User Guide

Version 2.0.0

mqarchitect@mqsystems.com

Contents

Introduction

A WebSphere MQ Infrastructure is the 'plumbing' required to allow business applications to exchange information. Building a WebSphere MQ Infrastructure requires careful planning, design, configuration, implementation and a lot of specialised, high value skilled people. BUT…
Most of the time these high value skilled people are spending a lot of time typing scripts, comparing information from one system to another and most of the time first need to find out what's going on or who did what, when… etc…

MQArchitect

MQArchitect has been designed to address all issues outlined above. MQArchitect combines the drawing and visualisation capabilities of Visio (the drawing tool most commonly used by integration architects) with the intelligence required to plan, design and configure a WebSphere MQ infrastructure.

MQArchitect provides you with a single combined repository of information for your complete WebSphere MQ environment, ranging from development to production, including all the information you need to plan, configure, understand and support the architecture.

MQArchitect combines a multi-layered graphical view with the technical detail required to automatically create the actual WebSphere MQ objects.

Moreover, whenever a requirement changes, all supporting material including drawings, documentation and configurations are kept automatically in "sync". Implementing changes becomes a simple task. Should rollback (back out) of changes be required, this is available at the press of a button.
MQArchitect can even cope with changes made in the implemented architecture, outside of MQArchitect. Should such a change be made, your drawings and documentation can be automatically updated.

MQArchitect provides an integrated development environment for integration projects!

Benefits

- Ease Of Use
  - ✓ Visualization of your WebSphere MQ environment/design, helps you close the gap between your business users and IT (a picture says more then a thousand words)
  - ✓ rapid production of WebSphere MQ object configuration
- Save Time, Improve Quality
  - ✓ generate quality WebSphere MQ object configuration and documentation
  - ✓ Deploy from Design, Controlled rollout and rollback of changes
  - ✓ produce on-line, shareable documentation
- Flexibility And Customisation
  - ✓ Using XML and XSL Style sheets
    - re-use the WebSphere MQ configuration information
    - customise layouts to fit corporate standards, the style sheets provide the framework *you* can extend to fit the need
- Secure
  - ✓ Does NOT require the command server to run, uses local `runmqsc` (or native equivalent)

Platform Support

Windows, AIX, Solaris, HP-UX, Linux, i5/OS (via QSH mode) and other systems providing the `runmqsc` WebSphere MQ command. For other platforms like z/OS or NSK you can use the MQSC Script generator to generate the required scripts and feed them into the native runmqsc equivalent.

Minimum Configuration

- ✓ WebSphere MQ Version 5.2 with CSD 4
- ✓ Java 1.3.1 runtime
- ✓ Internet Explorer with MSXML 3.0 or FireFox
- ✓ Visio 2003

Functions Provided by MQArchitect

Iconic Representation of Objects

MQArchitect provides an iconic representation of the entire integration architecture, including not only the WebSphere MQ objects, such as Queue Managers, Queues, Listeners and Channels, but also the Host systems on which WebSphere MQ resides, and the business applications connecting to WebSphere MQ.

Functional Behaviour Templates

MQArchitect maintains the essential properties of all the objects through a simple user interface. These properties are accessed by clicking the icons, the properties are used in the subsequent XML/MQSC generation (see below).

Layered Diagrams

The different levels of complexity required for different specialists are all held within MQArchitect. It is possible to drill down through the layers.

Generation Engine

MQArchitect provides the missing link in current tools. It takes the definitions that have been defined in the visual tool and uses them to publish MQ WebSphere XML/MQSC files that can create the physical integration objects on the required platforms. Using the MQArchitect implementation streamliner tool supported on Windows and UNIX (AIX/Solaris/HP-UX/Linux and i5/OS (via QSH mode)) platforms, you can take it one step further and you don't have to deal with scripts at all. For other systems like z/OS and NSK you can generate the required scripts from the XML/MQSC files.

Implementation Streamliner

MQArchitect recognises that changes are frequent in the real world of development and test. The product therefore interacts directly with your Queue Managers and provides a controlled change process with implementation result reports and instant rollback capabilities.

Reverse engineer existing infrastructure

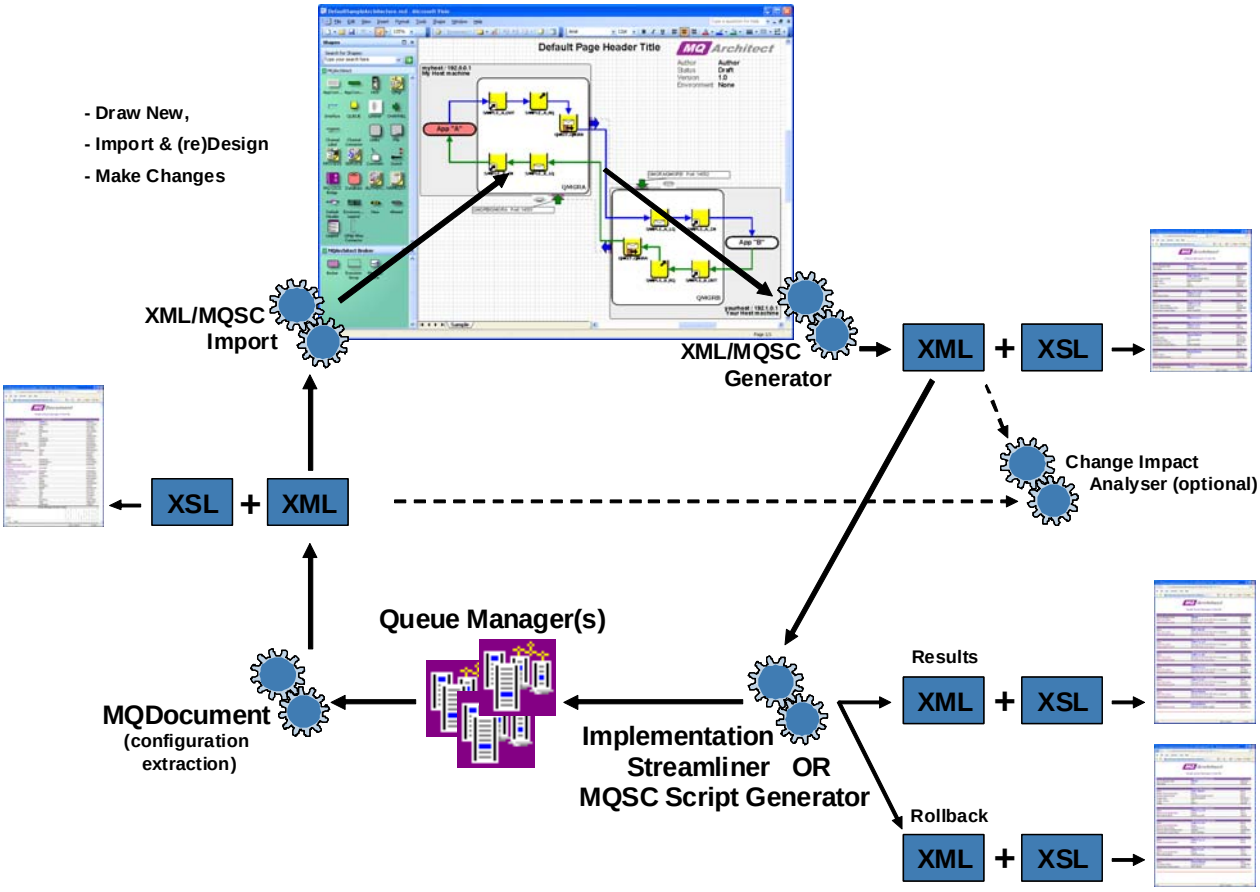
In order to add value to existing implementations, MQArchitect provides reverse engineering capabilities. This facility imports existing WebSphere MQ configuration details and makes them available for review, analysis, change and reporting within MQArchitect.

Change Control and Version Management

In a large environment with multiple developers it is important to control developments carefully. MQArchitect therefore supports the management of change control and version management re-using your existing change control and version management infrastructure.

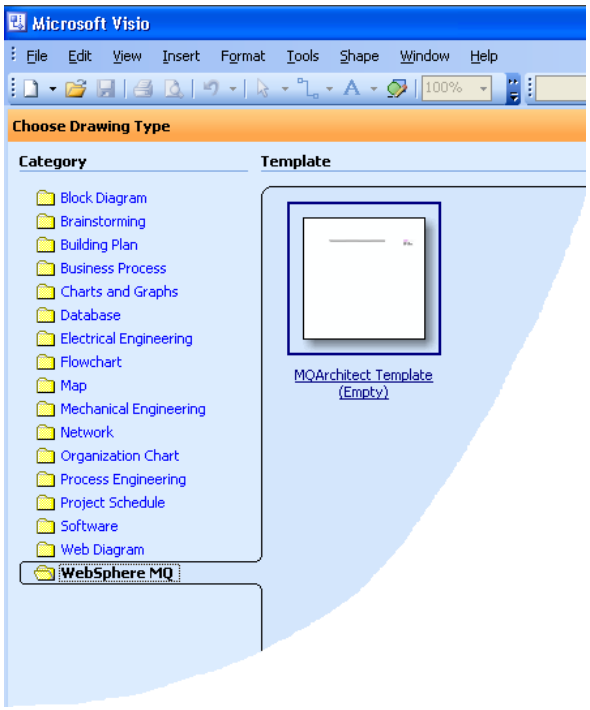
MQArchitect overview

following is an overview of how you would organise your work when working with MQArchitect.



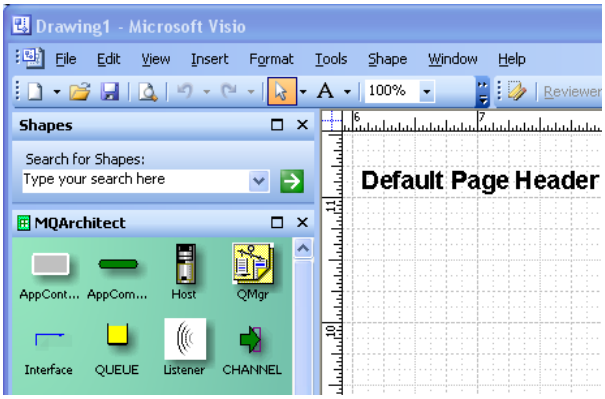
Everything starts and and ends with your design in Visio, either create a design from scratch or use XML/MQSC import objects from SNAPSHOT's. (See also Appendix A "reverse engineering scenario")

MQArchitect Template



To create a design from scratch, start Visio and then open the WebSphere MQ -> MQArchitect Template (Empty).

Start MQArchitect by dragging an object to the drawing canvas.



For *existing* drawings opened with Visio, you can **start** MQArchitect by double clicking any of the objects in the drawing.

To create initial SNAPSHOT's of your *existing* Queue Managers you can use the supplied MQDocument modules. For *new* Queue Managers you can take SNAPSHOT's as well, but if you want to start designing Queue Managers that don't exist yet, default SNAPSHOT's are provided with MQArchitect for a V5, V6 and a V7 Queue Manager.

There is no need to copy these files as MQArchitect will use the default.SNAPSHOT.xml for each new Queue Manager where no *QMgrName.*SNAPSHOT.XML exists in the MQArchitect Data directory.

Drawing your design or repositioning objects after import are regular Visio tasks as you would normally do using other Visio objects. Use the Interface connectors to connect objects.

Where applicable objects will inherit values from their related objects, so you will hardly need to enter information *twice*. A few samples:

- when you connect an Alias Queue object to a Local Queue or Remote Queue, the TARGQ value of the Alias Queue will automatically be set.
- the same applies for a Remote Queue that is connected to a Transmission Queue and then to another Queue on a remote Queue Manager.

The MQArchitect Generate XML/MQSC function will navigate trough your drawing to find the connected objects. (Navigation of connections is done based on color, just like we would do…
Note: Visio uses an RGB color pallete so two blue lines to us, need not be the same RGB color to Visio.

To see which colors match up in a 'stream' you can use:
Interfaces → Show "Interfaces to Select" Window to find out which lines match up according to Visio.
To correct inconsistencies use
Interfaces → Apply "Same Color" to selected interfaces to merge streams or
Interfaces → Get and Set "nearest" available color to selected interfaces to split streams.

For Transmission Queues and Channels, just put a Sender Channel near a Transmission queue and that transmission Queue will 'pickup' the Channel and the other way around.
On the receiving end of Channel, put a receiver on another Queue Manager. If there is a Listener present the port from the listener will be picked up, the Queue Manager name and even the hostsystem name. When the Sender and Receiver are connected via a Channel connector, the Sender will 'know' which CONNAME to use…

When you are 'done' desiging, you can Generate XML/MQSC, the resulting file(s) can then be used to implement the changes using Implementation Streamliner (Windows/AIX/Solaris/HP-UX/Linux and i5/OS (via QSH mode)) or you can generate MQSC scripts for Queue Managers on for example on z/OS or NSK.
Optionally the generated file(s) can be used to do Change Impact Analysis against the SNAPSHOT files you already had (this applies to all platforms).

When you use Implementation Streamliner additionally you get for each Queue Manager a results and a rollback file of the implemented change, you can use these files for change implementation reporting and if needed to rollback the entire change and return to the 'exact' state as before you started the change process.

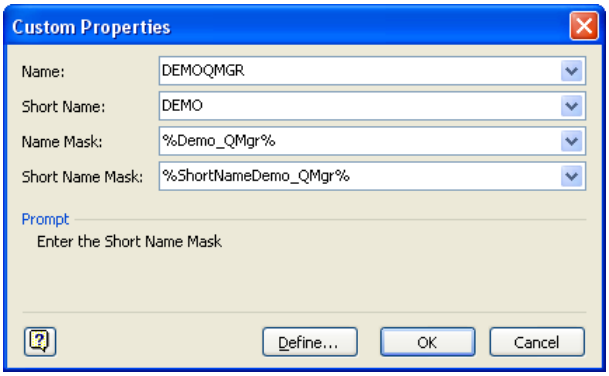
When the changes are implemented succesfully and you are happy with the testing results you can then take new SNAPSHOT's using the MQDocument modules and the cycle is completed.

On to the next design or change…

MQArchitect Advanced topics

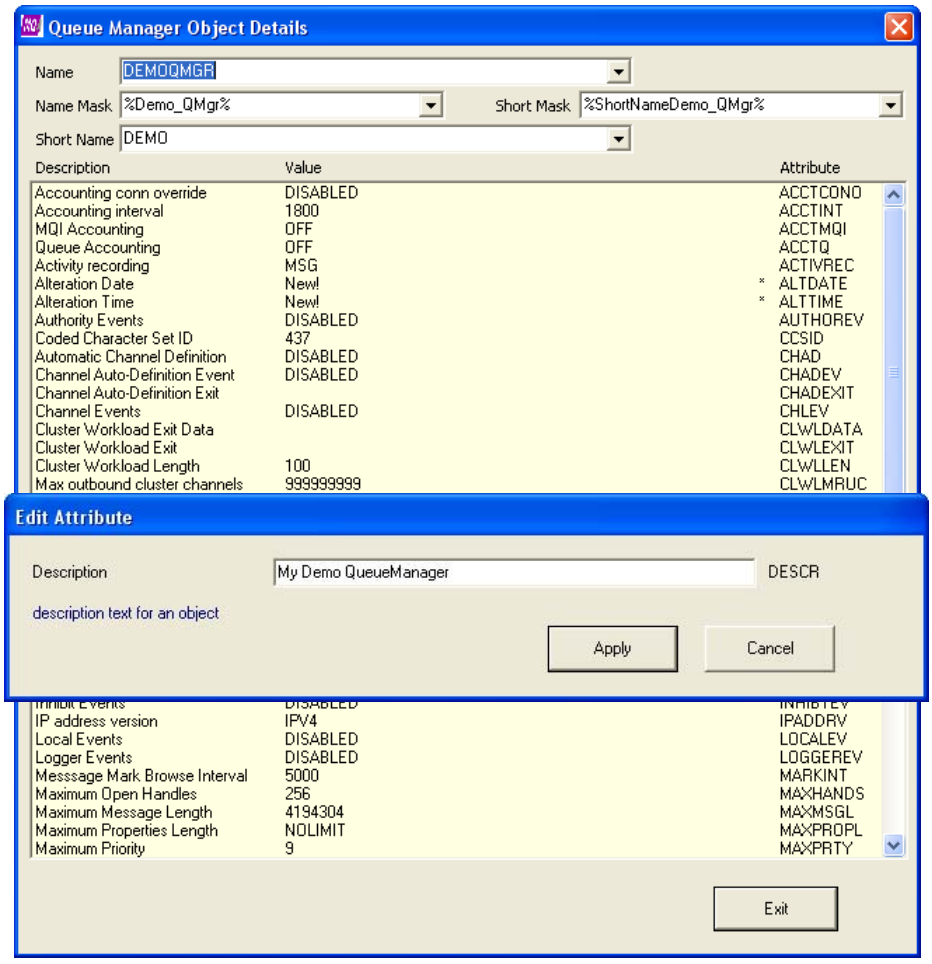
Extended object details / attributes

When you drop a new object onto the Visio canvas, you will be presented a small window (*Custom Properties*) where the object base information can be entered. This information is the minimum MQArchitect needs to know about an object.



When you have given the object a name and re-double click the object, you will be presented the extended *Object Details* window. The information shown in this window is either taken from the default values from that particular object (if it does not exist yet) or from the real object details from the SNAPSHOT's (if the object *does* exist)

Each attribute can be set by double clicking the item in the list window, you will be presented the *Edit Attribute* window (it will appear on top of the Object details list window), where you can change the value of the attribute and Apply the change (pressing Cancel or Enter will not change the value!)



In the Attribute List Window, changes will be marked with '*' next to the short attribute name on the right. When the change is deployed and a new SNAPSHOT is taken, the change will be marked with '**' to indicate there was a change and that it is implemented.

Masks (Name, Description, Port, Switch, CICS Region)

Mask are special variable fields and are present on objects like Host system, Queue Manager, Queues, Channels to deal with changes in values per DTAP environment. The variable names along with their respective values can be loaded or saved from the drawings using the Environments… Menu option. So when looking at the sample Queue manager above you'll notice the Name Mask variable set to %Demo_Qmgr1% (the % signs are part of the variable name and also re-appear in the exported XML file and in the Variable Name column of the supplied sample Excel spreadsheet with Import and Export capabilities for the Mask variables.

In addition to these 'static' variables there are 'dynamic' variables that can be used only in the Name Mask fields. Those variables are +FROMQMGR+, +TOQMGR+, +QMGR+ (if there is no specifice FROM or TO), +FROMSQMGR+, +TOSQMGR+, +SQMGR+ (if there is no specifice FROM or TO). +ENVSH+, +ENVPRF+, +ENVSUF+ for environment specific settings. +SHST+, +FROMSHST+, +TOSHST+ for settings related to Hosts. +FROMSAPP+, +TOSAPP+, +FROMSCMP+, +TOSCMP+ for settings related to Application(Container)s and Application Components. +FROMSBRK+, +TOSBRK+, +FROMSMS+, +TOSMSG+ for settings related to Brokers and Message Flows. These variables can be used in combination with normal text (**not** in combination with other %Variables%). This will allow you to set for example channel names as: +FROMQMGR+/+TOQMGR+ or +FROMQMGR+.TO.+TOQMGR+ or +FROMQMGR+.TO.+TOQMGR+.TCP

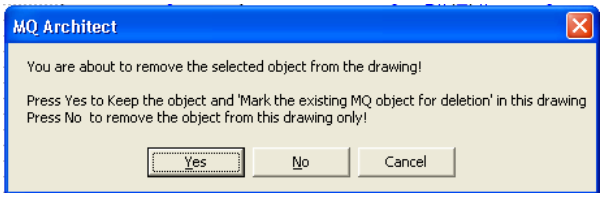
Note: in order to resolve the +FROMQMGR+ or +TOQMGR+ values the objects need to be 'connected' to their counterparts first. If MQArchitect can not resolve a name you will get a message on your screen (depending on how you have set the Help Mode option!)

When you 'set' your Queue Manager names using the %Variables% and then use the +FROMQMGR+ and +TOQMGR+ variables in Channel names or Transmission Queue names, all information will be updated correctly when Update drawing with %Variables% from the Environment XML file file, giving you a maximum of flexibility!

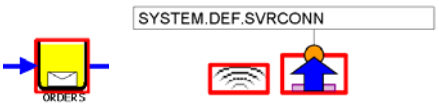
The Mask principle also applies to Description Mask on the Host system object, Port Mask on the Listener object, Switch Mask on the Switch object and CICS Region Mask on the MQ-CICS Bridge object.

Deleting / Removing Objects

When you delete an object or a selection of objects from the drawing, MQArchitect will look if these are a *real* objects existing in the SNAPSHOT files. If the objects *are* real objects, MQArchitect will ask you whether you want to 'Mark the objects for deletion' (answer Yes on the dialog shown) or just want to remove the objects from the drawing and not from the real Queue Manager (answer No on the dialog shown).



When you answer the question with 'Yes' the object(s) selected for deletion that *are real* MQ objects will be marked with a **red** border like this:

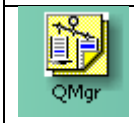


If you want to 'undo' this for a specific object that is 'Marked for Deletion' you can double click the object and 'uncheck' the 'Mark for Deletion' checkbox at the bottom of the Object details window **Note:** if an object is not a *real* object, this checkbox is not shown on the Object details window.

MQArchitect Intelligent icons (and their behaviour)

Application, Interface, Host, Queue Manager

	*Add or Drop to Canvas*	When the Application Container icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Application Container Name (AppContainerName). Additionally the user can select from a dropdown box or enter the Application Container Type (ContainerType). Application Container Short Name (ShortName) can be entered to be referenced/used in 'dynamic' variables +TOSAPP+, +FROMSAPP+. Name Mask (NameMask) and Short Name Mask (ShortMask) can be used to set Application Container Name and Application Container Short Name respectively using 'static' variables (type %var%)
	Display on Canvas	The Application Container icon with Application Container Name visible is displayed.
	Select on Canvas	The object is selectable for moving and resizing.
	Double Click	Prompt for the Application Container Name and Application Container Type, etc… reappears.
	*Add or Drop to Canvas*	When the Application Component icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Component / (sub)Application Program Name (AppComponentName). Additionally the user can select from a dropdown box or enter the Component Mode (Mode), the Component Type (Type) and the Component Programming Language (Programming). Component / (sub)Application Program Short Name (ShortName) can be entered to be referenced/used in 'dynamic' variables +TOSCMP+, +FROMSCMP+. Name Mask (NameMask) and Short Name Mask (ShortMask) can be used to set Component / (sub)Application Program Name and Component / (sub)Application Program Short Name respectively using 'static' variables (type %var1%) The Application Component can be dropped (/moved) on(to) an Application Container and will be added to that shape as a sub shape to form a group, but can also be used as an individual entity in the drawing.
	Display on Canvas	The Application Component icon with Component / (sub)Application Program Name visible is displayed.
	Select on Canvas	The object is selectable for moving and resizing.
	Double Click	Prompt for the Application Component Name, the Component Mode , the Component Type, etc… reappears.
	*Add or Drop to Canvas*	When the Host system icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Host / Machine name (HSNAME), Description (Description). Host Short Name (ShortName) can be entered to be referenced/used in 'dynamic' variables +SHST+, +TOSHST+, +FROMSHST+. Name Mask (NameMask), Description Mask (DescrMask), Short Name Mask (ShortMask) can be used to set Host / Machine and Component / (sub)Application Program Short Name respectively using 'static' variables (type %var1%)

	Display on Canvas	A rectangle, filled with a grey background, with Host / Machine Name and Description visible, is displayed. Always at the back (so object other objects can be put 'on' the Host system).
	Select on Canvas	The object is selectable for moving/resizing.
	Double Click	Prompt for the Host / Machine Name, Description, Host Short Name, etc… reappears
	*Add or Drop to Canvas*	When the Queue Manager icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Queue Manager name (QMNAME) and Short Name (ShortName). These values can be referenced/used in 'dynamic' variables +QMGR+, +TOQMGR+, +FROMQMGR+ for QMNAME and +SQMGR+, +TOSQMGR+, +FROMSQMGR+ for ShortName. Name Mask (NameMask) and Short Name Mask (ShortMask) can be used to set Queue Manager Name and Short Name respectively using 'static' variables (type %var%)
	Display on Canvas	A rectangle, filled with a white background with Queue Manager Name visible is displayed. Always at the back, but on top of a Host system (so object other objects can be put 'on' the Queue Manager).
	Select on Canvas	The object is selectable for moving/resizing.
	Double Click	Prompt for the Queue Manager name, Short Name, etc… reappears
	*Add or Drop to Canvas*	When the Interface icon is selected by the user and dragged to the Visio canvas, the user is prompted to select the interface **color** to identify the interface / stream it belongs to. Interface objects are used to connect Applications to Queues and Queues to Queues (like Aliases to Local or Remote Queues, Remote Queues to Transmission Queues, etc. When more then one connection goes through a Transmission Queue, MQArchitect will follow the Interface with the same **color** to find the next target Queue.
	Display on Canvas	When not connected a dashed line displays with 2 dots interval When connected to one object a dashed line displays with 1 dot interval When connected on both ends, a solid line displays
	Select on Canvas	The object is selectable for moving/resizing/reconnecting.
	Double Click	Text can be added or edited, the text need only be entered once for each 'color' and will be displayed on the Interface and in the 'Show Interfaces to select" window.

Queues and Switch

<table>
<tr><td rowspan="8"> 
</td><td>Add or Drop to Canvas</td><td>When the Queue icon is selected by the user and dragged to the Visio canvas, the user is prompted to select the Object Type from a dropdown list: QLOCAL, QALIAS, QREMOTE and QMODEL and enter the the Name (QNAME).

Additionally the user can enter the Cluster Name (CLUSTER) if this Queue is a Cluster Queue (only valid for QLOCAL QALIAS QREMOTE)

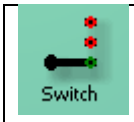
Select from a dropdown list (Usage) if this Queue is a Normal or an XmitQ Queue (only valid for QLOCAL)

Name Mask (NameMask) can be used to set Queue Name using ‘static’ variables (type %var%) or ‘dynamic’ variables (like +QMGR+) in combination with plain text… for example +QMGR+.QUEUE+ENVSUF+

Note: Queues can be dropped on an Interface and the Interface will ‘split’, so the Queue object will be inserted.</td></tr>
<tr><td>Display on Canvas</td><td>The relevant Queue icon with Queue Name visible is displayed. Following table shows the normal Queue icons per Object Type:
<table><tr><td>QLOCAL
  </td><td>QALIAS
  </td><td>QREMOTE
  </td><td>QMODEL
  </td></tr></table>When the Queue is Clustered, a cluster Icon with the Cluster Name will be displayed within the Queue icon
<table><tr><td>QLOCAL
 
</td><td>QALIAS
 
</td><td>QREMOTE
 
</td><td></td></tr></table>Note: When you hover your mouse over the Queue object, the Queue Type and Name will pop-up, this is especially handy when dealing with large topologies and your zoom level is not detailed enough to read the names. When you hover your mouse over the Cluster icon within the Queue object, the Cluster Name will pop-up…

When the Queue is a transmission Queue, QLOCAL with Usage set to XmitQ, a transmission Icon will be displayed within the Queue icon
<table><tr><td>QLOCAL
  </td><td></td><td></td><td></td></tr></table></td></tr>
<tr><td>Select on Canvas</td><td>The object is selectable for moving only.</td></tr>
<tr><td>Double Click</td><td>Prompt for the Queue name, Cluster and Usage, etc… reappears.

When the Queue has a name, then the additional object details are displayed in the Extended object details / attributes</td></tr>
</table>

	*Add or Drop to Canvas*	When the Switch icon is selected by the user and dragged to the Visio canvas, the object is just dropped. The user can connect interfaces to the connector points, the switch will indicate which 'path' the messages should follow, depending on the switch setting 'Normal', 'Test' or 'Unit Test'. Switch Mask (SwitchMask can be used to set / save the Switch position using 'static' variables (type %var1%)
	Display on Canvas	The Switch icon is displayed. When the Switch is set to 'Normal' a straight line is shown connecting the lower connections, the other connections are marked as 'blocked' in red. When the Switch is set to 'Test' a straight line is shown connecting the lower connector and the second connector, the other connections are marked as 'blocked' in red. When the Switch is set to 'Unit Test' a straight line is shown connecting the lower connector and upper connector, the other connections are marked as 'blocked' in red.
	Select on Canvas	The object is selectable for moving only.
	Double Click	Prompt for the Switch setting via a dropdown box.

Listener and Channel Label / Connector

	*Add or Drop to Canvas*	When the Listener icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Port number (Port) and optionally (mandatory for V6) the Listener name (LNAME) Port Mask (PortMask can be used to set / save the Port number using 'static' variables (type %var1%)
	Display on Canvas	The Listener icon is displayed. When there is one Listener on a Queue Manager, all channels will inherit the Port number from the Listener and it will be displayed in the Channel Label. When there are *multiple* Listeners, the Listener that is on the same 'side' (for example the bottom end) of the Queue Manager as the Channel the Channel will inherit the Port number from the Listener on the same 'side'. If there are more then one or none Listeners on the 'same' side, the Listener that is closest in distance (straight line) will be used.
	Select on Canvas	The object is selectable for moving only.
	Double Click	Prompt for the Port number (Port) and optionally (for V6) the Listener name (LNAME) reappears.
	*Add or Drop to Canvas*	When the Channel Label icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Channel name (ChannelName). Since both ends of a channel need to have the same name, it is only needed to enter the Channel name once. **Note:** The Channel Label 'connector' always needs to be connected to either a **Receiver**, **Requester**, **ServerConnection** or **Cluster Receiver** channel, the other end of the channel 'pair' will inherit the Channel name when the 'pair' is connected using a Channel Connector. Name Mask (NameMask) can be used to set Channel Name using 'static' variables (type %var%) or 'dynamic' variables (like +TOQMGR+) in combination with plain text… for example +FROMQMGR+/+TOQMGR++ENVSUF+
	Display on Canvas	The Channel Label icon is displayed. When the Channel Label is connected to either a **Receiver**, **Requester**, **ServerConnection** or **Cluster Receiver** channel and there is a Listener object, the Port number from the Listener is inherited and will be displayed in the Channel Label along with the Channel name.
	Select on Canvas	The object is selectable for moving and resizing.
	Double Click	Prompt for the Channel name (ChannelName) and (NameMask) reappears.
	*Add or Drop to Canvas*	When the Channel Connector icon is selected by the user and dragged to the Visio canvas, it is just dropped onto the canvas. Channel connectors are used to connect channel 'pairs' like for example Sender and Receiver channels, if the connector is connecting two non matching channel icons an error will be displayed.
	Display on Canvas	When not connected a dashed grey line displays with 2 dots interval When connected to one object a dashed grey line displays with 1 dot interval When connected on both ends, a dashed grey line displays When connected to two non matching channels icons the dashed line

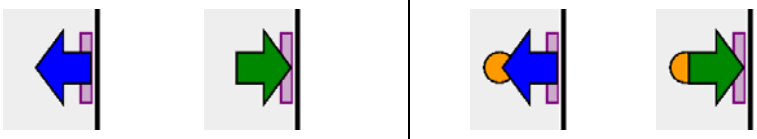
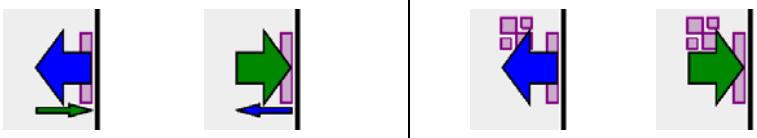
		will be red.
	Select on Canvas	The object is selectable for moving/resizing/reconnecting.
	Double Click	Not applicable, no function.

Channels

<table>
<tr><td rowspan="5"> 
</td><td>Add or Drop to Canvas</td><td>When the Channel icon is selected by the user and dragged to the Visio canvas, the user is prompted to select the Object Type from a dropdown list: SDR, RCVR, SVRCONN, CLNTCONN, SVR, RQSTR, CLUSSDR and CLUSRVCR.

Note: The Name of a Channel pair is determined / set by the Channel Label connected to the relevant RCVR, RQSTR, SVRCONN or CLUSRCVR icon!!!</td></tr>
<tr><td>Display on Canvas</td><td>The relevant Channel icon is displayed.

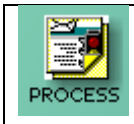
Following table shows the Channel icons per Object Type:

SDR RCVR SVRCONN CLNTCONN
 
SVR RQSTR CLUSSDR CLUSRVCR
  </td></tr>
<tr><td>Select on Canvas</td><td>The object is selectable for moving only.</td></tr>
<tr><td>Double Click</td><td>Prompt for the Object Type reappears,

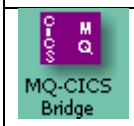
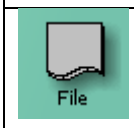
if there is a Channel Label Connected to the relevant RCVR, RQSTR, SVRCONN or CLUSRCVR icon and a Channel Name is set the Extended Object details are displayed for that end of the Channel pair.

If there are both Channel Label and Channel Connector connected to the SDR, SVR, CLNTCONN and CLUSSDR then also for these parts of the channel pair the Extended Object details are displayed</td></tr>
</table>

Process, Namelist, Service and Authinfo

	*Add or Drop to Canvas*	When the Process icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Process Name (PNAME). Name Mask (NameMask) can be used to set Process Name using 'static' variables (type %var%)
	Display on Canvas	The Process icon is displayed.
	Select on Canvas	The object is selectable for moving only.
	Double Click	Prompt for the Process Name (PNAME) reappears.
	*Add or Drop to Canvas*	When the Namelist icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Namelist Name (NNAME). Name Mask (NameMask) can be used to set Namelist Name using 'static' variables (type %var%)
	Display on Canvas	The Namelist icon is displayed.
	Select on Canvas	The object is selectable for moving only.
	Double Click	Prompt for the Namelist Name (NNAME) reappears.
	*Add or Drop to Canvas*	When the Service icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Service Name (SNAME). Name Mask (NameMask) can be used to set Service Name using 'static' variables (type %var%)
	Display on Canvas	The Service icon is displayed.
	Select on Canvas	The object is selectable for moving only.
	Double Click	Prompt for the Service Name (SNAME) reappears.
	*Add or Drop to Canvas*	When the Authentication Info icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Authentication Info Name (ANAME). Name Mask (NameMask) can be used to set Authentication Info Name using 'static' variables (type %var%)
	Display on Canvas	The Authentication Info icon is displayed.
	Select on Canvas	The object is selectable for moving only.
	Double Click	Prompt for the Authentication Info Name (ANAME) reappears.

Utility, Adapter, File, Database and Comment

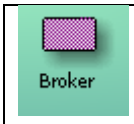
	*Add or Drop to Canvas*	When the Utility icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Utility Name (UtilityName).
	Display on Canvas	The Utility icon is displayed.
	Select on Canvas	The object is selectable for moving only.
	Double Click	Prompt for the Utility Name (UtilityName) reappears.
	*Add or Drop to Canvas*	When the MQ-CICS Bridge icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the MQ-CICS Bridge Name (MQCICSBridge). Additionally the user can enter the CICS Region (CICSRegion) and CICS Transaction name (CICSTransaction). CICS Region Mask (CICSRegionMask) can be used to set CICS Region Name using 'static' variables (type %var%)
	Display on Canvas	The MQ-CICS Bridge icon is displayed.
	Select on Canvas	The object is selectable for moving only.
	Double Click	Prompt for the MQ-CICS Bridge Name (MQCICSBridge), CICS Region (CICSRegion) and CICS Transaction name (CICSTransaction) reappears.
	*Add or Drop to Canvas*	When the File icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the File Name (FileName).
	Display on Canvas	The File icon is displayed.
	Select on Canvas	The object is selectable for moving only.
	Double Click	Prompt for the File Name (FileName) reappears.
	*Add or Drop to Canvas*	When the Database icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Database Name (DNAME). Additionally the user can enter or select from a dropdown box the database type (Type). DataBase Short Name (ShortName) can be entered to be referenced/used in 'dynamic' variable +SDB+. Name Mask (NameMask) and Short Name Mask (ShortMask) can be used to set DataBase Name and DataBase Short Name respectively using 'static' variables (type %var1%)
	Display on Canvas	The Database icon is displayed.
	Select on Canvas	The object is selectable for moving only.
	Double Click	Prompt for the Database name (DNAME) and database type (Type) reappears.
	*Add or Drop to Canvas*	When the Comment icon is selected by the user and dragged to the Visio canvas, the object is just dropped into its position. **Note:** via the first MQArchitect menu option "Hide comments" Comments on all pages can be hidden or unhidden.
	Display on Canvas	A rectangle, filled with a yellow background (like a 'sticky').
	Select on Canvas	The object is selectable for moving/resizing.
	Double Click	Add or Edit the text of the Comment.

Header, Legends , General graphics…. and Magic Connector!

	*Add or Drop to Canvas*	When the Default Header icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Environment (Environment), Page Version (PgVersion), Page Status (PgStatus), Page Author (PgAuthor) and Page Header Title (PgHeader). **Note:** These values are actually stored on Page Level, so can be different for each page in the visio drawing!
	Display on Canvas	The Default Header object with MQArchitect Logo and Page information visible is displayed.
	Select on Canvas	The object is selectable for moving/resizing.
	Double Click	Prompt for the Page Information fields is re-displayed
	*Add or Drop to Canvas*	When the Environment Legend icon is selected by the user and dragged to the Visio canvas, the Environment Legend Shape is dropped to the canvas and can be modified to suit your needs. Environment Description (EnvironmentDescription), Environment (Environment), Environment Prefix (EnvPrefix), Environment Short Code (EnvShort), Environment Suffix (EnvSuffix), Environment Last Loaded, Environment Last Loaded Date Time, Environment Last Saved, Environment Last Saved Date Time **Note:** The shape can be used as a template for your own needs and can be adjusted (delete fields or change order, etc…) **Note:** These values are actually stored on Page Level, so can be different for each page in the visio drawing!
	Display on Canvas	The Environment Legend table object with additional Environment (Page) information visible.
	Select on Canvas	The object is selectable for moving/resizing/etc..
	Double Click	N/A
	*Add or Drop to Canvas*	No specific function, added for graphical illustration of your drawings
	Display on Canvas	
	Select on Canvas	The object is selectable for moving/resizing.
	Double Click	Not applicable, no function.
	*Add or Drop to Canvas*	No specific function, added for graphical illustration of your drawings
	Display on Canvas	
	Select on Canvas	The object is selectable for moving/resizing.
	Double Click	Not applicable, no function.
	*Add or Drop to Canvas*	No specific function, added for graphical illustration of your drawings
	Display on Canvas	
	Select on Canvas	The object is selectable for moving/resizing.

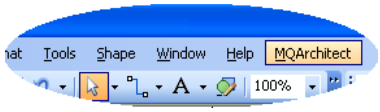
	Double Click	Not applicable, no function.
	*Add or Drop to Canvas*	When the Magic Connector icon is selected by the user and dragged to the Visio canvas, it is just dropped onto the canvas. Magic Connectors are used for special tasks like: • to resolve a QMgr Alias location for a QREMOTE Queue, connect the Magic Connector to the QREMOTE that needs to act as QMgr Alias and connect the other end to the intended QMgr object. • to associate a Transmission Queue with a Sender or Server Channel MCA, normally this is done by putting the Transmission Queue on the 'edge' of a QMgr and the intended Sender or Server Channel MCA right next to it, so they can be associated. If this does not work for you, you can use the Magic Connector instead and connect it to the Transmssion Queue and Sender or Server Channel icon…
	Display on Canvas	A purple dashed line will be displayed.
	Select on Canvas	The object is selectable for moving/resizing/reconnecting.
	Double Click	Not applicable, no function.

Broker, Execution group, Message flow

	*Add or Drop to Canvas*	When the Broker icon is selected by the user and dragged to the Visio canvas, the user can enter (after double clicking!) the Broker Name (Broker). Additionally the user can select from a dropdown box or enter the Broker Type (Type). Broker Short Name (ShortName) can be entered to be referenced/used in 'dynamic' variables +TOSBRK+, +FROMSBRK+. Name Mask (NameMask) and Short Name Mask (ShortMask) can be used to set Broker Name and Broker Short Name respectively using 'static' variables (type %var%)
	Display on Canvas	A rectangle, filled with a purple background. Always at the back, but on top of a Host system and Queue Manager object. (so other objects can be put 'on' the Broker).
	Select on Canvas	the object is selectable for moving/resizing.
	Double Click	Prompt the Broker name (Broker) and Broker Type (Type), etc… reappears
	*Add or Drop to Canvas*	When the Execution group icon is selected by the user and dragged to the Visio canvas, the user is prompted to enter the Execution group name (ExecGrp).
	Display on Canvas	a rectangle, with a transparent background. Always on top of a Broker (so other objects can be put 'in' the Execution Group).
	Select on Canvas	the object is selectable for moving/resizing.
	Double Click	Prompt for the Execution group name (ExecGrp) reappears
	*Add or Drop to Canvas*	When the Message Flow icon is selected by the user and dragged to the Visio canvas, the user can enter (after double clicking!) the Message Flow Name (MsgFlow). Message Flow Short Name (ShortName) can be entered to be referenced/used in 'dynamic' variables +TOSMSG+, +FROMSMSG+. Name Mask (NameMask) and Short Name Mask (ShortMask) can be used to set Message Flow Name and Message Flow Short Name respectively using 'static' variables (type %var1%) The Message Flow can be dropped (/moved) on(to) a Broker.
	Display on Canvas	the Message Flow icon is displayed.
	Select on Canvas	the object is selectable for moving only.
	Double Click	Prompt for the Message Flow Name (MsgFlow), Message Flow Short Name (ShortName), etc… reappears.

MQArchitect Menu Options

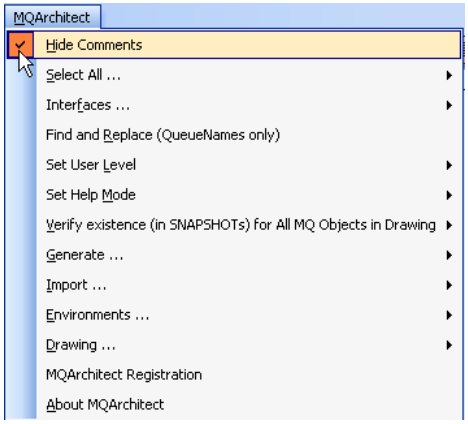
When MQArchitect is started in Visio (by double clicking any MQArchitect icon in a drawing) an extra menu item for MQArchitect functions is added to your Visio menu bar.



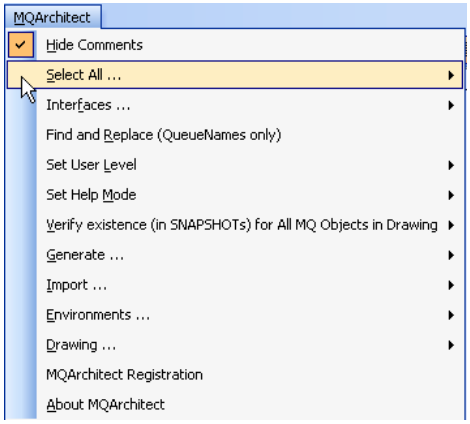
Hide Comments

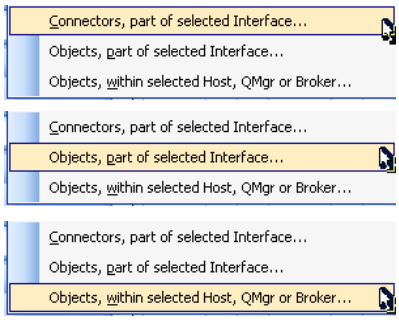
can be used to toggle the display of Comment boxes on and off.

This setting is maintained throughout an MQArchitect "session" for all drawings opened.



Note: An MQArchitect "session" starts from the moment you first start MQArchitect within Visio until the moment you close Visio itself. There is no option to stop MQArchitect within Visio other then closing Visio itself.





Select All…

has several submenus as shown above, these options are to make your working experience with objects easier, if for example you want to select, move, or delete a number of objects belonging together you can use these options.

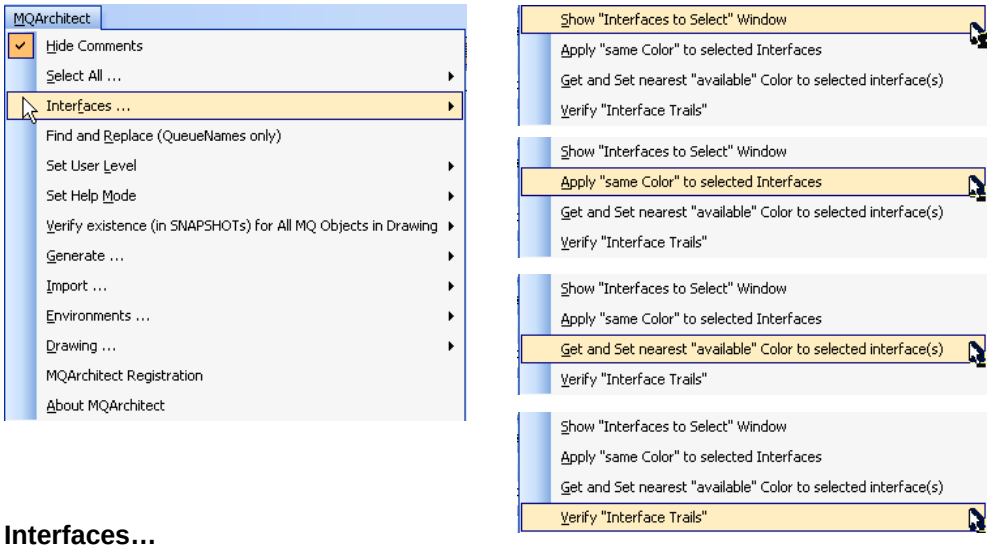
Connectors are the Interface connectors connecting Objects like Applications, Queues, etc.

Connectors, part of selected Interface. If you want to select all Connectors of an Interface, just select any Connector belonging to that interface (all Interface connectors with the same color), go to the Menu option

Objects, part of selected interface If you also want to select all Objects and the Interface connectors themselves.

Note: Transmission Queues are ignored by this selection as they may be part of another stream, so if you want to include them, use the "Ctrl" key and additionally select them manually.

Objects, within selected Host, QMgr or Broker to select all objects within selected object, this is more ore less self explanatory.



Interfaces…

has several submenus as shown above, the "**Show Interfaces to Select Window**" displays a window listing all different interfaces with their respective color and if entered the text belonging to the interfaces. By using the check boxes next to the colors, you can turn the display of these interfaces on (checkbox checked) or off (checkbox unchecked). With this window it should be easier the see which objects belong to which interface "stream" and if colors do not interfere with another (the colors are important as MQArchitect uses them to distinguish which objects belong together, just like we would in a drawing).

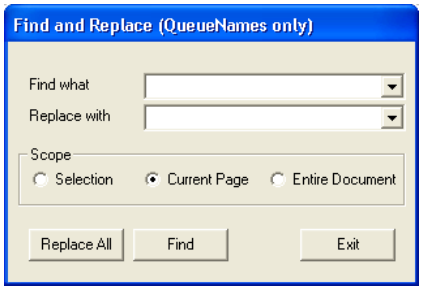
You can use the "**Apply same color to selected interfaces**" to merge 'streams' to make them belong together, all selected Interface connectors will get the same color assigned as the first one selected.

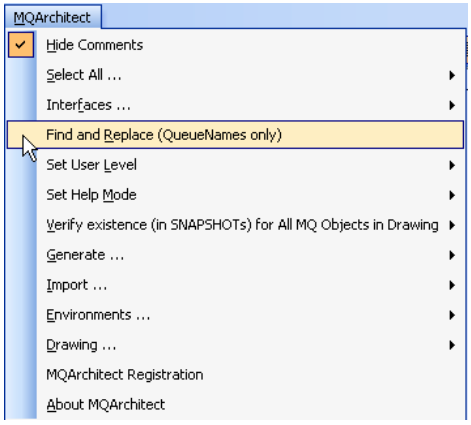
You can use the "**Get and Set nearest available color to selected interfaces**" to split 'streams' apart, but they will still 'look' the same color. Effectively a lookup is done of the first available color in the RGB palette closest to the color of the first selected interface connector and then assign that color to all selected interface connector.

You can use the "**Verify Interface trails**" to see if all Interface connectors are 'connected' to their respective objects.

Find & Replace

can be used to Find and Replace strings or part of strings in Queue Names (only!). If you select this option a window will appear

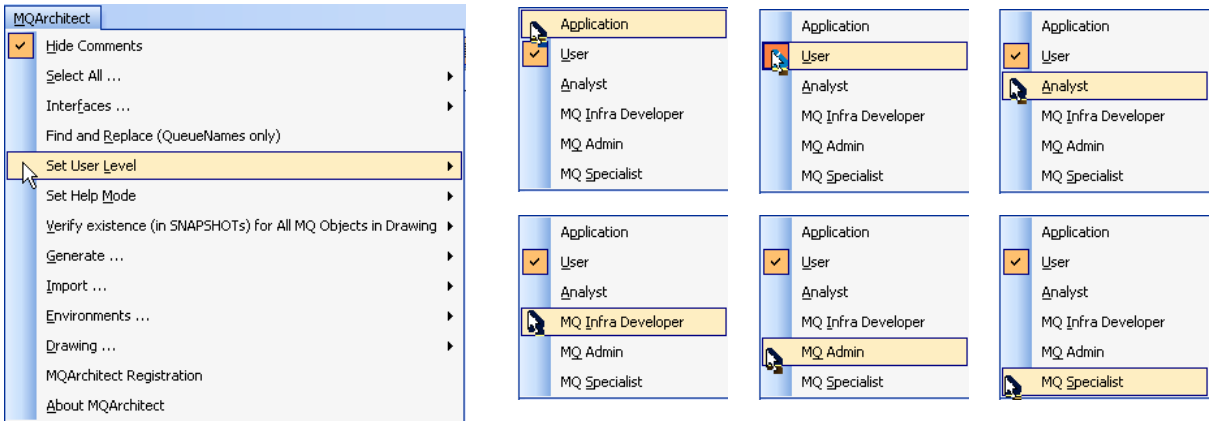




prompting you for the Find string and Replace string.

You can use this function against a Selection of Objects, the Current Page or the Entire document.

Note: used Find and Replace strings are kept in the dropdown boxes throughout the MQArchitect "session".

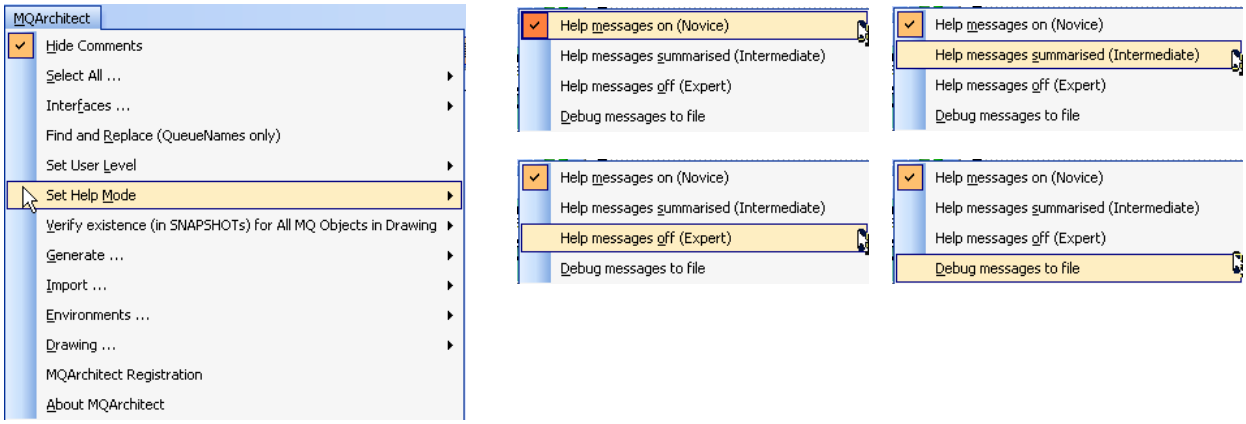


Set User Level

has several submenus as shown above, the selected level will show relevant objects to that level and hide irrelevant objects to that level. This will make your drawings more accessible to each group on their “own” level.

Showing and hiding of objects is “dynamic”, for example a business analyst is only concerned with the input and output queue of an application, everything in between is “irrelevant”, but if there is only one local Queue as input and output it should not be hidden. MQArchitect examines each interface and decides based on the relevant criteria what to hide or show.

Just try it ☺

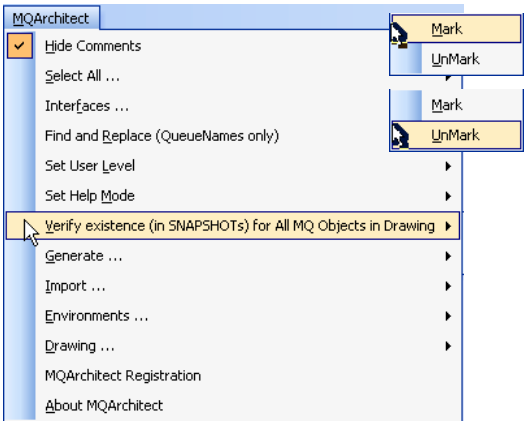


Set Help Mode

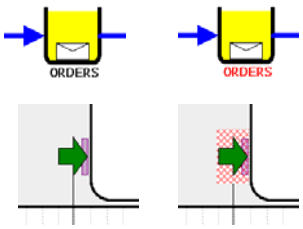
has several submenus as shown above, the selected level will show relevant messages from MQArchitect to the user or not…

Verify existence (in SNAPSHOT's) for Objects in Drawing

has two options: "Mark" and "UnMark".



"Mark" will compare all objects in the drawing (current page only) to the objects in the SNAPSHOT files and if the object cannot be found in the SNAPSHOT, it will mark the objects text by changing the text color from **black** to **red** (if the text color is not **black**, the object will be *ignored*), for channels it will mark the background with red stripes.



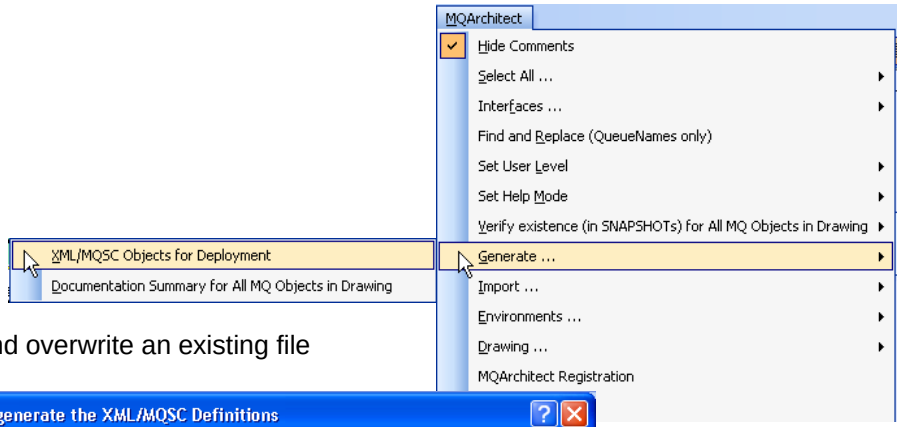
If a SNAPSHOT for a particular **Queue Manager** is not in your data directory no compare can be done, the Queue Manager *border* will be marked as **red** to indicate the SNAPSHOT could not be found for this Queue Manager.

With this function you can easily see which objects are missing from the SNAPSHOT's and if the SNAPSHOT's were recently taken, these objects will also be missing on the real Queue Managers.

"UnMark" will undo all "Marks from **black** to **red**", the backgrounds of channels will return to normal and marked Queue Manager *borders* will also return to normal (black).

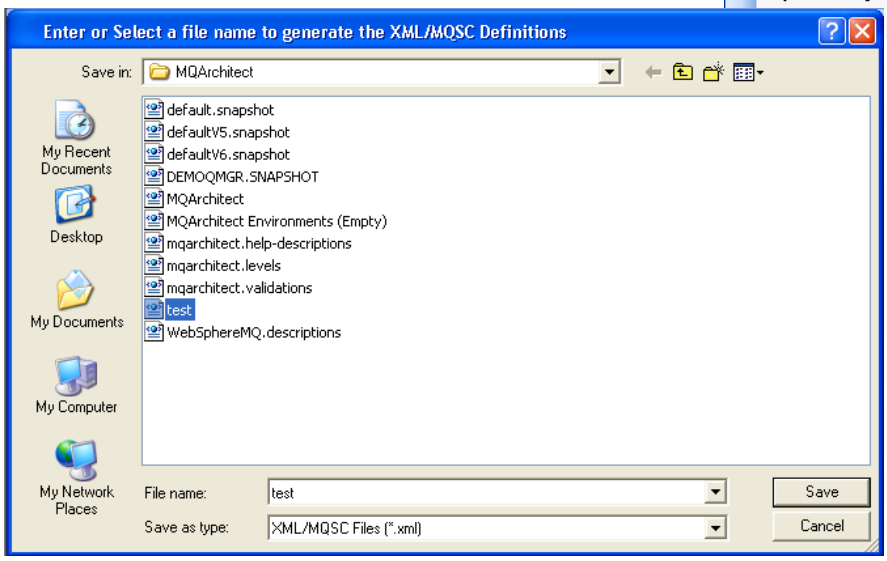
Generate…

has two submenus:

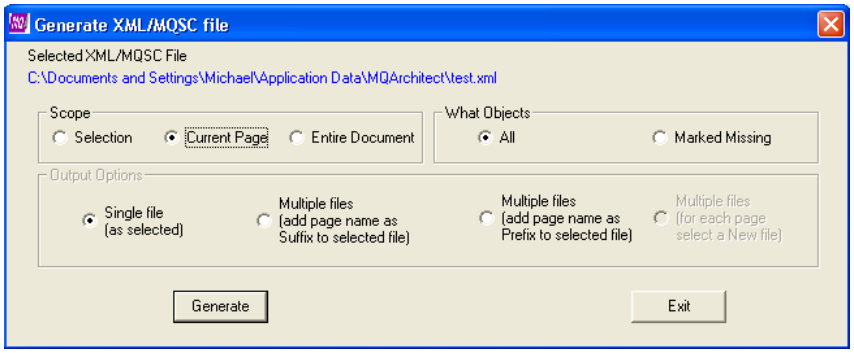


XML/MQSC Objects for Deployment

will allow you to create a *new* file or to select and overwrite an existing file



The file be will be filled with objects and their properties as recorded in the drawing.

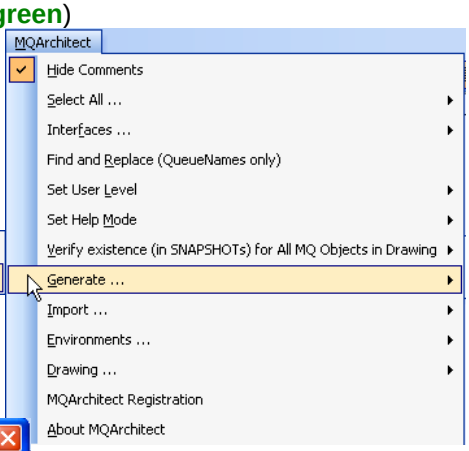


The *Scope* can be set to: Selection, Current Page or Entire Drawing.

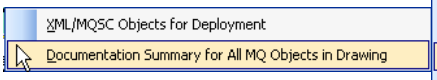
Within the scope you can *indicate* to generate All objects (the ones that are **black** or **red** (if the text color of an object is not **black** or **red**, the object will be *ignored*) or Marked Missing only (the ones that are marked **red**).

You can decide to put all information into one single file, multiple files (with page name as pre or suffix) or enter/select a file name for each page manually (if the scope is set to Entire drawing). All of this is driven through a selection screen (shown above), which will appear when this Menu option is used.

Note: To exclude an entire Queue Manager from the Generation process (because you may not be responsible for it as it is controlled by another department or company), just set the Queue Manager text color to any other color then **black** or **red** (my preference is to use **green**)

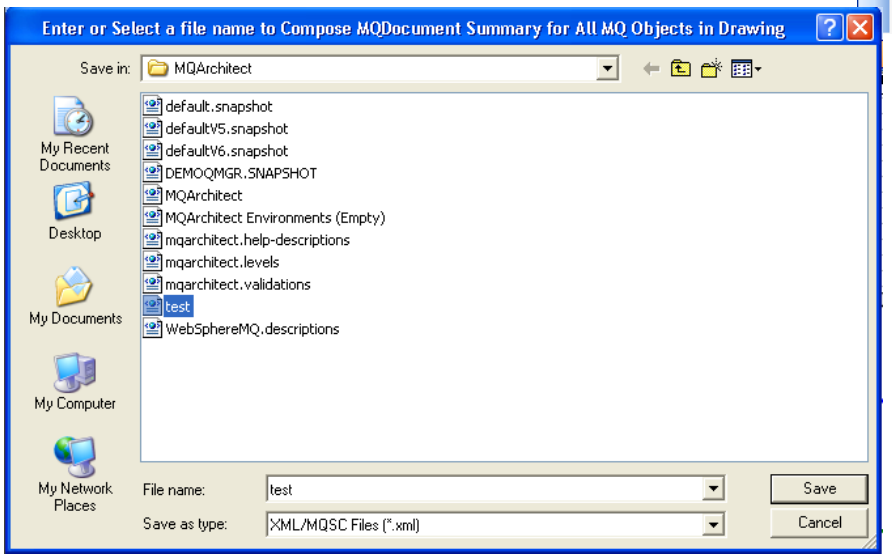


And the other Generate option is:



Documentation Summary for All MQ Objects in Drawing

will allow you to create a *new* file or to select an existing file

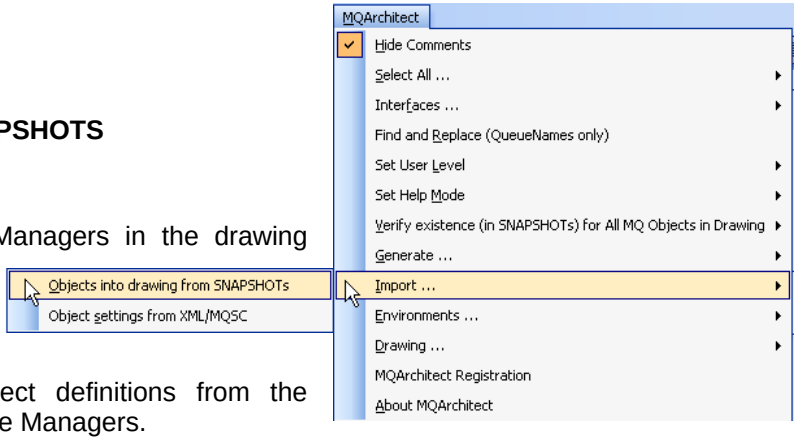


The file will be filled with all objects from the drawing including the information from the SNAPSHOT's about these objects.

This will allow your drawing to act as *a filter or selection mechanism* to generate documentation of all objects and their properties in the current drawing.

Import…

has two submenus:



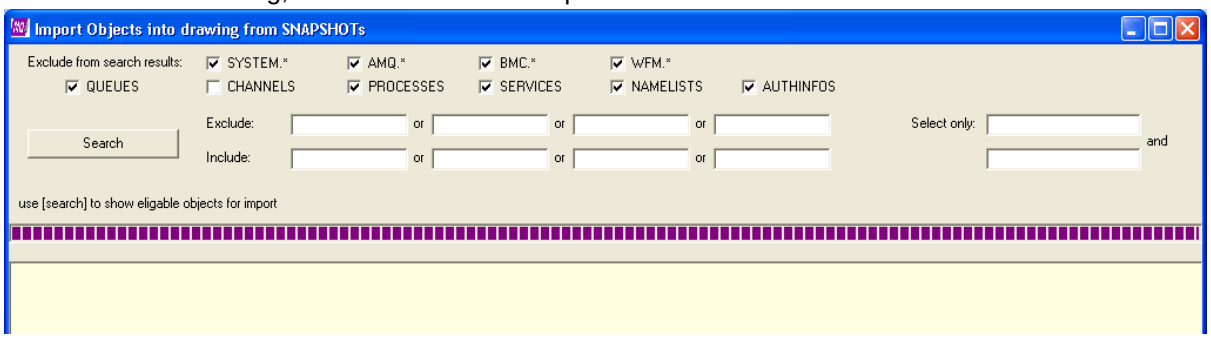
Objects into drawing from SNAPSHOTS

will:

- first build a list of Queue Managers in the drawing *(so to import objects for a certain Queue Manager you need to add that Queue Manager to the drawing first).*
- Then it will collect all object definitions from the SNAPSHOT's for these Queue Managers.
- Next all objects found in the SNAPSHOT's are compared to the ones already in the drawing (all pages of a drawing are searched) (if the object is already in the drawing, there is no need to import it again…).
- Lastly it will look in the **exclude** subdirectory of your data directory for any XML/MQSC files from other drawings, all objects found there will also be excluded from the search.

The result will be an "in memory" list of all objects belonging to the Queue Managers in the drawing, but are not in the drawing *yet* and are not in any other drawing files already.

Still this list can be long, so a search window is presented



By *default* "excluded" from the Search are: Objects starting with: SYSTEM.*, AMQ.*, BMC.*, WFM.*, Queues, Processes, Services, Namelists and Authinfos. Channels are unchecked and included in the Search as this is the best option to start with when you add new Queue Managers.

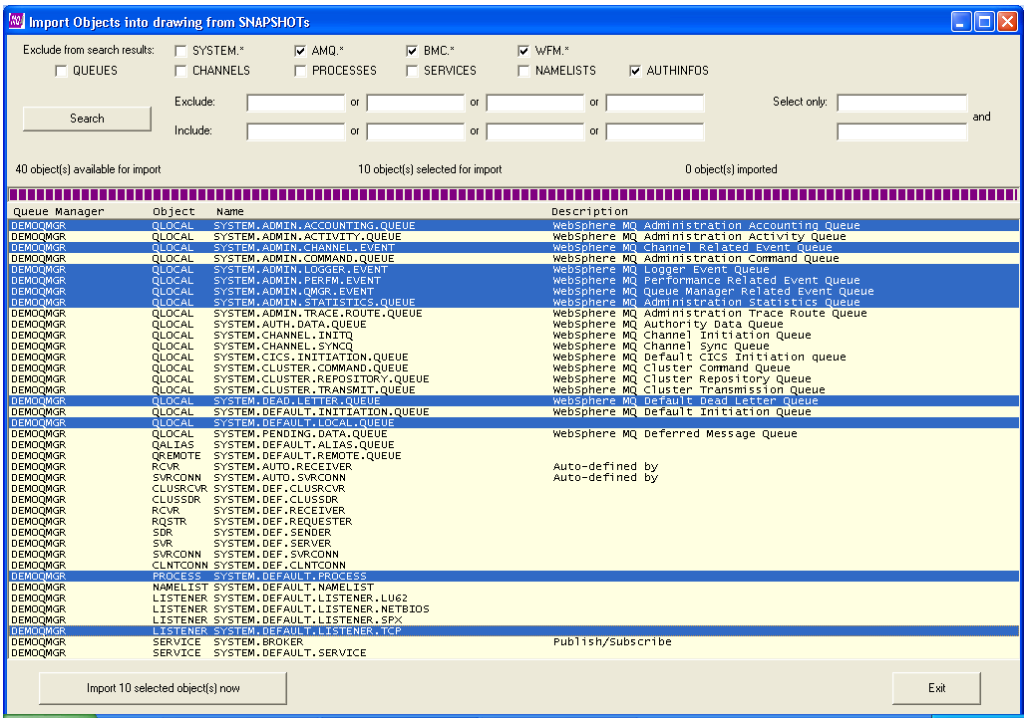
Note: The checkboxes **can** be unchecked to include them in the Search.

Selection fields are supplied to do "*OR excludes*" or "*OR includes*" on object names.

Selection fields for "*select only AND*" values.

Giving you a lot of flexibility to find the right objects in the "pile" ☺

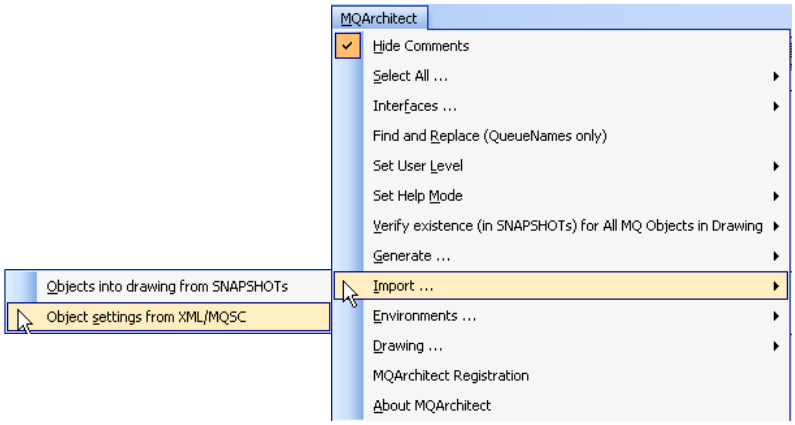
From the Search result list you can select the objects you want to import and hit the Import button…



When the Import is finished, objects will be nicely placed on their respective Queue Managers and connections with their associated objects have been made.

All you need to do is put them in the place you want them to be…

Note: import objects into drawing **can** also be used to **detect** new or "undocumented" objects…

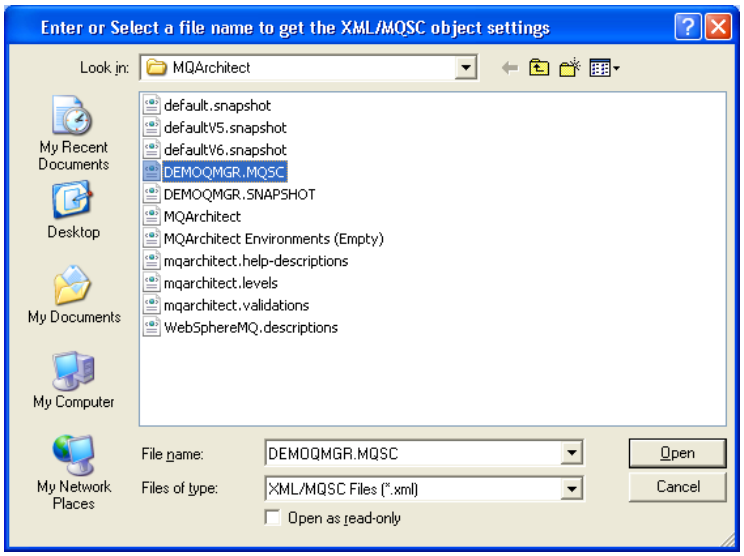


And the other Import option is:

Object settings from XML/MQSC

Will allow you to select an existing file which contains your XML-ified (original) MQSC script values.

Note: with the MQDocument command: **mqdocmqsc** you can convert your original MQSC scripts to XML/MQSC so you can import the object settings you set like PUT, GET, MAXDEPTH, MAXMSGL, DEFPSIST etc.. to the (imported) objects in the drawing.



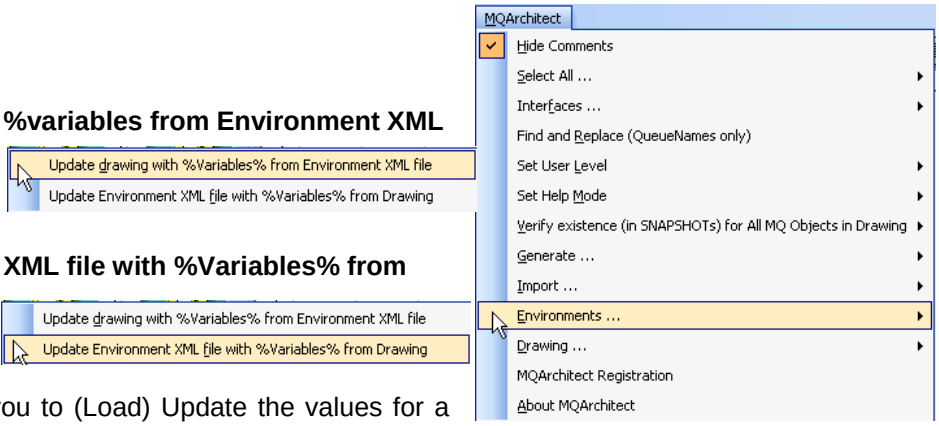
Environments…

has two submenus:

Update drawing with %variables from Environment XML file

and

Update Environment XML file with %Variables% from Drawing.



These options allow you to (Load) Update the values for a different environment into the current drawing, or to (Save) the current values from the current drawing to an Environment XML file.

These options are to save you from having *multiple versions* of the same drawing for multiple environments.

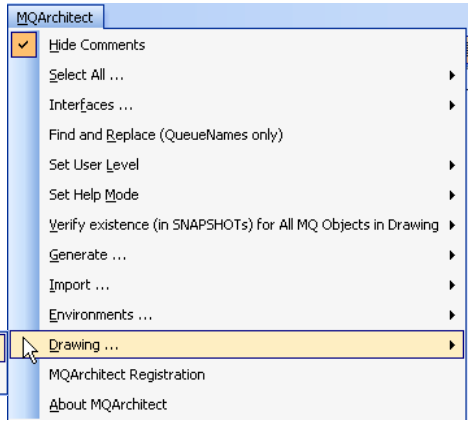
For example if you have a DTAP setup, the names of Host systems, Queue Managers, Transmission Queues etc, can be different, but the overall picture would be the same. To cater for this, you can use *%Variables%* in the **Mask** fields of the respective objects and key (for example Name, Description, Port) fields. The values of these *%Variables%* will be loaded into or saved from the drawing when required.

Note: in a DTAP setup not all objects need to be connected in the same way, for example in Development you can have one Queue Manager serving all Queues, but in Test, Acceptance and Production they would be on different Queue Managers. The differences in routing and setup can be reflected using the *Switch* object which also holds a **Switch Mask** to set the switch value and route based on the value of the *%Variable%* used for that environment.

%Variables% and their values for a particular environment are stored in XML format, which can be used for import into or export from your local configuration management system. If you don't have your own local configuration management system, an Environments Excel sheet is provided with import and export capabilities for the Environment XML files.

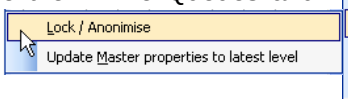
Drawing…

Has two submenus:

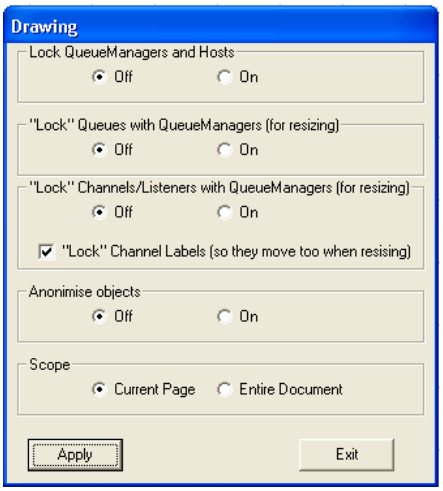


Lock / Anonimise

Provides you with the options to



- Lock the Host systems and Queue Managers from accidental movement (this can happen as you will notice…)
- "Lock" the Queues and/or Channels with the Queue Managers, so you can resize/move them. The Queues and Channels if "Locked" move along with the resized/ moved Queue Manager. After "unLocking" you can reposition any object normally.
- Anonimise the drawing, i.e. hide all the **real** names of objects, but still identify them uniquely, so you can exchange the drawings and the architecture you have designed with peers or partners, without giving out the **real** details…

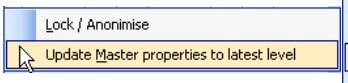


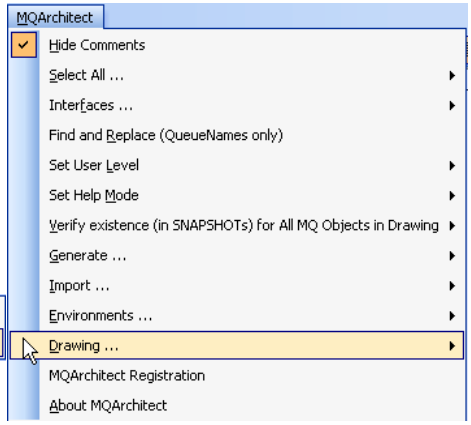
Warning: Anonimise only hides the real object names from *display*, so when exchanging the actual Visio files with someone who has MQArchitect as well, they can turn anonimise off again and display the real values… If you don't want this information to be spread, create PDF documents or screenshots (using Alt+PrintScreen) or use a free tool like Cropper)

And the other Drawing option is:

Update Master properties to latest level

Will allow you to update the Masters in your drawing to the latest level. When this option is used changes to your Master shapes in the Visio drawing will be updated. You will get the appropriate information informing you what updates have happened.

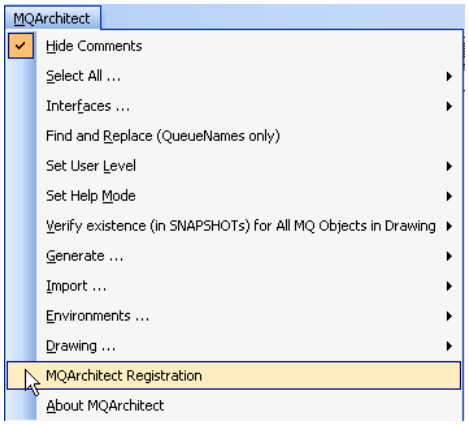




Note: You only need to run this *once* on a drawing after you have received an *update* of MQArchitect.

MQArchitect Registration

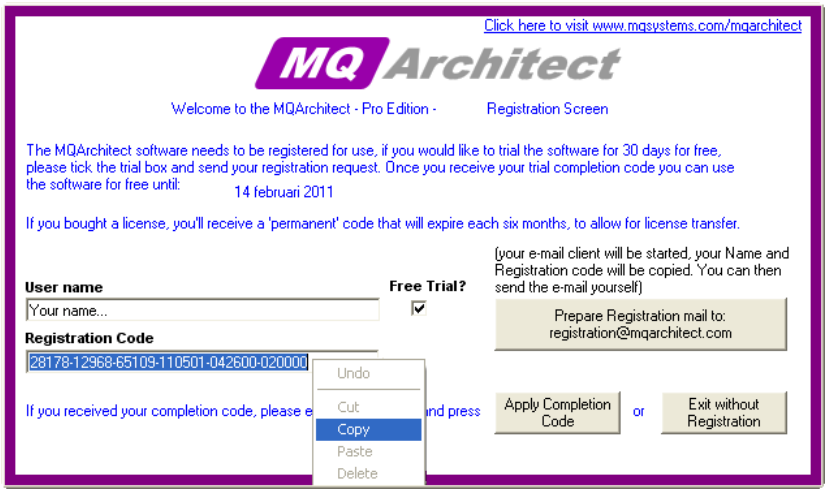
Each installation of MQArchitect generates a unique code and allows you to use the product for up to 30 days in free trial mode.



To register for the 30 days free trial, Enter your User Name, tick the selection box Free Trial and press the prepare registration mail button.

Pressing the button will copy all registration information to a draft e-mail and your e-mail client will be opened, so you can review the details and send the e-mail yourself (the e-mail will **NOT** be sent automatically!)

If for some reason your e-mail client does not automatically start you can manually select the registration code, copy the registration code string and send an e-mail to: *registration at mqarchitect.com*

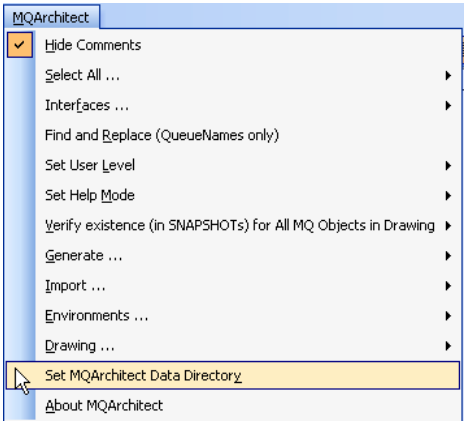


While waiting for the (trial) **registration completion code** you can **not yet** use MQArchitect…

When you have paid the appropriate license fee and annual maintenance fee (15% of the license fee) you will get a ‘permanent’ registration code. Once registered the registration code is valid for 6 months, after that period you need to extend the license (a new registration code is generated) and you have to re-register your copy of MQArchitect.)

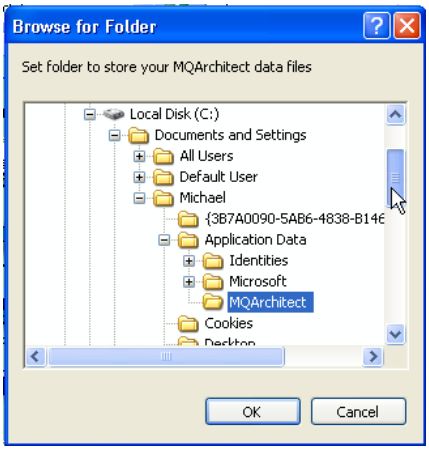
The MQArchitect license is based on user ‘seats’, the process of **re-registration was implemented to deal with ‘floating’ licenses** to transfer licenses within an organisation from one user/computer to another. The operational impact of re-registration will be kept to a minimum.

Set MQArchitect Data Directory



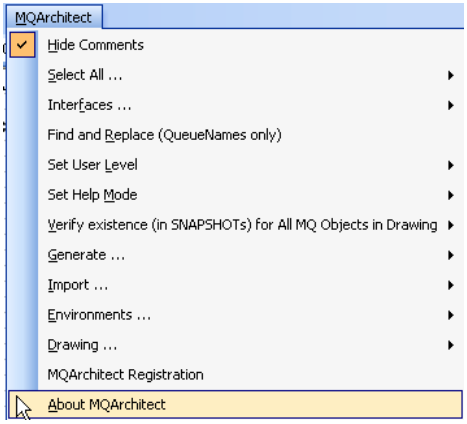
After successful registration, the MQArchitect Registration Menu item will be replaced by the Set MQArchitect Data Directory menu item and will allow you to set the data directory to any directory of your choice.

A directory selection window will appear where you can navigate to the desired directory and you can set it by clicking OK.



The MQArchitect default **User Data Directory** will be set to:
C:\Documents and Settings*UserName*\Application Data\MQArchitect
and the installer creates an MQArchitect Data Directory ***icon*** on your desktop.

About MQArchitect



A any point in time you can click the About MQArchitect menu item. The MQArchitect about window will appear with information about your current license and settings.

Using the about MQArchitect menu option also causes to program to internally 'reset', so if for any reason the program does not respond you can use this option to 'restart' the program. Also if you make changes to some of the MQArchitect base data files (for example if you want to change some of the descriptions or help texts to your local language), these files will also be re-read into memory.

To hide or remove this screen, press **any** key to continue…

MQArchitect Deployment

Implementation Streamliner (xrunmqsc)

```
xrunmqsc -m QMgr -i InputFile [-r ResultsFile] [-b BackOutFile] [-F]
                              [-noSec] [-noXSL] [-noPROC] [-noNUP]
```

`-m`	`QMgr` is the name is the name of the local Queue Manager, whose configuration you want to change/update (note -m ALL , can also be used!)
`-i`	InputFile is the name of the Input XML/MQSC file. This file contains the XML/MQSC definitions you want to implement. This file can contain XML/MQSC definitions for all Queue Managers involved in the change, xrunmqsc will **filter out** the ones that are applicable to the selected Queue Manager using the –m option. . **Note:** This is the file you just created using Generate… from the MQArchitect drawing

Optional parameters:

-r	is the name of the generated Results XML/MQSC file. This file will contain the results of your intended actions. (if no -r is specified, default name [QMgr].RESULTS.XML is used.
`-b`	is the name of the generated BackOut XML/MQSC file. This file will contain the 'BackOut' commands, to completely 'BackOut' what you did. (if no -b is specified, default name [QMgr].BACKOUT.XML is used
`-F`	allow overwrite of existing ResultsFileName and BackOutFileName. **USE WITH CAUTION!** (Results can be unpredictable as most likely you are re-running a command!)
`-noSec`	do **not** process ACL/Security items in input XML files
`-noPROC`	do **not** show 'Processed XML/MQSC Item xxxx of yyyy'
`-noNUP`	do **not** show 'No Update Needed' in Results XML/MQSC file.
`-noXSL`	do **not** include XSL style sheet for easy viewing of the XML files

Change Impact Analyser (xvermqsc)

```
xvermqsc -i InputFile -c CurrentFile
```

`-i`	InputFile is the name of the Input XML/MQSC file. This file contains the XML/MQSC definitions you want to implement. This file can contain XML/MQSC definitions for all Queue Managers involved in the change. **Note:** This is the file you just created from the MQArchitect design
`-c`	is the name of the Current Version XML/MQSC file. This is the file you created on your live system OR is the export file from your current configuration database.

MQSC Script Generator (xgenmqsc)

(DEPRECATED!!! Will be removed in next release!!! → start using mqdocgen instead)

```
xgenmqsc -i InputFile [-c CurrentFile] [-m QMgr]
```

`-i` InputFile is the name of the Input XML/MQSC file. This file contains the XML/MQSC definitions you want to implement.
This file can contain XML/MQSC definitions for all Queue Managers involved in the change.
Note: This is the file you just created from the MQArchitect design

Optional parameters:

`-c` is the name of the Current Version XML/MQSC file. This is the file you created on your live system OR is the export file from your current configuration database.
Note: If NO Current Version file is specified, definitions are generated as if there were no prior definitions

`-m` is the name of the Queue Manager for which you want to generate MQSC definitions if no CurrentFile is specified

Style Sheet

The default style sheet is `mqarchitect.xsl` but an environment variable, `MQA_XSL`, can be set to change the default and/or change the path to the file, e.g.:

```
set MQA_XSL=c:\mystylesheets\mqarchitect.xsl          (Windows)
export MQA_XSL=/mystylesheets/my-mqarchitect.xsl      (UNIX)
```

Licence File

The default licence file is mqarchitect.lic but an environment variable, `MQA_LIC`, can be set to change the default and/or change the path to the file, e.g.:

```
set MQA_LIC=c:\mylicenses\mqarchitect.lic             (Windows)
export MQA_LIC=/mylicenses/my-mqarchitect.lic         (UNIX)
```

MQArchitect Installation

MQArchitect consists of a number of components that need to be installed.

MQArchitect Visio Add-In

The MQArchitect Visio Add-in is a Windows Installer package (msi) that when run, prompts you for the installation location: (When installing a **new** version, **always remove the previous** version first!)

For Visio 2003 this should be: C:\Program Files\Microsoft Office\Visio11 ←(*default* of the installer!)

For Visio 2007 this should be: C:\Program Files\Microsoft Office\Visio12

For Visio 2010 this should be: C:\Program Files\Microsoft Office\Visio14

The MQArchitect default **User Data Directory** will be set to:
C:\Documents and Settings*UserName*\Application Data\MQArchitect
and the installer creates an MQArchitect Data Directory ***icon*** on your desktop.

In the User Data Directory the following files are stored: (You need to copy these files when you change the location of the User Data Directory, **not** available in the trial edition)

mqarchitect.xsl	(mqarchitect stylesheet, **not** supplied with the trial edition!)
MQArchitect.bmp	(used by mqarchitect.xsl)
WebSphereMQ.descriptions.xml	(used by mqarchitect.xsl and the Visio add-in)
default.snapshot.xml	(copy of defaultV5.snapshot.xml, used as default)
defaultV5.snapshot.xml	(default snapshot based on V5 Queue Manager)
defaultV6.snapshot.xml	(default snapshot based on V6 Queue Manager, copy this file to default.snapshot, if you want to use V6 as your default for new Queue Managers)
defaultV7.snapshot.xml	(default snapshot based on V7 Queue Manager, copy this file to default.snapshot, if you want to use V7 as your default for new Queue Managers)

The following files are also installed in above mentioned directory and need **not** be copied when you change the data directory location (**not** available in the trial edition):

mqarchitect.validations.xml	(used by the MQArchitect Visio add-in)
mqarchitect.help-descriptions.xml	(used by the MQArchitect Visio add-in)
mqarchitect.levels.xml	(used by the MQArchitect Visio add-in)

The files:

MQArchitect Environments (Empty).xls	(Sample Excel sheet with Import/Export capabilities for the Environments.xml files)
MQArchitect Environments (Empty).xml	(Sample Environments XML file, contains the same info as MQArchitect Environments (Empty).xls

are provided to show the value of the Environment %Variables% capabilities of MQArchitect.

MQArchitect Deployment modules

MQArchitect deployment modules are 3 java classes:

- Implementation Streamliner (xrunmqsc.class)
- Change Impact Analyser (xvermqsc.class)
- MQSC Script Generator (xgenmqsc.class)

These files need no special installation and can be copied to any location. The MQArchitect java classes need the NetRexxR.jar file to be in the CLASSPATH and need access to the mqarchitect.lic file.

To run these modules without any preparation, put the *class* files, *NetRexxR.jar* and the *mqarchitect.lic* in any directory and run from that directory the following commands:

java -cp .;NetRexxR.jar xrunmqsc ?

java -cp .;NetRexxR.jar xvermqsc ?

java -cp .;NetRexxR.jar xgenmqsc ?

The license file location can be set using an environment variable, see also License File for instructions.
Note: on **UNIX** systems the cp / CLASSPATH **separator** is **:** and not **;**

MQDocument modules

MQArchitect **reuses** the MQDocument modules. Installation of these (java) modules is documented in the MQDocument Package.

Appendix A: “reverse engineering scenario”

The following scenario describes the steps to quickly reverse engineer an existing WebSphere MQ infrastructure.

Start by selecting a number of Queue Managers you want to “draw the picture” for.

1. Take SNAPSHOTS of these Queue Managers using the MQDocument tools.

For Windows/UNIX Queue Managers:

Put the files NetRexxR.jar, MQDocument.jar and mqdocument.lic on the machines where the Queue Managers run.

For Windows use the command: **java -cp .;NetRexxR.jar;MQDocument.jar mqdocument -m ALL**

For UNIX use the command: **java -cp .:NetRexxR.jar:MQDocument.jar mqdocument -m ALL**

Note: on UNIX systems the cp separator is **:** and not **;**

For other platforms like z/OS or NSK use the mqdocclient (if you have MO72) or use mqdocmqsc (if you have MS03 SaveQmgr files) or use mqdocmakedef (if you have MAKEDEF files of these Queue Managers. For *full instructions* on these commands see the **MQDocument UserGuide**.

2. Put the SNAPSHOTS in the MQArchitect Data Directory

3. Open Visio -> WebSphere MQ -> MQArchitect Template

4. Drop a number of Queue Managers from the template to the drawing canvas and give them the names of the Queue Managers for which you put the SNAPSHOT files in your data directory.

5. Go to the MQArchitect Menu -> Import -> Objects into Drawing from SNAPSHOTS

6. Use the default setting and Import Channels only, this will also import any V6 Listener objects and if applicable: Transmission Queues. Only objects related to the connectivity between the selected Queue Managers are imported. So if you have one HUB Queue Manager with hundreds of channels and transmission Queues, only the ones related to the Queue Managers in the picture are imported.

8. From the list select the objects you want to import using Shift+Select or Ctrl+Select or Ctrl+A to select ALL. And import the objects by pressing the “Import nnn selected object(s) now”

This should now give you a basic overview of the connectivity objects related to the connections between these Queue Managers.

7. Next check the “CHANNELS” and uncheck “QUEUES”, to narrow your search of applicable objects use “*OR excludes*” or “*OR includes*”.

8. From the list select the objects you want to import using Shift+Select or Ctrl+Select or Ctrl+A to select ALL. And import the objects by pressing the “Import nnn selected object(s) now”

This should now give you an overview of the selected objects related to the connections between these Queue Managers.

9. Use the Interfaces -> Show “Interfaces to select” to show which “streams” exist. You can check and uncheck the items in the list to show or not to show the “stream”.

Since all imported links are a “shades” of blue you may want to change some of these colors.

10. in the Show “Interfaces to select” pick a stream you wish to change the color for and click in the “colored” box, a window will popup showing the current color, change it to what you want and press “OK”, all connectors with the “old” color will now change to the “new” color.

11. If you still have the original MQSC scripts that you used to create these Queue Managers and their objects, you can use the MQDocument module: mqdocmqsc to convert the scripts to XML/MQSC. Once you have converted the MQSC to XML/MQSC you can use these files to import the object settings from XML/MQSC

Values like MAXDEPTH, MAXMSGL, PUT, GET, etc will be taken from these files and added to the objects in your MQArchitect drawing.

Once you have imported all object settings, the information in the MQArchitect drawing should match your actual configuration.

12. use Generate … XML/MQSC Objects for Deployment to Generate the XML/MQSC file from the drawing.

13. To verify if this is true, use the Change Impact Analyser and use the Generated XML/MQSC file as InputFile and use the SNAPSHOTs of your existing Queue Managers as CurrentFiles.

```
java –cp .;NetRexxR.jar xvermqsc -i InputFile -c CurrentFile
```

so if you generated: mytestdesign.xml

containing Queue Managers: QMGRA and QMGRB

you would run:

```
java –cp .;NetRexxR.jar xvermqsc -i mytestdesign.xml -c QMGRA.SNAPSHOT.XML

java –cp .;NetRexxR.jar xvermqsc -i mytestdesign.xml -c QMGRB.SNAPSHOT.XML
```

14-A. If these Queue Managers exist on a Windows/AIX/Solaris/Linux platform, you can use the Implementation StreamLiner to apply any changes you have seen (and agree with!)

Put the file mytestdesign.xml to the machines involved (including the files: NetRexxR.jar xrunmqsc.class and mqarchitect.lic) and run:

```
java –cp .;NetRexxR.jar xrunmqsc -i mytestdesign.xml –m QMGRA

java –cp .;NetRexxR.jar xrunmqsc -i mytestdesign.xml –m QMGRB
```

14-B. If these Queue Managers do not exist on a Windows/AIX/Solaris/Linux platform, you can use the MQSC Script Generator to generate the MQSC Scripts and apply the changes you have seen (and agree with!) by running the scripts on the platform later.

To Generate the MQSC Scripts run:

```
java –cp .;NetRexxR.jar xgenmqsc -i mytestdesign.xml –m QMGRA

java –cp .;NetRexxR.jar xgenmqsc -i mytestdesign.xml –m QMGRB
```

This will Generate the scripts with **no knowledge of what is there or not!**

If you want the MQSC Script Generator to **take into account what is there already** use:

```
java –cp .;NetRexxR.jar xgenmqsc -i mytestdesign.xml -c QMGRA.SNAPSHOT.XML

java –cp .;NetRexxR.jar xgenmqsc -i mytestdesign.xml -c QMGRB.SNAPSHOT.XML
```

Note: if you add the current directory or the directory where you put the .class and .jar files into your **CLASSPATH** you can run the above commands without **-cp .;NetRexxR.jar !!!**