organizational structures encountered when deploying modeling software for a large corporation. Each has its own requirements and software configurations.

- Single Site
  - One building with a large user base on a LAN
  - Multiple buildings with Cloud Connectivity
- Single Site – Multiple Projects
  - Multiple repositories
  - One repository – many projects
- Multiple Sites
  - Multiple sites with a low speed WAN or Cloud connection
  - Head Office with multiple external contractors working on customer sites

Deployment features

Enterprise Architect has a number of features designed to simplify deployment across large organizational structures. In combination, these features give users the flexibility to create their own specific layout whilst allowing for expansion.

These features are dependent on the edition of Enterprise Architect installed, as well as the data storage used. The model's data store is referred to as a 'Repository'.

For more information on the features and repository types supported by the different Enterprise Architect editions, see the *Editions Comparison* web page.

The key features covered in this whitepaper are those that assist in the enterprise-wide deployment of Enterprise Architect:

- Choice of repository
- Data Access: local versus Cloud Services or RAS
- Change Management
- Team Collaboration

Choice of Repository

For editions above the Enterprise Architect Professional edition, models can be stored in either a file based repository or a DBMS repository. There are pros and cons for both solutions.

Table 1 Provides a simple comparison of file-based and DBMS repository types:

Function	**EAP**	**Firebird**	**DBMS**
Replication	Yes	No	No
File Based	Yes	Yes	No
Number of LAN users	1..~10	1	Unlimited
Cloud users	N/A	Unlimited	Unlimited
Non-Corruptible	No	No	Yes

Table 1: Comparison of file based repositories with DBMS repositories

EAP file repository

EAP (Enterprise Architect Project) files are based on the Microsoft Jet 3.5 database engine (MS Access '97 format .mdb). Enterprise Architect also supports the Jet 4.0 database engine.

Note: The Jet 3.5 file format does not support Unicode. For details on multilingual Unicode support using .eap files, see the appendix Unicode Support for .eap Files and the Notes section on the *Basics* Help page.

The benefits of this repository type are:

- Replication of the repository
- Simple file access across a shared network drive

The limitations of this repository type to consider are:

- Concurrent access is limited to small groups of users
- Data corruption can occur if there is a network/power failure while editing
- There are limitations on the data size supported by the Jet database (less than 30-40 mb is recommended).

Firebird file repository

Firebird file repositories are based on the Interbase DBMS. Like the EAP repository, these are file based and can be accessed from:

- A local drive for single user operation
- A cloud service for multi-user access

Note that Firebird files do not support multi-user file access from a network drive.

Firebird files are more robust than .eap files, in that they support Unicode. However, they do not support the MS Jet replication of the .eap files.

DBMS repository

Using a DBMS repository overcomes the limitations of file-based repositories. Typically, dedicated DBMS servers provide faster response times for a larger user base than the file based repositories. Further, any network errors are handled by the ability of the DBMS server to roll back transaction failure caused by external conditions. DBMS repositories can be accessed from a Cloud service. Although replication is not supported with DBMS repositories, with the Cloud connectivity replication is largely superseded.

Enterprise Architect supports the 10 most popular types of DBMS server repositories. For details on configuring DBMS servers for Enterprise Architect model repositories, see the *Server Based Repositories* Help page.

Deployment Options

Enterprise Architect supports a wide variety of options for a diversity of deployment layouts. As it is a workstation-based application it can simply run without the need for a server based environment. However, there are several server-based options available for large scale deployment, including:

- Connectivity to a DBMS
- Floating Licensing key store service
- Connection via the Cloud Service
- MDG Technologies

When deploying Enterprise Architect in a corporate environment you might want to use a combination of these options. Use in these environments can range from users updating their own file based repositories, through to large teams interacting with a common model over numerous geographic sites.

Note: For information on remote (unattended) installation across a corporate network see Appendix: 7: Remote Installation of EA

The following section discusses deployment scenarios based on the typical enterprise structures outlined in the Introduction.

Single site

The most common scenario is a single site with a single, shared repository catering for one specific project. In larger organizations, however, projects will typically share a set of common aspects. These can range from common procedural code through to common report templates that meet corporate standards. The following covers these scenarios.

Using an EAP file on a shared drive

For .eap files this is a simple setup. A repository file is stored on a file server, which is accessible to users across a local area network (LAN). As noted earlier, there are limitations of an .eap file-based model repository in that they are best used for groups of fewer than 10 users.

Figure 1 shows a typical file based configuration.

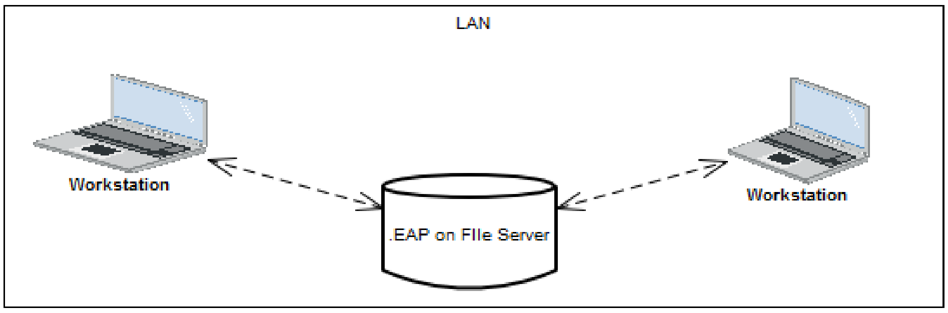


Figure 1: Workstations accessing a file based repository

Using a DBMS repository

The DBMS repository offers a far more robust environment in situations where there are a large number of users.

For very large repositories the DBMS interface supports a 'Lazy Load' option. When opening a repository with this enabled, only the data for the visible portion of the project tree is loaded. This allows a faster load of the model, at the expense of small delays on the initial access of sub-packages.

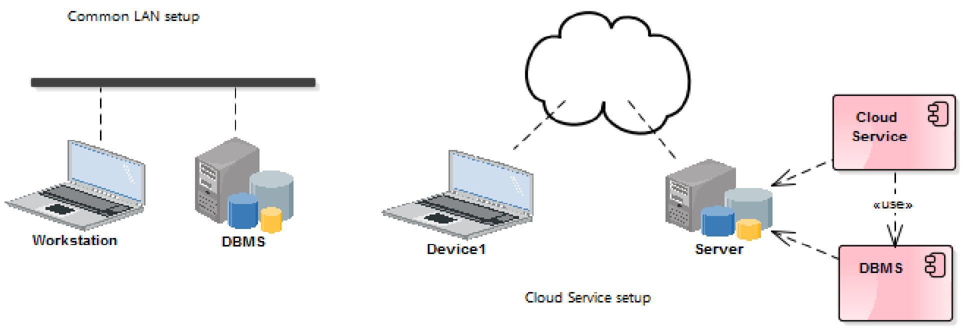


Figure 2: LAN and Cloud Connectivity to a DBMS

Multiple Projects

When there are multiple projects under development there may be code foundation classes, scripts, templates and reports that are common to all the projects. As Enterprise Architect is only limited in the size of a repository by the specifications of the active DBMS, the cleanest solution for this is to combine all projects in the one repository and set access rights based on Security settings to keep specific groups of users working on specific models. However there can be reasons for needing multiple repositories with some interchange. The following points cover these scenarios:

One Repository – Multiple Projects

Use one model repository with multiple project Root Nodes and multiple Views (see Figure 3). Each Root Node or View can represent a separate project. Commonality between projects (frameworks, foundation classes etc.) can also be captured under a Root Node or View.

Using a single repository will ensure that common code and references need only be changed once to affect all projects.

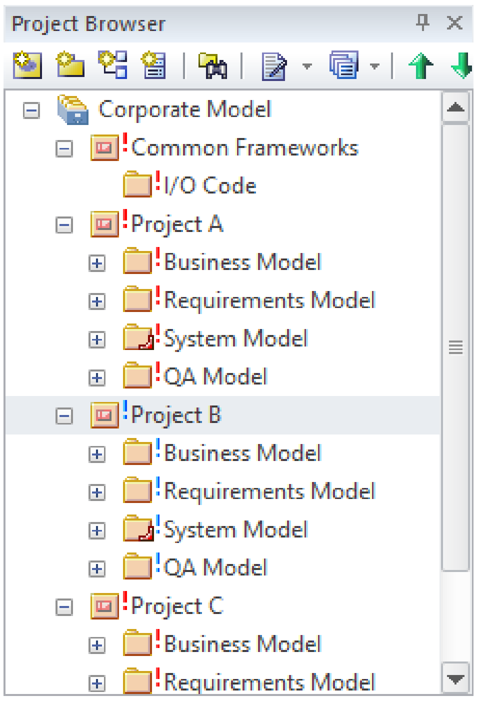
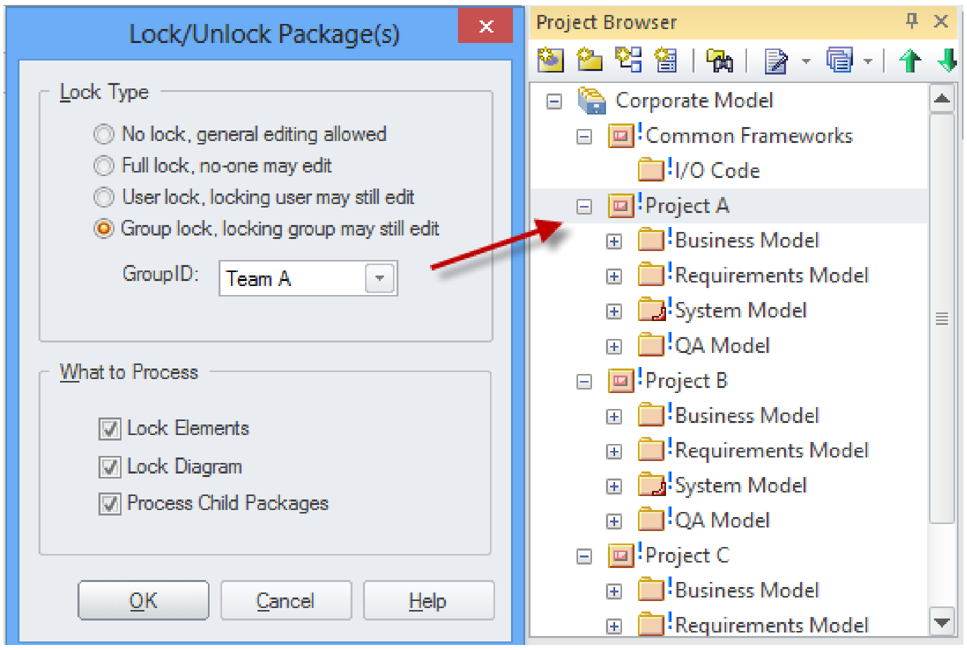


Figure 3: A number of Projects using Common Frameworks.

Additionally, Enterprise Architect's security can be used to restrict a user's right to make modifications in designated areas (package sub-trees). Elements outside a designated area can be accessed for use in diagrams, however, data modification to these Elements is restricted to the group of users that have security access to the package-tree that Element is contained in.

Figure 4 is an example where an administrator uses group locking to restrict a Package View for a specific team to access.



Lock/Unlock Package(s)

Lock Type

- No lock, general editing allowed
- Full lock, no-one may edit
- User lock, locking user may still edit
- [✓ Group lock, locking group may still edit]

GroupID: Team A

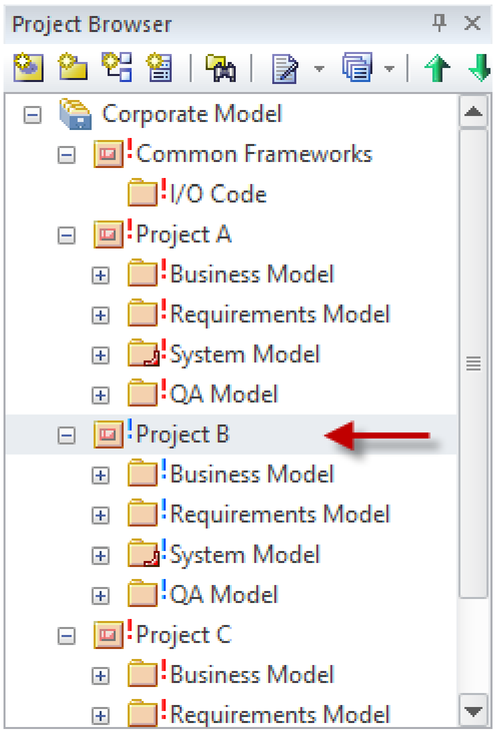
What to Process

- [✓] Lock Elements
- [✓] Lock Diagram
- [✓] Process Child Packages

```
Corporate Model
    !Common Frameworks
        !I/O Code
    !Project A
        !Business Model
        !Requirements Model
        !System Model
        !QA Model
    !Project B
        !Business Model
        !Requirements Model
        !System Model
        !QA Model
    !Project C
        !Business Model
        !Requirements Model
```

Figure 4: Project Browser view showing the group locking by the administrator.

Figure 5 shows when accessed by a user in the group 'Team B'. Note the coloring of the lock markers. ! indicates the locked packages are accessible to the user currently logged in.



```
Corporate Model
    !Common Frameworks
        !I/O Code
    !Project A
        !Business Model
        !Requirements Model
        !System Model
        !QA Model
    !Project B
        !Business Model
        !Requirements Model
        !System Model
        !QA Model
    !Project C
        !Business Model
        !Requirements Model
```

Figure 5: Project Browser showing the locking access for a Team B user.

Note: The packages are organized under a single root-node to ensure reports can be generated across the model.

Multiple Repositories – Sharing resources

If you have multiple projects on multiple repositories, and you need to use the same resources across these projects, you can use a number of features for interchanging data or accessing common resources. In all cases, it is recommended that a master repository is used to maintain a single source for the common project data.

In this scenario there are two key sets of data:

- Packages (i.e. modeling packages and common frameworks)
- Resource data (such as type definitions and report templates)

There are several options for interchanging both of these sets of data between project repositories and a master repository.

Packages:

- Baseline Difference and Merge (see Using Baselines)
- RAS
- Package Control
- Version Control
- XMI import/export

Details on each of these features are provided in the Change Control section.

Resource data:

For resource data interchange there are two core options:

- **Reference data Import/Export**
  Reference data can be periodically updated from the master repository using the **Export Reference Data** option.
- **MDG Technologies**
  Using MDG technologies you can build a module of resources that you can export. This MDG Technology can then be referenced by any Workstation instance of Enterprise Architect.

  For more details see the section on MDG Technologies.

Large Projects – Branching and Merging Multiple Phases

In large 'phased' development projects, the design of the next phase can be carried out in a ‘Branch’ repository that is a copy of the 'Trunk' repository being used in the main development. After a phase of design work, ‘Baselines’ can be created against the Trunk model and using 'Baseline, Difference and Merge' the Branch can then be merged back into the Trunk model. For more information see the section Using Baselines.

With the combination of MDG Technologies and a package transfer using the Baselines Merge 'Load Other Baselines' feature you can set up a number of repositories that interact with common data/resources and keep the common data/resources up to date.

Global sharing of Assets

For scenarios where there are multiple repositories scattered across different organisations, the Reusable Asset Service (RAS) can be used to collaboratively interchange model data. A good example is when collaborating on modelling an industry standard with contributors working for a number of different corporations. This requires a global access point with options to contribute new modelling or download the latest changes implemented to the specification/framework.

For more information see the section Reusable Asset Service.

Multi site

There can be many scenarios where connectivity is required external to a Local Area Network. These can range from corporations or organizations with multiple sites across a continent through to intercontinental outsourcing development. There are several key features covering these scenarios:

- The Cloud Service
- WAN Optimizer
- RAS services

Cloud Service (Connect via HTTP)

The Cloud service is an HTTP connectivity Service that can be deployed on an internal LAN based server or on an external web based server. It is not a hosting service, but can be deployed on hosting services (such as Amazon or Azure).

Using HTTPS SSL connections you can ensure a secure connection for individual or team-based access to a common repository.

The Cloud Service provides for scenarios for accessing a repository ranging from multi-building WAN access through to intercontinental access to a repository.

The advantages of using an HTTP access via the Cloud service include:

- Workstations do not require ODBC setup to access a DBMS
- Connection details are simple and can be easily passed to users
- Supports off-site and on-site connectivity
- Supports SSL secure connections

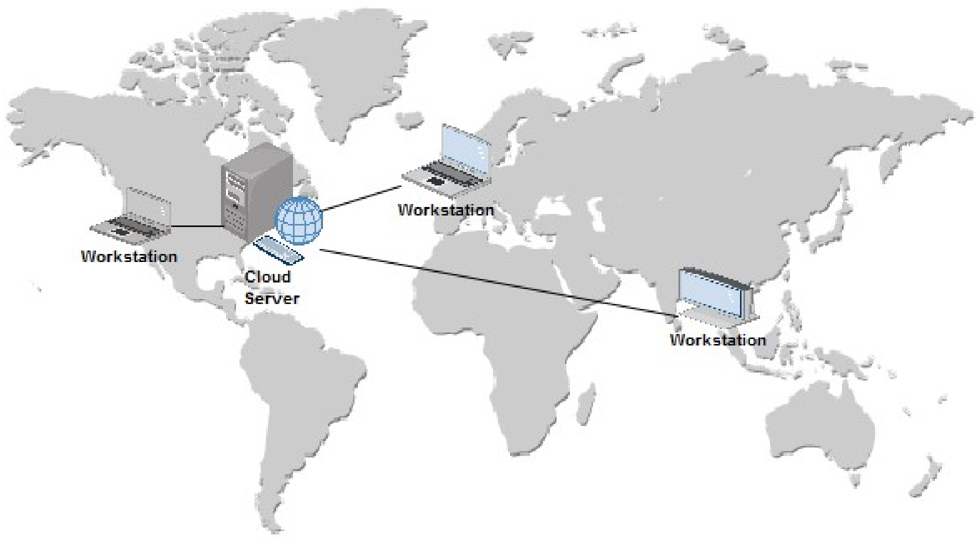


Figure 6: Cloud Service cross continent connectivity

With the Cloud service you can set up linkages with different access rights to multiple repositories. Each linkage is based on a Port. Each DBMS can be set to operate under different ports allowing for setting different access rights.

A simple setup will involve the components laid out in Figure 7.

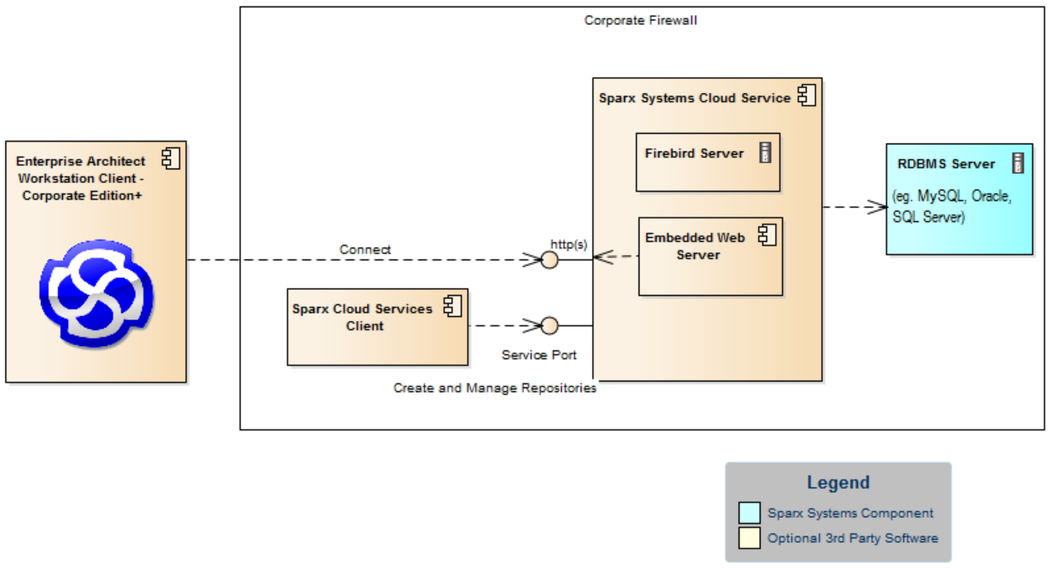

Corporate Firewall
Sparx Systems Cloud Service
Firebird Server
Embedded Web Server
RDBMS Server
(eg. MySQL, Oracle, SQL Server)
Enterprise Architect Workstation Client - Corporate Edition+
Connect
http(s)
Sparx Cloud Services Client
Service Port
Create and Manage Repositories
Legend
Sparx Systems Component
Optional 3rd Party Software

Figure 7: Cloud Service Architecture

Note: The Cloud service is available for the Corporate or extended editions of Enterprise Architect.

For more details on using the Cloud service see the *Connecting to Projects Via the Cloud* Help topic.

WAN Optimization

WAN optimization provides a data compression service between the workstation and the DBMS repository. This has largely been superseded by the Cloud Service, which includes the same data compression, but also supports a more parallel flow of data requests than is supported by a single ODBC connection.

For any scenario that requires very strict security (that does not allow for HTTPS linkage), the Wan Optimizer can be used as an alternative connection across a Wide Area Network.

For more details see the *WAN optimizer* Help pages and the sections on the *Cloud Service*.

RAS

See the section on the Reusable Asset Service below.

Change Control

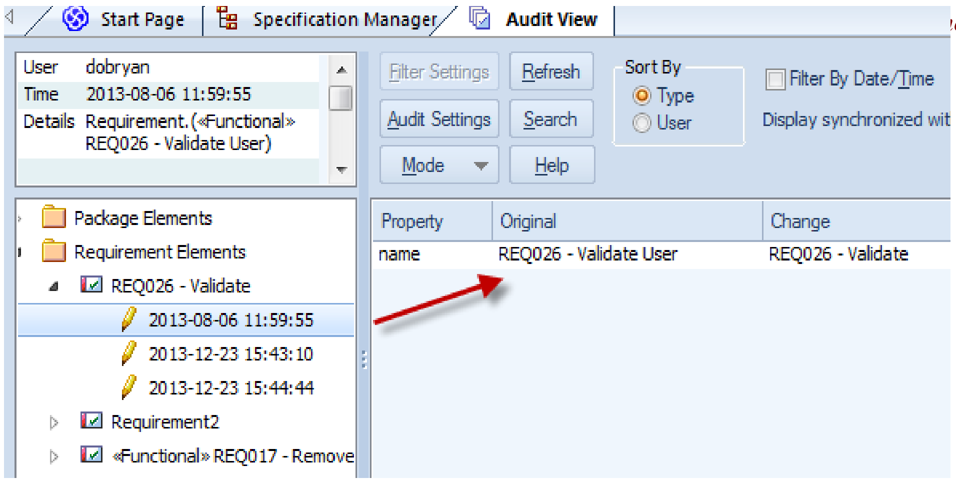
Enterprise Architect supports a number of features for monitoring changes to the model. Each of these features have their different uses depending on the organization of your modeling and your application development.

These change control features include:
- Auditing
- Managing Baselines
- Version Control
- Reusable Asset Service

Auditing

The Audit feature enables you to record model changes in Enterprise Architect. It records details of **who** changed an element, **when** and **what** was changed, and the prior state of the model. This can be particularly useful for recording a history of changes to requirements models.

Figure 7 is an example of viewing alterations to an element directly in the Audit View. This shows a number of alterations with the first selected to show the details on the right pane.



User	dobryan
Time	2013-08-06 11:59:55
Details	Requirement.(«Functional» REQ026 - Validate User)

Sort By: [✓ Type] User

- Package Elements
- Requirement Elements
  - REQ026 - Validate
    - 2013-08-06 11:59:55
    - 2013-12-23 15:43:10
    - 2013-12-23 15:44:44
  - Requirement2
  - «Functional» REQ017 - Remove

Property	Original	Change
name	REQ026 - Validate User	REQ026 - Validate

own

Figure 7: Audit view showing a change to a Requirement

With the Auditing View enabled the View | System Output >Audit History window can be used to show the list of changes for the selected element. Figure 7 shows a requirement selected in the Specification Manager and a set of alterations to this element logged in the Audit History view.

The System Output window can be accessed from the main menu View | System Output **(Ctrl+Shift+8).**

Although the Audit features provide a useful history of change, the data required over a long period or with large numbers of users can be taxing on the repository (DBMS). This can be avoided by keeping a fixed audit period and creating baselines for each period, then saving and clearing the audit log. This can provide a longer term, lower volume, but more segmented history.

For more information on using the Auditing features see the *Auditing* Help topic.

Using Baselines

The auditing feature outlined above provides continuous tracking and logging of changes to requirements. The Baseline Management feature provides a more periodic means of tracking changes, along with support for comparing and merging changes. It allows Baselines of a model to be created on a periodic basis (such as by month, phase, version or build). Baselines can then be compared to the current model and changes selectively rolled back.

Branching using Baselines
Baselines can also be used for 'Branching' by creating a duplicate repository (a Branch) from the source repository (the Trunk). The branch can then be updated.

After updating the model in the Branch repository, you can merge the changes back to the Trunk repository by creating a Baseline in the Branch and then using the 'Load other Baselines' feature on the Trunk repository.

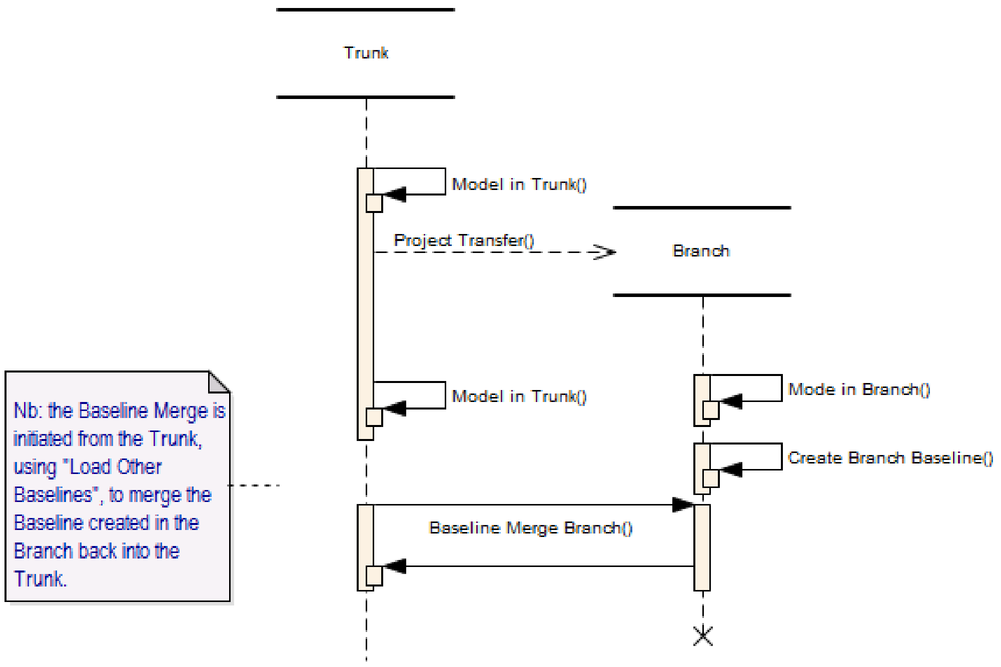

Trunk
Model in Trunk()
Project Transfer()
Branch
Model in Trunk()
Mode in Branch()
Create Branch Baseline()
Baseline Merge Branch()
Nb: the Baseline Merge is initiated from the Trunk, using "Load Other Baselines", to merge the Baseline created in the Branch back into the Trunk.

Figure 8: Using Baselines for transfers between a Trunk and a Branch

For more information on setting up baselines and viewing differences see the *Package Baselines* Help topic.

Version control / Package control

Enterprise Architect supports Version Control of packages and their component sub-packages to a central Version Control repository, which is maintained by a third-party version control application. This provides two key benefits:

- Saving a history of changes to Enterprise Architect packages, including the ability to retrieve previous versions
- Coordinating the sharing of packages between users

Version Control can be set up using any Version Control software that is compliant with the following standards:

- SCC standards
- CVS
- Subversion
- Microsoft TFS

Why use Version Control?

Although code-based version control is used extensively in code development, it is not so easy to implement model based Version Control. A core reason is that code is a simple text based file that does not support explicit cross referencing, making it far simpler to be version controlled than a model that has links referencing other parts of the model. The introduction of tightly controlled cross-referencing limits some of the features that are available in code based Version Control – it is like trying to use a two dimensional tool in a three dimensional scenario.

This is not to say Version Control should not be used, but rather to outline why it is recommended to explore the alternative options available in Enterprise Architect before following the more complex scenario of using a code-based version control system.

Benefits:

- More streamlined, where regular tracking of change is required
- A history of revisions can be viewed and restored
- A version packet can be a package (not a full package tree)
- Version Control can be used for multi-site interchange of modelling
- Package Locking can be used to avoid editing conflicts

Shortfalls:

- The versioned data is stored external to the repository; this data, being separate to the modelling repository, can be lost, whereas Baselines are maintained within the model repository
- Due to the processing required to interface with the Version Control repository, the granularity of versioned packages must be kept small; this creates complications with cross-referencing
- Version Control locking can restrict access to models, especially when packages are left checked-out by a user
- For cross-continental usage the interchange with a Version Control repository can be very slow
- When using multiple Projects, only package data is version controlled. Reference data such as Document Templates, Calendar Events and Glossary must be manually exported/imported into each Project
- If users perform frequent imports from the Version Control system it can be detrimental to the efficiency of other users, due to locking
- External Version Control can introduce configuration issues as these settings need to be configured by each individual user, which is a complicated process and if set incorrectly can result in data loss or corruption

Alternatives:

These alternative options, as discussed in the earlier sections, include using Auditing, Baselines and RAS. They can be used in different combinations to achieve a similar outcome to version control depending on what is required.

The following points cover the broad features of version control along with the alternatives that Enterprise Architect offers.

Model Sharing

- Cloud:
  - Cloud Services allow cross-site connectivity, avoiding the need for multiple repository interchanges
  - Reference data such as Document Templates, Calendar Events and Glossary do not have to be updated and interchanged

- RAS:
  - RAS, using the Cloud Services, supports cross-continent interchange of models as well as a version-based history of items

- Baselines:
  - Baselines can be used for multiple repository interchange (including cross-site interchange)
  - Baselines allow for a more refined selection of alterations to be re-instated

  (merge feature)
  - Baselines support Trunk and Branch cycling of repository data; Version Control for models does not

Revisions

- Auditing:
  - Auditing provides a simple log of alterations for viewing a history of changes

- Baselines:
  - Baselines support periodic storage of models over a period
  - A history of changes is available and differences can be viewed

Package Locking

- Security:
  - Where locking is required, rather than using Version Control use the Security lock in edit mode to ensure users are not attempting conflicting editing

So, in short, where a history of changes is required it is suggested that you first weigh up the pros and cons of these alternatives before deploying a Version Control option.

> **Note:** It is assumed that with version control you have a lower user count on each repository and that this justifies using .eap files. However this is not recommended for large .eap files, as these can cause corruption in processing the XMI interchange for version control.

For more information on setting up the version control systems in Enterprise Architect, see the *Version Control* Help page.

For more details on the use of Version Control see the *Version Control* white-paper.

Reusable Asset Service

The Re-usable Asset Service (RAS) provides a mechanism to distribute reusable model structures from a Cloud-based repository in an open, uniform and timely manner. These models are maintained by designated administrators who submit these for download by other users accessing the Cloud.

This is useful for organizations that are not bounded by borders or continents, but need the ability to model common data and foundation structures without having to set up a complex global version control system.

A good example of this is in creating standards for global standards bodies. The authors of these standards might be members of different companies, based in different organizations on different continents, who want to share common structures in developing the global standard. Rather than manually distributing models or XMI files, the key architects can use a Cloud repository through the Re-usable Asset Service to collaborate in updating the Profile for the standards-based model, and publish the Profile for colleagues to review and use it.

For more information on using the RAS service see the *Reusable Asset Service* Help page.

Team Collaboration

When working with inter-disciplinary teams it is crucial to have mechanisms for collaborating on both resources and ideas. Enterprise Architect supports a variety of features for team collaboration, including a repository based forum, model mail and the means of sharing resources such as report templates (MDG Technologies). When deploying Enterprise Architect for team work it is useful to consider how these features can be used. Note that some of these features are dependent on the user security.

MDG Technologies

In large organizations there can be a number of different teams developing different models or applications that need to work with common resources. These resources can range from company specific Profiles used in modeling, through to coding templates and reporting using company based formatting.

With an MDG technology you can set up common resources to be used by groups using different repositories, across your organization. These shared resources include:

- Profiles (for defining or modifying a modeling language)
- Patterns (for creating model structures for re-use)
- Tagged Value Types (for setting user-defined fields)
- Code Modules (Code generation templates)
- MDA Transforms
- Report Templates
- Linked Document Templates
- Images
- Scripts
- Workspace Layouts
- Model Views
- Model Searches
- Model Templates

Once created, an MDG Technology can be deployed on a common network drive and referenced by settings within Enterprise Architect.

For more details on setting this up see the *MDG Technology SDK* Help page.

Reference Data Import/Export

As a simpler alternative to using MDG Technologies, the Reference Data Import/Export feature can be used for supporting multiple projects that use common data. Typically a master repository is kept up-to-date and a selection of the reference data is periodically propagated to other repositories.

Parts of a repository that are shared include:

- Glossaries

- Type definitions (such as status types)
- Resources, Clients
- RTF and HTML templates
- Security

When reference data is exported, Enterprise Architect writes it out to a custom XML file. This includes table information, filter information, rows and columns.

For more details on exporting data see the *Sharing Reference Data* Help topic.

Security

Although security is an option that is not required for a small group of users there are numerous benefits in using it. These range from providing simple user access rights through to the critical support it provides for some of the multi-user features of Enterprise Architect.

Some of the key facilities that use Security include:

- Workflow scripting
- Project Management Gantt charts and Calendar
- Model mail
- Audit tracking

Enterprise Architect Corporate and extended editions provide user definable security – which allows for restrictions of user access to the model update functions. A password is required to log in, and elements can be locked by a user or a user group.

Security in Enterprise Architect is not designed to prevent unauthorized access; rather it is a means of improving collaborative design and development by preventing concurrent editing, as well as limiting the possibility of inadvertent model changes by users not designated as model authors.

Enterprise Architect provides two security policies:

Standard Security Model
All elements and diagrams are considered unlocked and a user, depending on their user-rights, may lock any element or set of elements at the user or group level as required.

Rigorous Security Model
This assumes everything is locked until explicitly checked out by a user lock. In this mode, an Enterprise Architect model is read-only until a user applies an editing lock to one or more elements.

Security allows elements to be locked for change; however users retain access to place them as linked items in diagrams where they have write permission. These elements will be shown in the diagrams, but are not editable. As an example, a definition of a server is locked by the architect, but remains displayable in a diagram created by the deployment manager.

For more information on security features in Enterprise Architect, see the *User Security* Help page.

Workflow

In a model driven development environment there can be many workflow processes operating in the design and the development of a project.

Using Enterprise Architect's Workflow scripting you can set the order of work to be performed by members of the team and ensure that any specified outcomes are obtained on completion of the Workflow routine.

Workflow scripting is intended to be used by those administering the overall project management of a design and development process. As an administrator, you can use workflow scripting to define your own solution to a workflow process.

For a whitepaper and an example model covering Workflow scripting see the *Workflow Scripting in Enterprise Architect* Community site article.

For more detailed information on using the script see the *Workflow Scripts* Help pages.

Model Forum (Team Review)

Enterprise Architect's Team Review facility helps users to discuss the development and progress of a project. Team members can view and post messages within the modeling environment and can link their posts directly to elements within the model. For distributed team environments, users can connect their Enterprise Architect model to a Team Review hosted in a remote model repository. For more details see the *Team Review Tools* Help topic.

Where multiple repositories are used in a project, Team Review forums in external repositories can be accessed. For more details see the *Team Review Connections* Help topic.

Model Mail

In a large team of developers there can be a lot of message based dialog. It is essential that any messaging can reference items in the model. Enterprise Architect provides an internal mail service for the exchange of messages between the users working on a repository. Two key points for using model mail are:

- Model Mail messages support hyperlink references to internal details (diagram, elements) - not available in email
- Using Model Mail avoids overloading your email with model specific entries

Note: There is support of the mail service in the Automation interface, allowing for setting up code to interface with other mail services if required. See also the script available in the *Workflow Scripting in Enterprise Architect* whitepaper for interfacing with Outlook.

Optimizing Performance

When sharing large models over a network (LAN, WAN or a Cloud), some consideration should be given to optimizing performance. In general, performance depends on:

- The model repository type used (EAP, FEAP or remote DBMS)
- The network response time, which is based on:
    1. Network load
    2. WAN latency (a low latency WAN connection has under 40-50ms latency)

Table 2 identifies which network and repository type configurations are best for high performance.

Repository Type	Network Type	HTTP*	Users	Optimal	Comments
EAP	Local	N/A	1~10	✓	Access files on the workstation drive.
EAP	LAN	N/A	1~10	✓	Performance depends on LAN load. Requires a common network drive.
EAP	WAN	N/A	1~10	✗	Not recommended.
EAP Replication	Local	N/A		✓	Useful for off-site compilation of data. Can be replaced by Cloud Connectivity.
FEAP	Local	✗	1	✓	No ODBC connection. Feap does not support multi-user access.
FEAP	LAN/WAN/ Web	✓	∞ *	✓	Cloud service is required.
DBMS	LAN	✗	∞ *	✓	Good for large teams. Workstations require ODBC configuration.
DBMS	WAN/ Web	✓	∞ *	✓	Optimal for a WAN/Web connection. Workstations do not require ODBC configuration. (MySQL & SQL Server models cause less load than Oracle).

Table 2: Configuration options

** Concurrent user limit depends on physical capacity of database server hosting the repository*

Remote installation

Remote Installation of Enterprise Architect

When deploying Enterprise Architect across a network of workstations there are a number of applications and methods that can be used. Deployment can be carried out using Windows Server tools or deployment software like Microsoft SCCM. This section provides a brief explanation of the methods for using this type of remote installation.

Overview

Enterprise Architect is set up using an MSI installer. You can use this file to do the installation through Windows Server or SMS.

For documentation on the options available for Windows MSI installers see:
http://technet.microsoft.com/en-us/library/cc759262%28v=ws.10%29.aspx

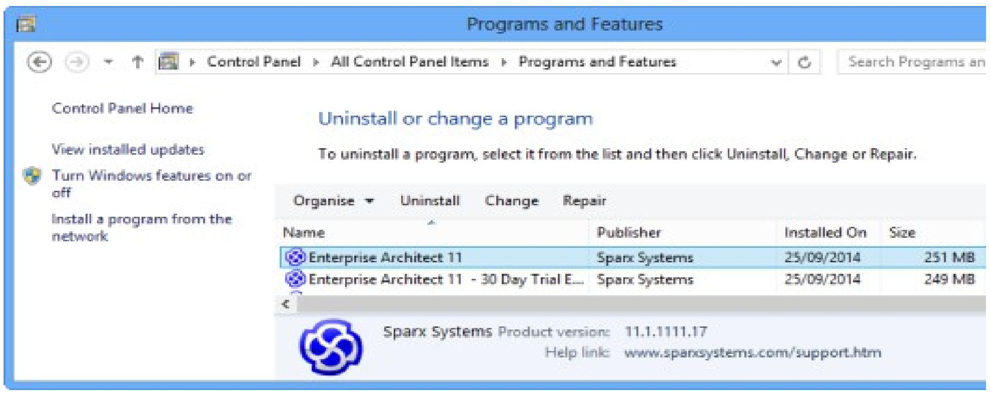
Note: By default Enterprise Architect will install for the Current User. To install for All Users specify :

```
    msiexec /i c:\easetupfull.msi /q allusers=2
```

Checks Post Installation:

Please note that in this example Enterprise Architect was installed for All Users.

1) After installation, as shown in Figure 9, check Windows | Add/Remove Programs for evidence that Enterprise Architect installed successfully.


Programs and Features
Control Panel > All Control Panel Items > Programs and Features
Search Programs an
Control Panel Home
View installed updates
Turn Windows features on or off
Install a program from the network
Organise
Uninstall
Change
Repair

Uninstall or change a program

To uninstall a program, select it from the list and then click Uninstall, Change or Repair.

Name	Publisher	Installed On	Size
Enterprise Architect 11	Sparx Systems	25/09/2014	251 MB
Enterprise Architect 11 - 30 Day Trial E...	Sparx Systems	25/09/2014	249 MB

Sparx Systems Product version: 11.1.1111.17
Help link: www.sparxsystems.com/support.htm

Figure 9: Windows application installer showing the Enterprise Architect installation

2) Check the 'All Users' profile for desktop and start menu items to verify that Enterprise Architect was installed successfully for all users. Based on the operating systems see the following directories:

Windows XP:	C:\Documents and settings\All Users\Desktop
Windows 7+:	C:\Users\Public\Desktop

Remote Floating License Installation

During automated installation of Enterprise Architect, registry entries can be set for each user, giving them access to a floating license key when they start Enterprise Architect. The registry settings differ for the file based and the service based keystores, as described below.

1. Example registry settings for the file based keystore:

```
[HKEY_CURRENT_USER\Software\Sparx Systems\EA400\EA\OPTIONS]
"SKT"=dword:00000000
"SharedKeyFolder"="Y:\\Dev\\Licenses"
"AutoCheckoutEx"=hex:1a,00,00,00
```

2. Example registry settings for the service based keystore:

```
[HKEY_CURRENT_USER\Software\Sparx Systems\EA400\EA\OPTIONS]
"SKT"=dword:00000001
"SSKSAddress"="ssks://pathToKeystoreService"
"SSKSPassword"="service password (encrypted)"
"AutoCheckoutEx"=hex:1a,00,00,00
```

Key Definitions

SKT (Shared KeystoreType)	Specifies the type of keystore to obtain shared keys from. Permitted values are 0x00 (File based keystore) or 0x01 (Service based keystore).
SharedKeyFolder	This value should point to the mapped directory path, or network path, containing the shared sskeys.dat file. This setting is only used if the SKT key has a value of 0x00 (file based keystore). The above example points to a directory on a network drive: Y:\dev\licenses. **Note:** – Be aware of the double back-slashes used in the registry entry – A full UNC path is recommended, for example: “\\DevelopmentServer\EA Licenses” – Enterprise Architect users must have read and write access to this file – If accessing a key-file on a **Novell server** the file path is case-sensitive
SSKSAddress	The ssks address to the Shared Keystore Service endpoint. This setting is only used if the SKT key has a value of 0x01 (service based keystore).
SSKSPassword	If the shared keystore service requires a password, it can be entered into this value. Note that this value is encrypted and cannot be entered in plain text. This setting is only used if the SKT key has a value of 0x01 (service based keystore).
AutoCheckoutEx	Indicates which product keys Enterprise Architect should automatically try to obtain on start-up. Each key is represented by 4 bytes eg: 1a 00 00 00 Where bytes 1-2 are the license code (1a00) and bytes 3-4 are the license type flag (0000). The allowable values are listed below.

License codes for AutoCheckoutEx:

License	**Code**
Enterprise Architect Corporate	0200
Enterprise Architect Ultimate	1a00
Enterprise Architect Business & Software Engineering	1800
Enterprise Architect Systems Engineering	1900
MDG Integration for Visual Studio	0a00
MDG Integration for Eclipse	1400
MDG Link for Visual Studio	0300
MDG Link for Eclipse	0800
MDG Link for Doors	0e00
MDG Technology for SysML	1000
MDG Technology for DDS	1200
MDG Technology for Zachman Framework	1600
MDG Technology for TOGAF	1d00
MDG Technology for UPDM	1b00
RaQuest	0c00

License types for AutoCheckoutEx:
Full License: 0000
Academic License: 0100

Example: If all users were to be allocated both a Corporate license and a Visual Studio Integration license, the registry key value would be:

```
"AutoCheckoutSharedKeyArray"=hex:02,00,00,00,0a,00,00,00
```

Appendix

Unicode Support for .eap files

By default, the .eap file format is a Microsoft Jet 3.5 database format (Access 97). This does not support Unicode characters. For full support of Unicode text, you must use the Jet 4.0 file format (Access 2000).

To enable Unicode operation using Jet 4.0:

1) From the main menu select:
   Tools | Options | General
2) Enable the option: [x] Use JET 4.0
3) Download the Jet 4.0 version of the EABase.eap file from:
   http://www.sparxsystems.com/bin/EABase_JET4.zip
4) Extract this eap file to your local machine
5) Run: **Project | Data Management | Project Transfer**

6) In the **Project Transfer** dialog perform a **File to File** transfer with
   - **Source**: your own file
   - **Target**: the jet 4 eap file download

7) On opening the Jet 4.0 .eap file you should now be able to enter the characters you require then save and open the file again without any data loss.

To ensure all new .eap files created by EA use the Jet 4.0 format, replace the EABase.eap file in your EA install directory with a copy of the file downloaded in the EABase_JET4.zip file.