Implementing Forefront Identity Manager 2010

Student Manual

Module 2: The Synchronization Service Manager

Table of Contents

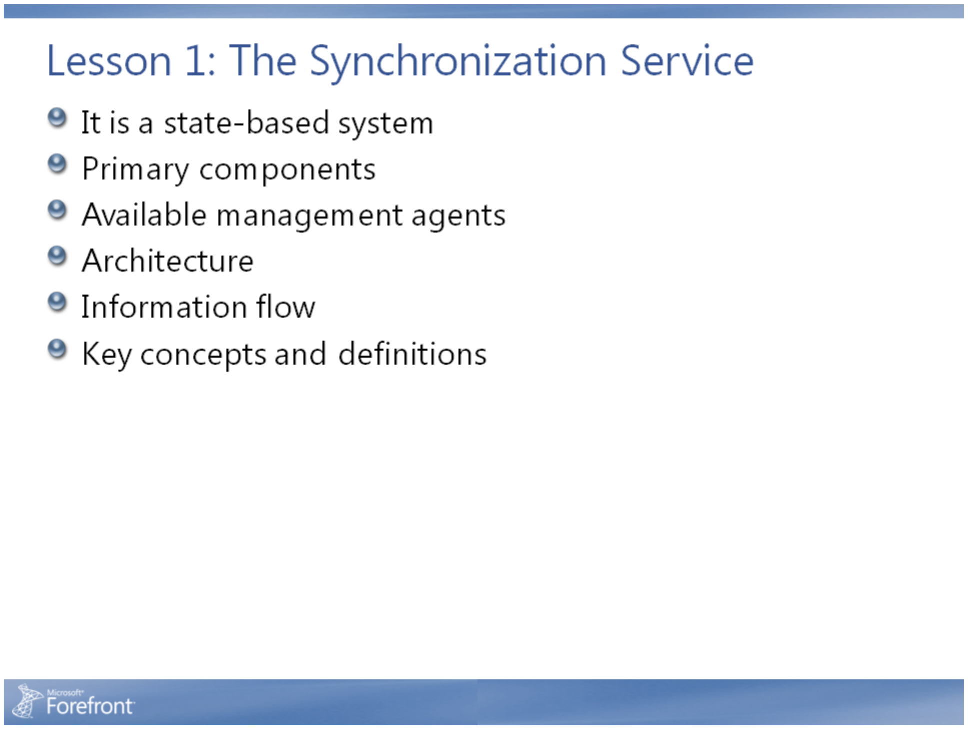
Module 2: The Synchronization Service Manager

Module Overview

This module introduces the synchronization service and the Synchronization Service Manager (the user interface for managing the service).

The biggest topic here is handling management agents (MAs), and a good deal of time is spent on that topic, followed by a lab mostly on that topic.

Lesson 1: The Synchronization Service

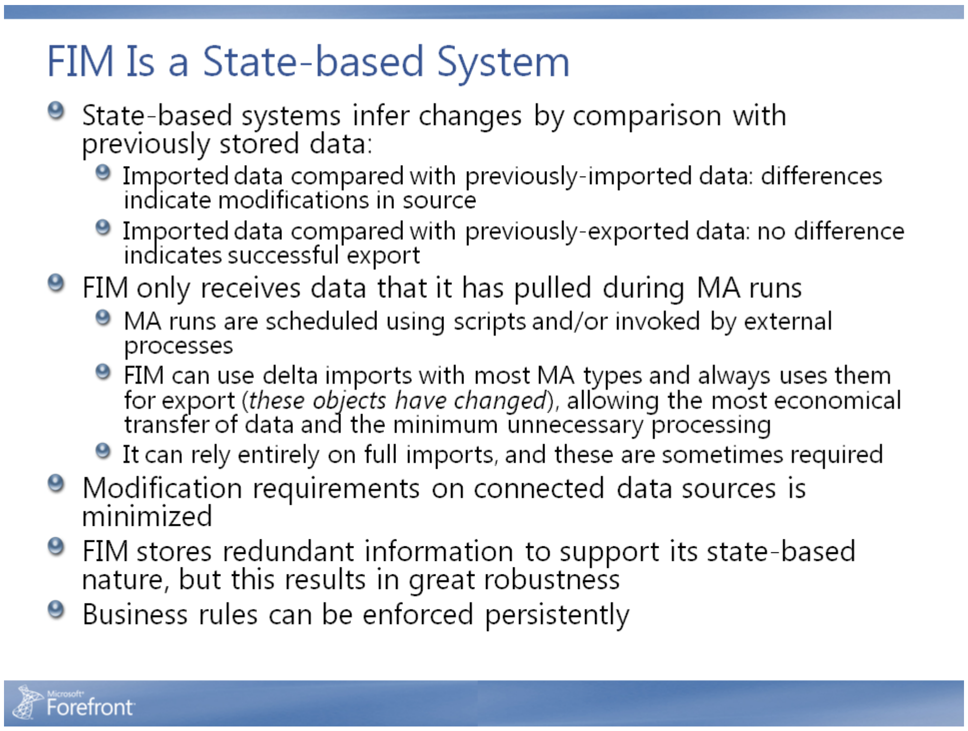


Lesson 1: The Synchronization Service

- It is a state-based system
- Primary components
- Available management agents
- Architecture
- Information flow
- Key concepts and definitions

This lesson introduces the synchronization service (there is further coverage in the next module and deeper coverage in a more advanced course). It covers key principles, concepts, nomenclature, and so on.

FIM Is a State-based System



FIM Is a State-based System

- State-based systems infer changes by comparison with previously stored data:
  - Imported data compared with previously-imported data: differences indicate modifications in source
  - Imported data compared with previously-exported data: no difference indicates successful export
- FIM only receives data that it has pulled during MA runs
  - MA runs are scheduled using scripts and/or invoked by external processes
  - FIM can use delta imports with most MA types and always uses them for export (*these objects have changed*), allowing the most economical transfer of data and the minimum unnecessary processing
  - It can rely entirely on full imports, and these are sometimes required
- Modification requirements on connected data sources is minimized
- FIM stores redundant information to support its state-based nature, but this results in great robustness
- Business rules can be enforced persistently

The FIM synchronization service is a state-based system. There are advantages to this (particularly robustness) as well as potential disadvantages (extra processing and storage), but the actual result is a very effective and flexible compromise.

State-based Systems

FIM stores a binary object called a *hologram* for each external object about which it knows; this hologram represents the current state view of the data stored in each data source. During a subsequent import of the data from the data source, the state of the imported object data is compared with the previous known state. If any differences are detected between the two (for example, the values for the Job Title attribute do not match, or a new or missing object is detected), a change is inferred and the change is handled by the Sync Engine and propagated through the metadirectory if appropriate.

The same kind of idea is used for:

- **Export** – Changes propagated through the metadirectory are compared with what is understood to be in the data source, so that a set of exports can be generated.
- **Export confirmation** – Re-imported source data is compared to what was exported, to confirm that it got there.

The FIM Synchronization Service Pulls Data

In a deployed system, MA runs are invoked by scheduled scripts which are run either on a scheduled basis or in response to external events (perhaps an HR system could invoke a run to ensure that a recently terminated person has their accounts disabled urgently).

FIM then asks for data—it is a pull system—which avoids the need for a push agent on each connected data source. However, FIM can work with delta imports—imports of only those objects that have changed (and, as it happens, exports are always delta in nature). Some data sources support this already, others may be able to with some modification, yet others simply can't support this feature. Where deltas can be used, there are considerable savings in processing time (traffic and state comparisons).

If required, FIM can work entirely with Full Imports, minimizing the intrusion on data sources. Sometimes it is necessary to use a Full Import (for example when recovering from a data source, Microsoft® SQL Server® store or connectivity failure).

Storage and Robustness

This architecture does involve additional storage, because FIM keeps a hologram for each data source, representing a real world object, in addition to the combined metadirectory representation. It is also very robust in the sense that a communication or storage failure will usually only require fixing the immediate problem and restarting the process of checking the states of all objects in all data sources. (More complicated disaster recovery situations will require more careful handling.)

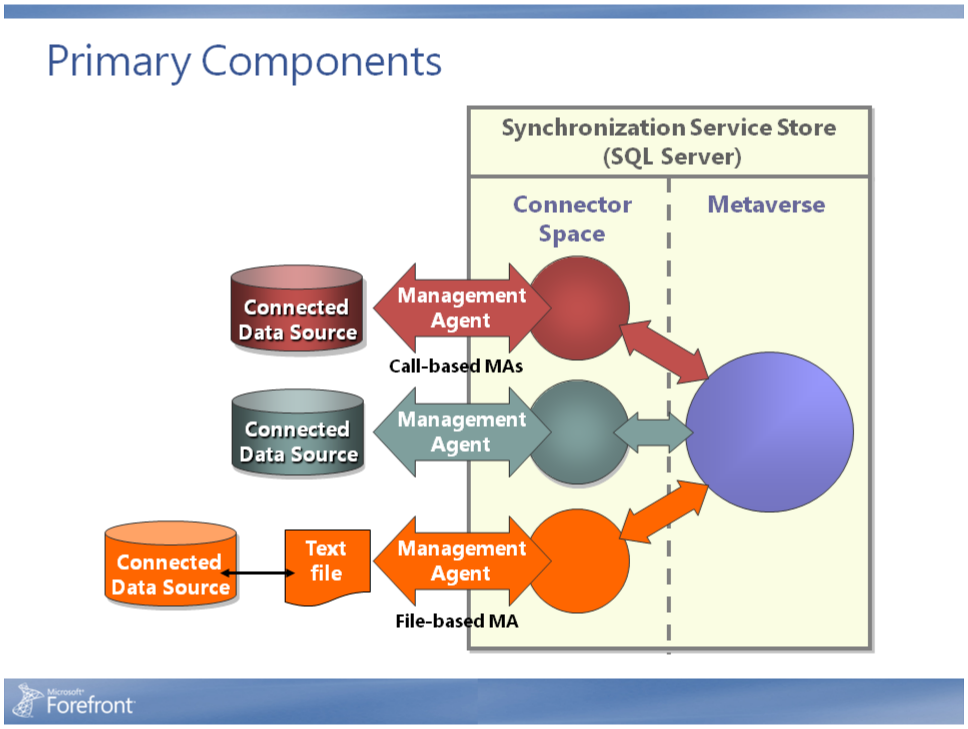
Up-to-date Info

FIM is not real time; few metadirectories are. The answer to the question of how up-to-date is that it depends on many factors, including processing power and intelligent configuration. Some implementations will require that data can be processed as fast as it is changing; in other situations overnight processing may be perfectly acceptable.

Enforcement

Another positive feature of a state-based system is that business rules are enforced continuously. The simplest way to explain this is by example: if an account is supposed to exist for a new identity, it is provisioned; if it is arbitrarily deleted (when it shouldn't be) then, provided FIM has been appropriately configured, it will detect that the state has changed and again enforce the business rule by re-provisioning it. This equally applies to an account being disabled—if that is the intended state, FIM can ensure that it is in this (inactive) state.

Primary Components



The Primary FIM Components Are:

- Metaverse
- Connector space
- Connected data sources
- Management agents

Connected Data Sources

A connected data source (or just a data source, or even just a source) is a data repository that contains identity data to be integrated within FIM. Sources can be enterprise directories, NOS directories, databases, or data in flat files, such as LDIF, DSML, or delimited text.

Management Agents

An MA manages the data associated with a specific data source. The MA not only connects to the data source, but is responsible for managing the flow of data (inbound and outbound). There is at least one MA for each data source.

For many MAs, FIM communicates directly with the data source—these are call-based. For others, an intermediary file is used, such as AVP, LDIF, or fixed-width—these are file-based MAs. Call-based sources are preferable, file-based MAs are usually used as a last resort.

In some cases, the situation may be more complex—there may be no MA specifically for the data source, the data source may, for example, support a mixture of file- and call-based activities, so that a simple file-based MA is not feature-rich enough. The extensible connectivity MA allows a developer to create code to handle almost any data source. (Extensible connectivity MAs are beyond the scope of this course.

Metaverse

The metaverse (MV) is a set of tables within FIM that contain the integrated identity information from multiple connected sources. All identity information about a specific person or object, which is stored in multiple connected sources, is synthesized into a single entry in the MV.

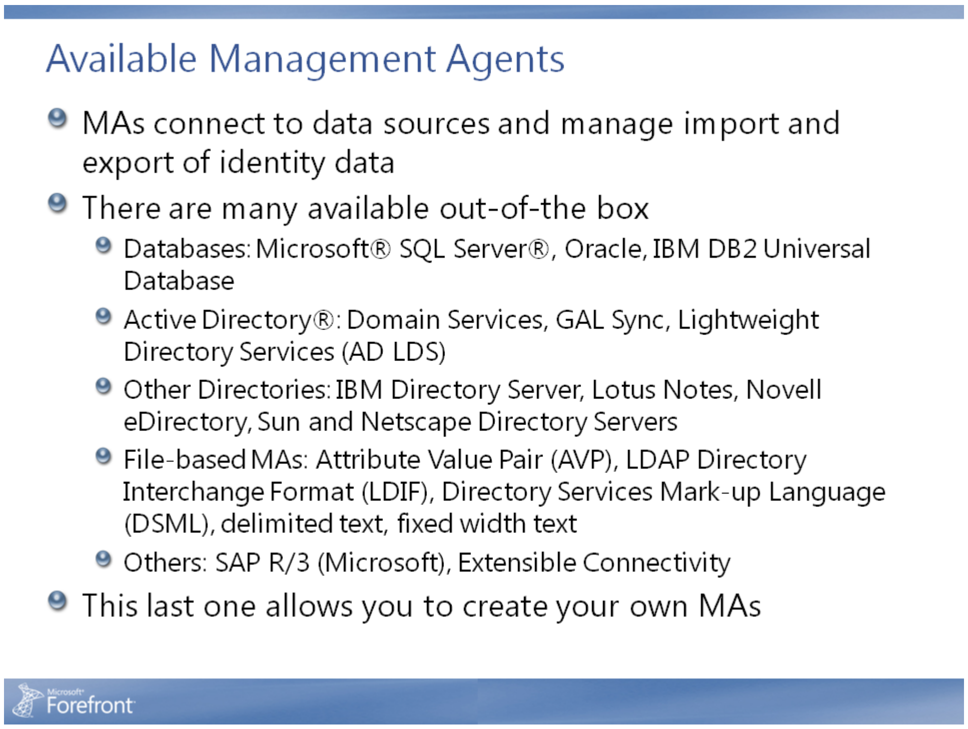
Connector Space

The connector space (CS) is a storage area and a staging area. It stores the holograms (states) that are used to decide whether information in a data source has changed or needs to be changed. It is also where changes are staged on their way into or out of FIM. Each data source has its own logical area in the connector space, which is managed by its corresponding MA. The CS is essentially a mirror of (or at least a version of) the related data source, with each object in the data source having a corresponding entry in the CS. The CS does not contain the data source object itself; it contains a subset of the object's attributes as defined by the MA.

MAs are primarily configured by setting their properties within the wizard-like interface in synchronization service manager. There are occasions when more complex operations are desired than those possible through the user interface; in these cases, an MA can be augmented by .DLL extensions produced using Microsoft® Visual Basic® .NET or C#, or any language making use of the Microsoft® .NET Common Language Runtime (CLR).

During development, MAs are executed via the user interface. In production systems, it is desirable to run MAs in sequence without user intervention, both on a scheduled basis, and occasionally in response to specific events (for example, the submission of a new user registration in an Intranet system). Automated execution of MAs is achieved using the Windows Management Instrumentation (WMI) functions of the FIM synchronization service in conjunction with a scheduling agent (described in detail later).

Available Management Agents



Available Management Agents

- MAs connect to data sources and manage import and export of identity data
- There are many available out-of-the box
  - Databases: Microsoft® SQL Server®, Oracle, IBM DB2 Universal Database
  - Active Directory®: Domain Services, GAL Sync, Lightweight Directory Services (AD LDS)
  - Other Directories: IBM Directory Server, Lotus Notes, Novell eDirectory, Sun and Netscape Directory Servers
  - File-based MAs: Attribute Value Pair (AVP), LDAP Directory Interchange Format (LDIF), Directory Services Mark-up Language (DSML), delimited text, fixed width text
  - Others: SAP R/3 (Microsoft), Extensible Connectivity
- This last one allows you to create your own MAs

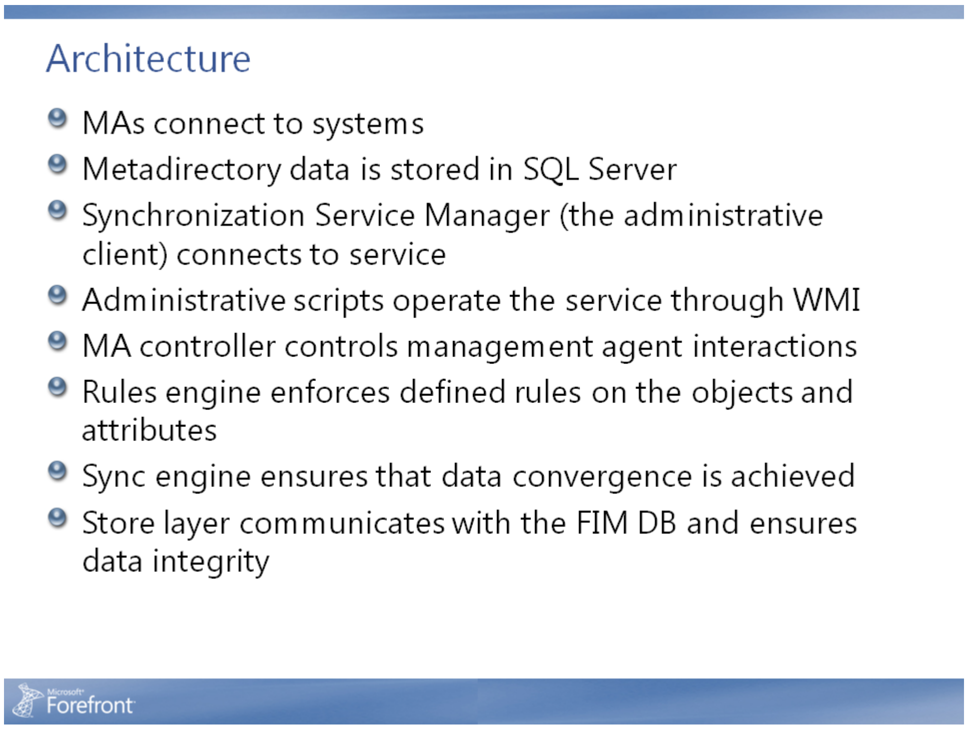
Microsoft Forefront

FIM ships with the following MAs:

- **Databases:** SQL Server, Oracle, IBM DB2 Universal Database
- **Active Directory**®: Domain Services, GAL Sync, AD LDS
- **Other Directories**: IBM Directory Server, Lotus Notes, Novell eDirectory, Sun and Netscape Directory Servers
- **File-based MAs**: Attribute Value Pair (AVP), LDAP Directory Interchange Format (LDIF), Directory Services Mark-up Language (DSML), delimited text, fixed width text
- **Others**: SAP R/3 (Microsoft), Extensible Connectivity

Additionally, it is possible to customize MAs using the extensible connectivity MA type. This calls a DLL which you must define, and which handles import and export. They can be quite sophisticated and there are many examples available from vendors or free on the Web, for example: for mainframe systems, MySQL, Microsoft® SharePoint® 2010, several different SAP approaches, etc.

Architecture



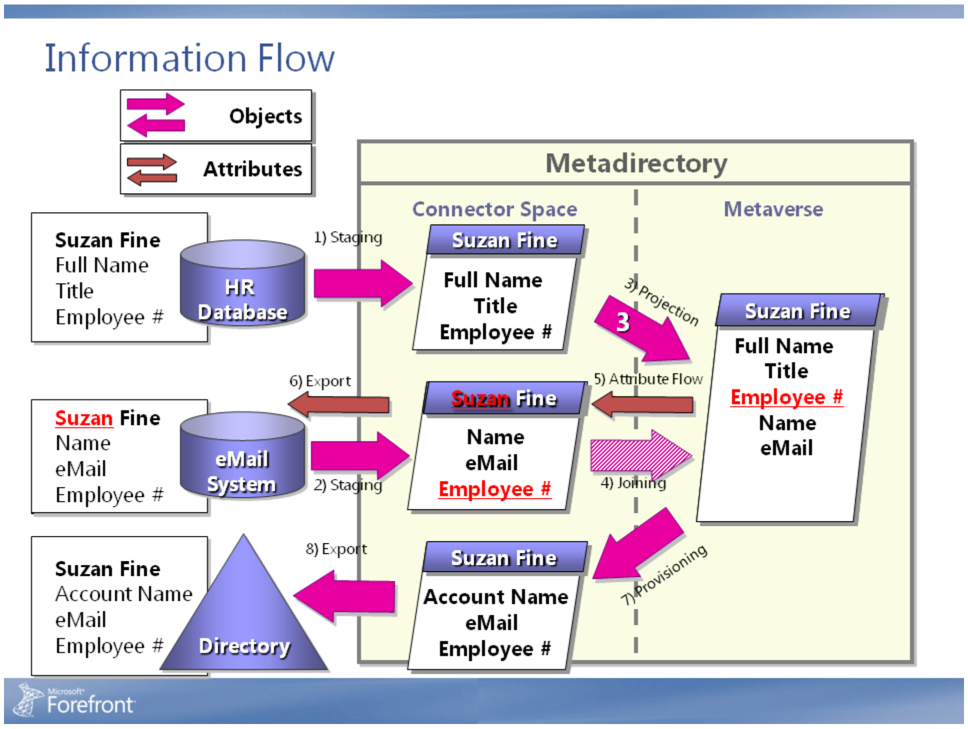
Architecture

- MAs connect to systems
- Metadirectory data is stored in SQL Server
- Synchronization Service Manager (the administrative client) connects to service
- Administrative scripts operate the service through WMI
- MA controller controls management agent interactions
- Rules engine enforces defined rules on the objects and attributes
- Sync engine ensures that data convergence is achieved
- Store layer communicates with the FIM DB and ensures data integrity

The FIM Synchronization Service architecture has the following characteristics:

- It runs as a Windows® service.
- MAs connect to connected data sources.
- FIM synchronization service data is stored in SQL Server.
- The Synchronization Service Manager (which is the administrative client) connects to the synchronization service, and is used to configure and control it.
- Administrative scripts can control the service through its WMI interface. Note that although it is a state-based system, it is able to respond to external events through WMI control.
- The MA controller controls MA interactions.
- The rules engine enforces defined rules on the objects and attributes (such as attribute flow and precedence).
- The sync engine:
  - Ensures that data convergence between the metaverse and the data sources is achieved.
  - Enforces transactions for each object. If there are errors in any part of an update, the entire transaction for that object can be rolled back to preserve consistency.
  - Handles errors.
  - Resolves references.
- The store layer communicates with the database and ensures data integrity.

Information Flow



Let’s a use a simple initial load scenario in which Suzan Fine is represented in two data sources, but not yet in a third—and not yet in the metadirectory. The process of how her information flows between data sources and the metadirectory could be configured as follows (note that this is just one possible scenario to demonstrate a few principles):

1. The MA for the first data source imports and creates her entry (including her attributes) in the CS. This is called staging. It is also called import, but this can mean the entire import all the way to the MV, so for clarity use staging.

2. The MA for the second data source does a similar job. Note that in the preceding illustration, there is an entry for Suzan Fine in each of the data sources. Therefore, there will be two entries in the CS, both representing Suzan Fine—you might also note an inconsistency in how her displayName is represented (and of course this is typical).

3. Depending on how the rules have been configured, one of the MAs projects a new entry in the MV (that is, creates a new MV object) that corresponds to the CS entry. The CS entry is now termed a Connector (since it is logically connected to a new MV object); until this point it has been a disconnector.

   Attributes may also flow, depending on the rules that have been defined, so attributes flowing in this example are displayName, Full Name, Title, and Employee #. (This flow has not been shown with an arrow, to avoid clutter.)

4. Again, depending on how the rules have been configured, the second MA attempts to match this entry with a corresponding entry in the MV—in this case using the Employee #. This action is called a *join* because all CS entries that represent the same real-world object are joined

together, or integrated, as one entry in the MV. Such CS entries are termed connectors (had it failed to find a suitable match, or had the rules simply not resulted in a join, it would persist as a disconnector).

Clearly, where possible, some unique attribute is used as the criterion by which entries are joined (such as an employee number). Equally clearly this is not always possible, so other approaches are possible, such as softer matching or manual joining.

At this point, Suzan Fine is represented by an entry in each of the Data Sources, by two entries (connectors) in the CS, and by an entry in the MV.

Note that during the join, no object flow took place as such, but attributes may have flowed (again, depending on how rules have been set)—in this case Name and eMail have flowed (again, this flow is not shown with an arrow.)

5. As well as inbound attribute flow associated with the join, there may be outbound attribute flow according to the attribute flow rules you have defined—in this case the display name flows correcting the inconsistency.

   In general, where the same attribute could have been provided by more than one source (like displayName in our example), one is chosen as authoritative, or several are allowed to be authoritative but with an order of precedence.

6. When the MA is next run in export mode, the attribute will be flowed out to the data source.
7. Whenever an MV object is created (or changes in some way), any defined provisioning logic will be applied. In this case a new CS object is created in that part of the CS associated with the third MA. If there are outbound attribute flow rules defined, attributes will flow—in this example, e-mail and employee number, as well as a suitably generated account name.
8. When the MA is next run in export mode, the newly provisioned account (object) will be created in the data source, along with its attributes.

Further Attribute Flow

Now that the connections have been made, further attribute flow may take place (for example there may be attributes in the third MA that can be imported, and then usefully exported to the first MA.

Anchor IDs and MV GUIDs

Looking at the diagram, and bearing in mind that each CS or MV object is just an entry in a SQL Server table, FIM must store unique identifiers in order to maintain connections between objects.

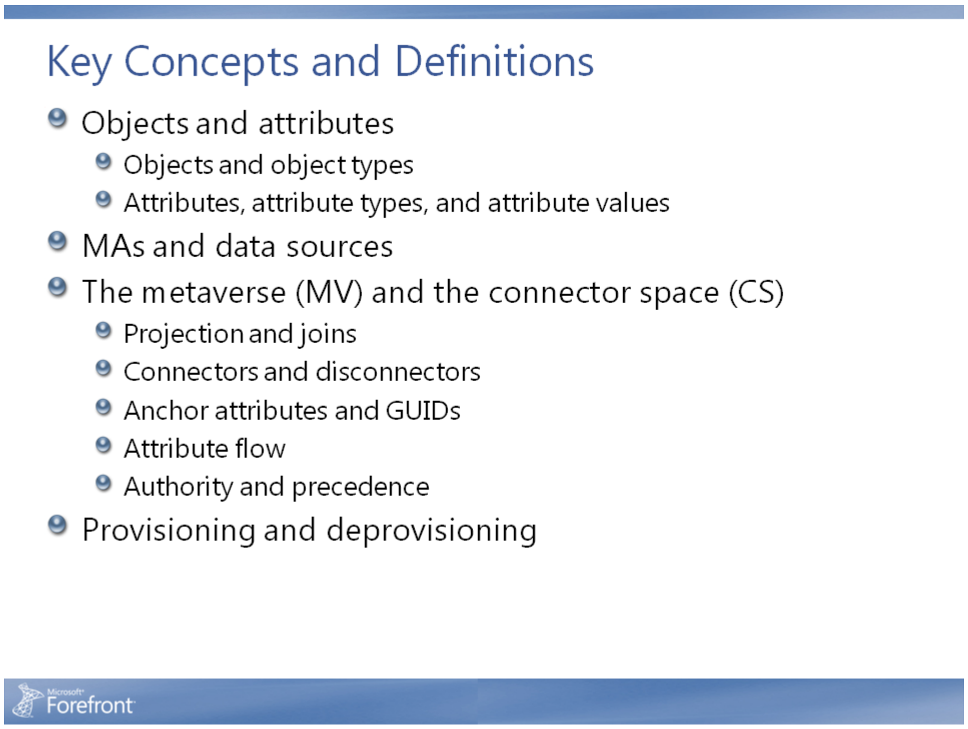
For each MA, a unique data source attribute is defined (such as employee number, or perhaps a GUID, or unique database ID)—this is called an **anchor**.

Each MV and each CS object is given a GUID (a globally unique ID) generated by FIM. These are used to define connections.

The anchor in the case of a directory such as Active Directory (AD) is the distinguished name (DN), and it is a sign of the strong relationship between the FIM synchronization service and AD that you will see anchor and DN being conflated into a single term: anchor (DN)—even where no DN exists as such.

It may seem a complicated point to make at this juncture, but in the case of a directory such as AD, although the DN appears to be the anchor, and behaves as though it were an anchor, there is usually a more fundamental anchor under the covers, and in the case of AD it is the AD GUID. This allows DN renames to take place without anchor issues.

Key Concepts and Definitions



Key Concepts and Definitions

- Objects and attributes
  - Objects and object types
  - Attributes, attribute types, and attribute values
- MAs and data sources
- The metaverse (MV) and the connector space (CS)
  - Projection and joins
  - Connectors and disconnectors
  - Anchor attributes and GUIDs
  - Attribute flow
  - Authority and precedence
- Provisioning and deprovisioning

Microsoft Forefront

This topic briefly summarizes important concepts and their abbreviations, which contribute to an understanding of the architecture and function.

Objects and Attributes

Each entry in the metadirectory is an object of a specific type, which is comprised of a number of attributes. For example, the entry concerning Suzan Fine is a specific person object, which possesses attributes (for example, displayName, title, department, telephoneNumber) defined for the person object type. Examples of other object types are computer, group, and organizational unit.

Note: While object types are similar in function to LDAP object classes, they do not exhibit inheritance, and an object may be of only one type.

Attributes come as different attribute types, too, such as string, binary, or Boolean, and for each object an attribute has a particular attribute value (or, in the case of a multi-valued attribute, a set of values).

Data Sources and MAs

The external sources of data for FIM are referred to collectively as connected data sources or just data sources (or even just sources).

The data flow between FIM Synchronization Service and a specific data source is defined in a management agent (MA), and data flows only when an MA is run.

The Connector Space

There is actually just one connector space (CS), but each MA has a logical division of the CS, an area in which it stores its holograms (from which changes in data-state are calculated—see the discussion on state-based systems, above) and in which pending imports and exports are staged on their way from and to their respective data sources. An MA may handle all objects in a data source or may be configured to deal with only a subset of objects and their attributes. Every object and attribute from the data source handled by an MA is represented in the CS.

Metaverse

The collection of objects and attributes that hold the combined data of objects of interest in the data sources are stored and managed in the FIM Metaverse (MV). The MA configuration defines which objects and attributes are synchronized with the MV.

Joins and Projection

When a CS object is connected to a corresponding entry in the MV, that is called Joining. If an entry in the CS should be represented in the MV, but no corresponding entry in the MV can be found by the Join process, a new MV object may be created for this entry (depending on how the MA is configured). The creation of a new MV object is known as projection, and following projection, the CS entry will stay connected to the new MV object.

Connectors and Disconnectors (and placeholders)

Each object in the CS is either connected to an object in the MV, or it is not. If it is, it is referred to as a connector. If it is not, it is a referred to as a disconnector.

It is worth mentioning that there is a special kind of disconnected object—a placeholder. Suppose that you import an object which has a reference to a second object that is not present (for example an AD user pointing to a manager that is in another organizational unit (OU) not being managed by FIM), that second object will be invented as a placeholder in order to maintain the reference correctly. If the second object is subsequently imported, the placeholder is replaced by the real object.

GUIDs and Anchor IDs

Each MV object (and for that matter each CS object) is identified by a unique value—its globally unique ID (GUID). The connection between a CS object and its corresponding MV object is achieved using GUIDs —by storing the GUIDs of related objects in a connections table.

Similarly, there must be a link between each object in the data source and the CS, so that the MA can manage the entries in the CS with the corresponding entries in the data source. This is achieved through a unique attribute (or a unique combination of attributes) which originates in the data source, and is referred to as the anchor—or, as has already been mentioned, anchor (DN).

Attribute Flow

Once connections are made (Data Source to CS to MV, and back again), attribute flow can take place according to defined attribute flow rules.

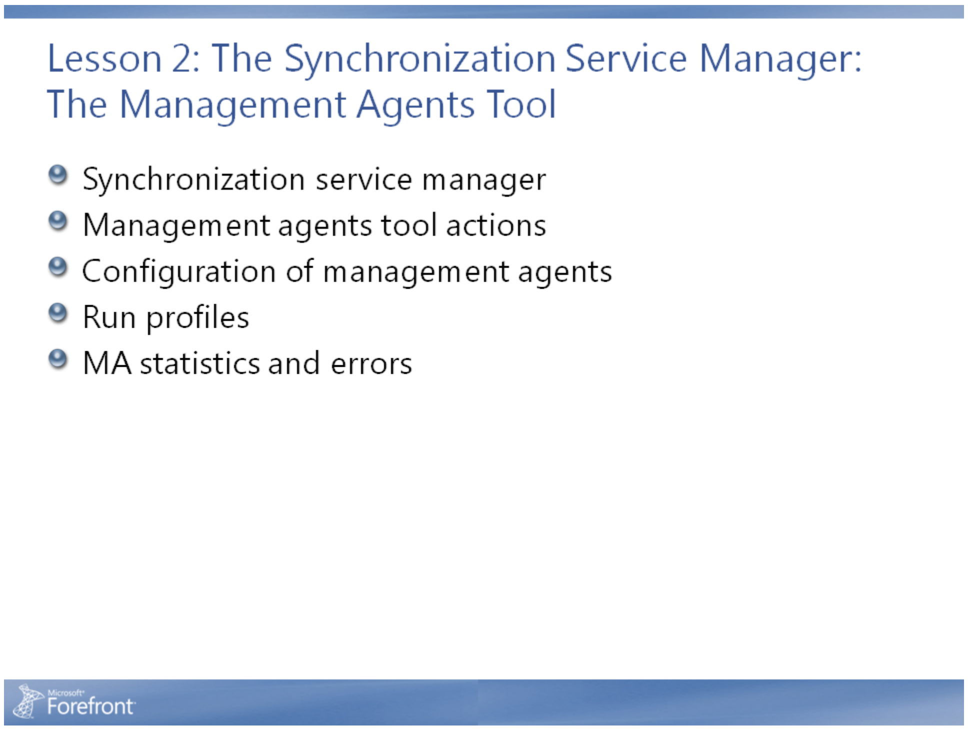
Authority and Precedence

Each object in the MV generally represents entries in more than one data source. It is possible that each attribute value could be provided by more than one of these data sources (in other words multiple attribute flow rules may exist for each attribute). In general, then, there may be a conflict of attribute values. Therefore, it is important to understand which rules have authority for each object type and attribute (that is, could create/delete an object, or could provide an attribute value), and where there is more than one, which rule takes precedence. The rules in FIM will be governed by business rules, which arise from an understanding that comes primarily from discussions with the business owners of the various systems, and is one of the key requirements for success of a metadirectory implementation project.

Provisioning and Deprovisioning

It is possible to define object flow rules in the FIM synchronization service which state that the presence of an entry in one data source requires the presence of a corresponding entry in another data source. For example, the existence of an entry in a human resources system (representing an employee) might require the existence of an AD account for this employee. In order to enforce this rule, FIM may be configured to create and remove entries in (for example) AD—the creation of entries is termed provisioning and the removal of entries is termed deprovisioning.

Lesson 2: The Synchronization Service Manager: The Management Agents Tool



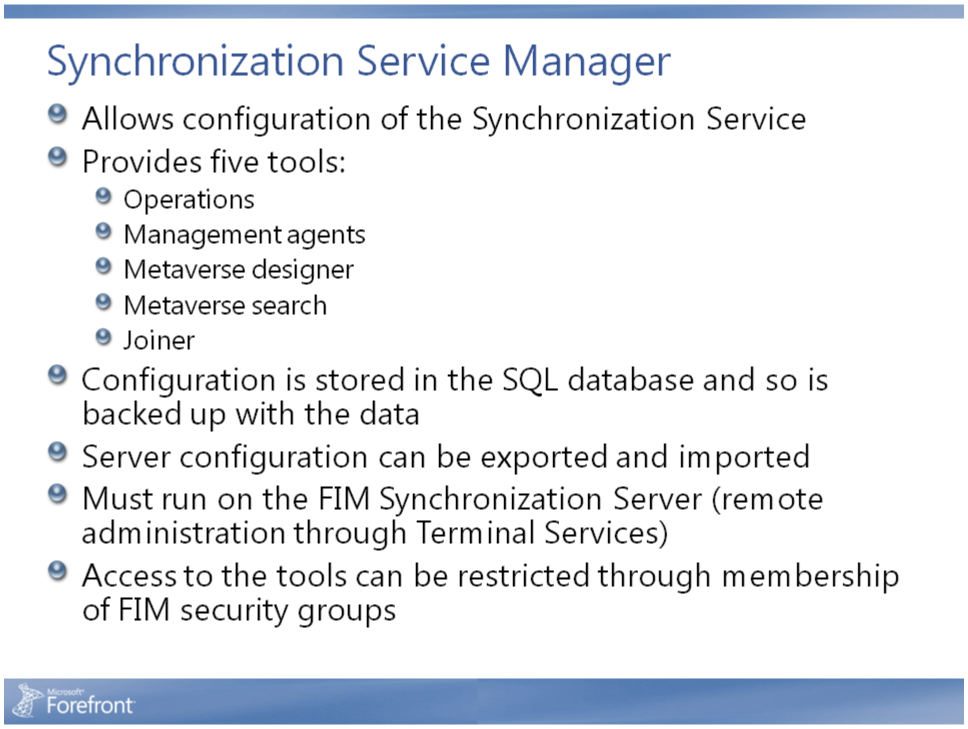
Lesson 2: The Synchronization Service Manager: The Management Agents Tool

- Synchronization service manager
- Management agents tool actions
- Configuration of management agents
- Run profiles
- MA statistics and errors

This lesson introduces Synchronization Service Manager through the (arguably) most important tool it provides, the Management Agents Tool.

It covers the configuration of MAs and run profiles, and considers the MA statistics and errors it reports.

Synchronization Service Manager



Synchronization Service Manager

- Allows configuration of the Synchronization Service
- Provides five tools:
  - Operations
  - Management agents
  - Metaverse designer
  - Metaverse search
  - Joiner
- Configuration is stored in the SQL database and so is backed up with the data
- Server configuration can be exported and imported
- Must run on the FIM Synchronization Server (remote administration through Terminal Services)
- Access to the tools can be restricted through membership of FIM security groups

The Synchronization Service Manager provides five configuration tools to aid FIM administration and operation.

Server Configuration

All the synchronization service data is stored in a SQL Server database—the identity data and configuration data (MV, MAs, and the statistics concerning past management agent runs). This architecture, with all this important information in one place, allows for straightforward backup and restore of the system (more later).

Synchronization Service Manager must run on the same server as the synchronization service itself. If remote administration is required, it can be performed by using Remote Desktop from the remote location or, to a much more limited extent, via the WMI interface.

It is possible to export (and import) the entire server configuration to an .XML file, using the options on the File menu. This exports the metaverse design to a file called MV.XML, and each MA configuration (including run profiles) to files called **MA-{MA-GUID}.XML**, where MA-GUID is the object GUID for the MA. This might be done, for example, in order to transfer a configuration from a test system to a production system (without including the identity data), or for analysis by an external documentation tool.

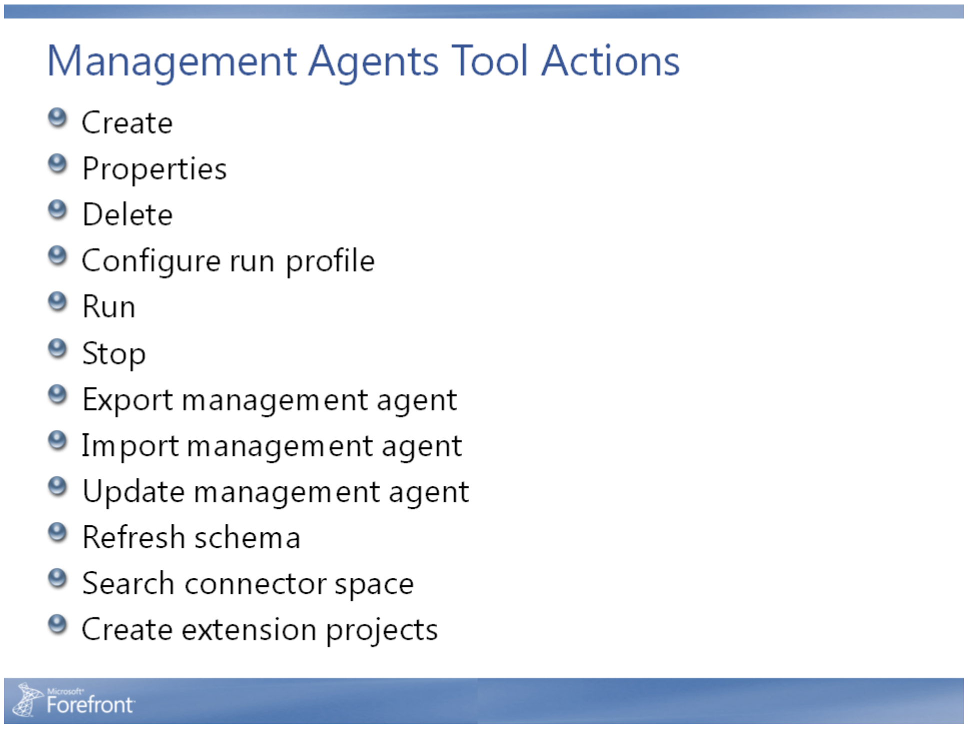
Management Tools and Security Groups

Note that during FIM installation, several security groups are created which can be used to limit a user's access to these tools.

A relatively low-level administrator, responsible for monitoring the running of the server, but not in a position to modify the configuration of the server, would be assigned membership of the FIMSyncOperators group, granting them access the operations tool only.

- Members of the FIMSyncAdmins group have full access to all tools.
- Members of the FIMSyncJoiners group only have access to the Joiner tool.

Management Agents Tool Actions


Management Agents Tool Actions

- Create
- Properties
- Delete
- Configure run profile
- Run
- Stop
- Export management agent
- Import management agent
- Update management agent
- Refresh schema
- Search connector space
- Create extension projects

MAs are managed using the management agents tool within Synchronization Service Manager. So a good deal of time is expended using this tool initially, although once an implementation is complete, other tools become more important.

Create

This invokes a wizard-like interface for creating a new MA, starting with the type of MA and working through the entire configuration.

Properties

This is for adjusting the properties of an existing MA (and it is the default action when you double-click an MA). With a few exceptions, it gives access to the same wizard pages as the Create action.

Delete

This has two functions: one is to delete the CS objects associated with that MA; the other to delete the CS objects and the MA itself. Obviously deleting an entire MA is unusual, but during development and testing you quite often find yourself deleting the CS and then re-importing or re-provisioning to try out your rules.

Configure Run Profiles

MAs run in different modes (such as export or import). For each type of run, you must configure a run profile. You can configure any number of run profiles for a particular MA, each performing a specific set of steps. At least one run profile with at least one step is required to run an MA.

Run; Stop

Run launches the selected MA according to a run profile you specify. Stop attempts to stop the currently running run profile (this may take a while if it is busy).

Export, Import, Update

These actions are for exporting an MA configuration to an XML file, and for importing that information into the same, or a different, FIM instance.

An MA, along with its defined run profiles, can be exported to an .XML file, and later imported. You can save an MA configuration to an XML file using Export.

If you use import to import an exported file, you create a new MA (with a new MA GUID); the import process goes through a validation process (for example, checking the data source schema), and a verification process in which the user is required to check each setting as though creating a new MA (in fact it is a bit like creating a new one, but with defaults pre-entered).

You can also update an existing MA from a previously exported XML file. This doesn't actually create a new MA, and can be used (for example) to roll out changes from a test system to a production system.

Export, import, and update are particularly useful for:

- Deploying changes to MAs.
- Developing new MAs based on existing ones.
- Sending an MA to a remote administrator for analysis or debugging purposes.
- Version control of MAs (suitably named, exported MA files can be saved, and possibly included in a version control system like Visual SourceSafe).
- Documentation: the MA Configuration Display tool in the Resource Kit uses an XML file as input.

For security reasons, no passwords for connection to the data source are stored in the XML file, although username and domain details are. For call-based MAs you must be in a position to provide the appropriate connection password(s) during the import step.

Refresh Schema

This feature causes FIM to re-read the data source schema (that is, the available objects and related attributes of the Data Source). This is used when the structure of a data source is changed (after an MA has already been created).

> **Note:** Refresh Schema can result in significant changes in MA configuration (for example, attributes hitherto relied upon may now be modified or even missing), so FIM takes the user through a validation and verification process.

Search Connector Space

This feature is for locating and examining objects in the CS for the selected MA—there is more on this later in the course.

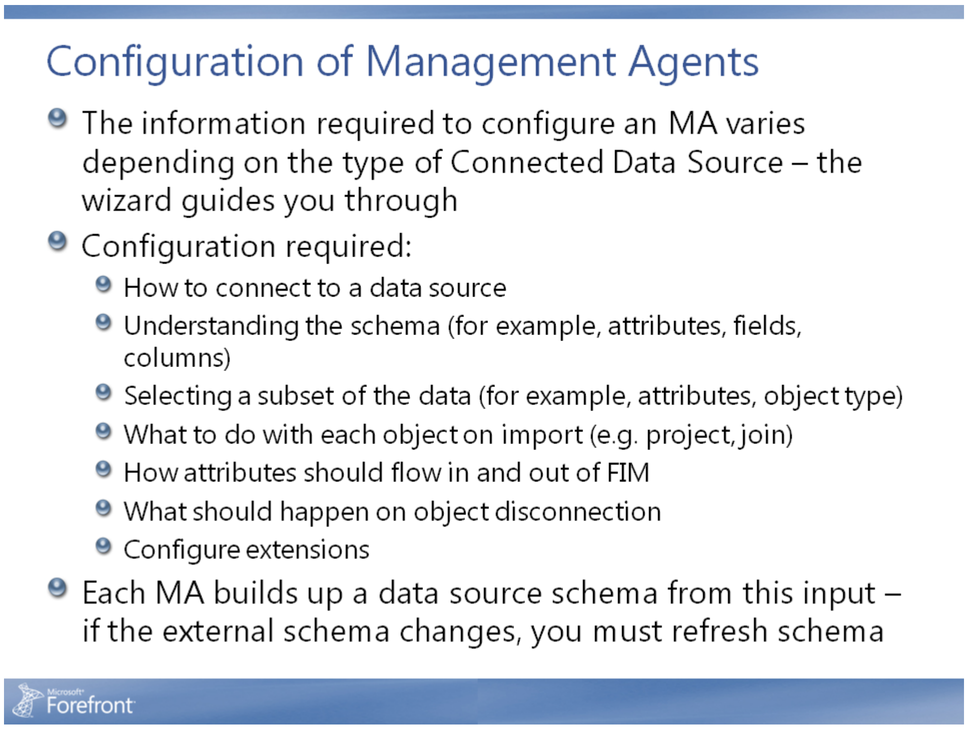
Create Extension Projects

This action creates a template code module, depending on the type of extension project selected, which is then modified and compiled into a DLL. Such rule extensions allow immense flexibility in how MAs are configured but are beyond the scope of this course.

Statistics and Error Reporting

If you run an MA from this interface, you are presented with statistics (how many additions, deletions, etc.) and any error messages. Generally, if you select an MA, you will see the statistics for the most recent run for the MA concerned (and the type of run, etc.) Most of these statistics take the form of hyperlinks, which you can click to drill down into further detail.

Configuration of Management Agents



Configuration of Management Agents

- The information required to configure an MA varies depending on the type of Connected Data Source – the wizard guides you through
- Configuration required:
  - How to connect to a data source
  - Understanding the schema (for example, attributes, fields, columns)
  - Selecting a subset of the data (for example, attributes, object type)
  - What to do with each object on import (e.g. project, join)
  - How attributes should flow in and out of FIM
  - What should happen on object disconnection
  - Configure extensions
- Each MA builds up a data source schema from this input – if the external schema changes, you must refresh schema

The same basic principles apply whether you are modifying an existing MA, creating a new one, or importing one. In this topic you'll cover enough to be prepared for the lab that follows. The next module goes into more detail.

MA Configuration Steps

The wizard-like interface that guides you through the configuration of an MA varies depending on the type of MA being configured. However, it is possible to generalize the steps:

Specify the Data Source

For file-based MAs this will involve providing a template input file.

For call-based MAs this involves providing some kind of connection data and credentials: a combination of user name, password, domain, server, port, database name, table, view, etc.

Note that databases do not naturally present delta information (that is, *changes since we last asked*), and also that multi-valued attributes cannot be stored easily in a single (two-dimensional) table, so FIM provides for a delta view (a view that, probably with some considerable effort, SQL Server could be persuaded to populate with delta information), and a multivalue table (which could be populated with transactional data representing multiple values for attributes.) Neither of these is covered in this course, but fortunately most of the sources of interest support these features natively.

Understand the Data

FIM has to work out how to interpret the object types and attributes (or fields or columns) in the data source—the schema:

- For file-based MAs it uses the template input file.
- For call-based MAs it either:
    - Reads the schema directly—this is true for AD or SQL Server.
    - Understands it to be a fixed schema—this is the case for Lotus Notes.

Some MAs will completely understand a schema, as is the case for AD; others require some manual tweaking, as is the case for SQL Server and file-based MAs.

First you must make sure that FIM recognizes an anchor—a unique and reliable attribute (or combination of attributes), that can be used to clearly identify and track a data source attribute.

You may need to edit attributes. In an extreme case an attribute may have been misinterpreted (for example, a string attribute may have been interpreted as a number), which you would clearly need to fix. A more common adjustment is to identify attributes that are references. For example, a manager column in an HR database might contain references (pointers) to EmployeeIDs, and you would obviously like these references to carry through to the metaverse and eventually to other sources (such as AD). You need to configure the manager attribute as a reference attribute to prevent FIM from handling it as a simple attribute (such as string).

In the CS, reference attributes can only point to the anchor, so in this example, EmployeeID would have to be the anchor for this manager reference to work.

Note: Somewhat confusingly, FIM uses the terms **DN** and **anchor** almost interchangeably. This is because in LDAP-style MAs such as AD, the DN appears to be the anchor (actually the true AD anchor is the AD GUID). So you may see the term **reference (DN)** even though there is no DN involved.

You may need to configure object types—perhaps a fixed type for the MA or perhaps you can indicate an attribute that holds the object type.

Delta and multi-value configurations can also be configured but this is not covered here.

Selecting a Subset of Data

Where relevant, a subset of data can be selected. You can define a filter to exclude objects from synchronization. Note that if a change to an object is imported, such that it matches the filter, and that object was previously a normal connector, it will be disconnected and then take no further part in synchronization.

However, if there are objects that you simply don’t want to be considered for synchronization, and which it would be a waste of time and effort to import in the first place, you can do this:

- For some call-based MAs, notably directories, you may be able to specify a combination of partition, OU, object types, and attributes in which you are interested (excluding the rest).
- For all others you need to present just the objects of interest using (for example):
    - For a database: a suitably configured view that presents the objects and attributes of interest.
    - For a file-based MA: a file that only contains objects and attributes of interest.

What to Do with Imported Objects

Those not filtered out are considered for joining to existing MV objects, or projecting as MV objects according to rules you define. The rules can be simple, for example:

- Join using EmployeeID.
- Project a person object.

You can have many join rules that are tried in order, and the process stops on success. Since projection comes last, it will only happen if all join attempts fail. Rules can be much more complicated, involving coded rules extensions.

It may be obvious that creating duplicate MV objects is not a good thing, so you should always attempt a join before projecting. Note that if a connector becomes accidentally disconnected (this could be in a disaster recovery situation or simply that an administrator has disconnected something they shouldn't have), the only way to connect up again is through a join, and while you can effect manual joins, this can be time-consuming and may introduce further errors. Therefore, follow these maxims:

1. Always try to identify, or if necessary make available, a reliable, unique attribute for joining purposes.
2. Always configure a join rule, even if it doesn't seem very important at the time.

How Attributes Should Flow

For each attribute to be imported (or exported), for each object type, you must specify how attribute values in the source should be flowed to attribute values in the MV (and vice-versa).

In simple cases these will be direct: CS object attribute to/from MV object attribute (where both are of the same type). Since schemas vary, there will generally be mapping of attributes from the particular source schema to a common MV schema, for example: LastName in an HR system to sn in the MV.

You can also configure flows that:

- Flow constant values into attributes.
- Flow a particular element of a DN into an attribute (for example, you could flow the second element of CN=fred,OU=it,OU=…) and it would flow "it."
- Take multiple attributes and transform them in some way and then flow the result to another attribute, but this involves code and will not be covered in this course.

What Should Happen when Entries are Removed

If an MV object is deleted (perhaps because its corresponding real-world object is no longer part of the system), what should this MA do with any orphans, that is CS objects that have just become disconnected from their corresponding MV objects? This question also arises if the disconnection results from a manual or programmatic action (this is how an account would be de-provisioned as a result of a role change, or employee suspension, for example).

The choices are:

- **Make it a Disconnector** – A disconnector is treated just like a new object that has just been imported—referred to as a **pending add**. During every synchronization the rules will be applied, and as a result it could be joined to an MV object, projected, or perhaps sit forever as a disconnector (but being reconsidered with every synchronization).
- **Make it an Explicit Disconnector** – An explicit disconnector is kept aside from synchronization; nothing more will happen to it unless it is manually changed back to being a normal disconnector (you can do this in the Joiner tool); this would be an appropriate choice if you were sure that the identity concerned should never again be involved, but that a record should be kept in the Data Source concerned, and should not be deleted.
- **Stage a delete on the object for the next export run** – This turns it into pending delete, to be pushed out on the next export run, resulting in the corresponding data source object really being deleted. This would be appropriate when the data source concerned is not authoritative, and you are choosing to deprovision it (in all probability having provisioned it sometime before).
- **Determine with a Rules Extension** – Write code to do something else (such as disable, or move an account to a special OU) and then make a choice from the above three programmatically (beyond the scope of this course).

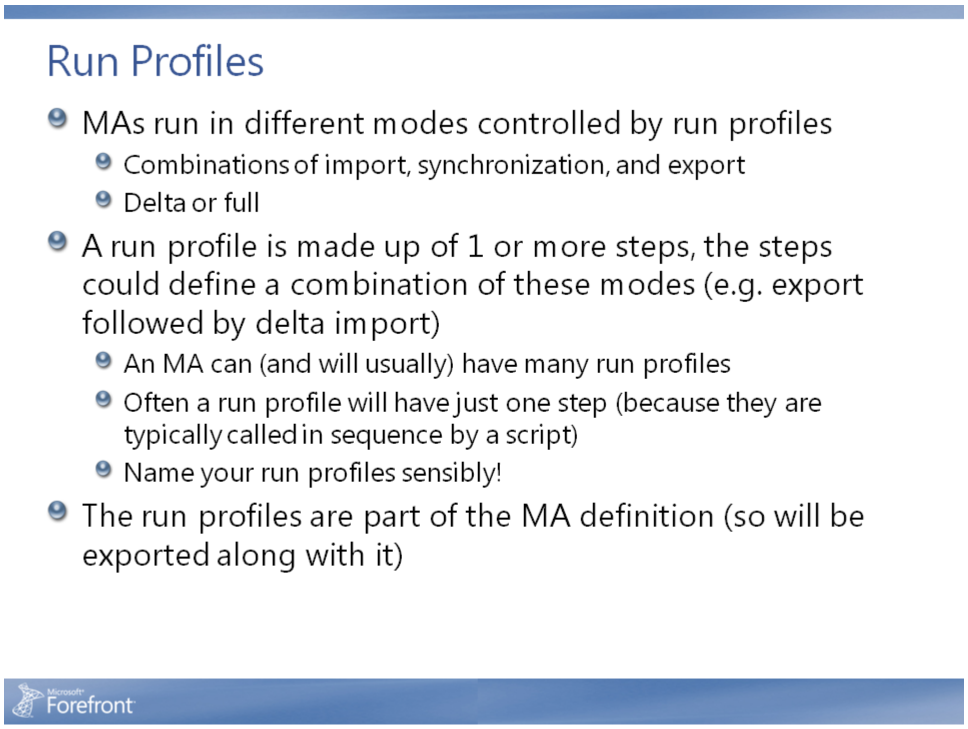
Configure Extensions

This page allows you to configure password management (see Module 6) and configure (coded) rules extensions, which are beyond the scope of this course.

Refreshing the Schema

MAs do not automatically query the data source to re-establish a picture of the schema. If the underlying data source changes in a way that is significant to FIM, you must refresh the schema for all MAs that connect to the data source.

Run Profiles



Run Profiles

- MAs run in different modes controlled by run profiles
  - Combinations of import, synchronization, and export
  - Delta or full
- A run profile is made up of 1 or more steps, the steps could define a combination of these modes (e.g. export followed by delta import)
  - An MA can (and will usually) have many run profiles
  - Often a run profile will have just one step (because they are typically called in sequence by a script)
  - Name your run profiles sensibly!
- The run profiles are part of the MA definition (so will be exported along with it)

Microsoft Forefront

A Run Profile is made up of steps—a set of steps that specify how to execute, or run, an MA. Each step is one mode of run, like **full import** or **export** or **delta synchronization**. An MA can have many run profiles, which are stored (and exported) with the MA.

Run Profile Steps

A run profile is composed of at least one run profile step. For example, if a full import (stage only) profile step is specified, the MA imports all data into the CS but does not process it any further. As another example, an export step could be followed by an Import step (which is usually required to confirm that export was successful).

Run Profile Step Types

These are the available run profile steps (though note that those involving delta import are only available if the MA supports them locally or they have been otherwise configured):

- **Delta Import (Stage Only)** – This is the most efficient kind of import because it only imports changes; as a result the CS will hold pending imports with the modification type Add, update, or delete.
- **Full Import (Stage Only)** – Imports all data source identity data into the CS, compares it with what is already there, and decides what has changed, creating pending imports with the modification type add, update, or delete. Those in the final category are established by the simple fact that they have gone missing (this is termed *obsoletion*). This is quite unlike the case for a delta import where a deletion effectively stems from an instruction to delete.

- **Delta Synchronization** – Applies synchronization rules to any CS objects that need it, which means newly imported changes—pending import add, update, or delete—plus any disconnectors (which are always treated as pending import adds).
- **Delta Import and Delta Synchronization** – With this run profile, each change that is imported is immediately synchronized, too, but nothing else happens (a normal delta sync synchronizes everything that is pending).
- **Full Import and Delta Synchronization** – Simply a quick way to do both of these in one go: imports all the latest data and synchronizes everything that needs it.

Note: While a full import and delta synchronization as a combined step does the same things as a full import step followed by a delta synchronization step, the same is not true for a delta import and delta synchronization. A delta import and delta synchronization as a single step only synchronizes what it imports, whereas a delta synchronization step on its own synchronizes everything that needs synchronizing.

- **Full Synchronization** – Re-synchronizes all CS objects with the MV (and, if necessary, propagates to objects in the CS governed by other MAs); this run profile is used to apply new rules to old data, and is not normally used in regular daily production runs. Full syncs apply to *all* objects and can be very time-consuming.

Note: Sometimes when you run a delta synchronization, you will see a run step warning. If FIM detects any change in the configuration, however trivial, it will warn you that you may need to run a full sync (to make sure that new rules are run against the existing data). You have to decide whether or not to proceed (or you may be given the option to switch to a full synchronization). If you are confident that a full synchronization is not necessary (and this will often be true) you can select **Do not display the warning again** and proceed with the delta synchronization.

- **Full Import and Full Synchronization** – Another combination of two others.
- **Export** – Suring synchronization, any changes in the MV trigger necessary export activity, building up pending exports, which also have the modification types of add, update, or delete; an export run sends these out, but just the changes, because exports are always delta.

Which Steps to Use

An MA can (and will usually) have many run profiles—some used regularly during normal production running, and others used more occasionally. For example, a full sync is used to apply new rules to old data, and a full import might be used periodically to make sure that nothing was missed during delta imports (which could be an issue where file transfers are part of the process).

Often a run profile will have just one step—in a script it is just as easy to execute lots of single step run profiles as it is to execute a few multiple ones.

Part of the MA Definition

The run profiles are part of the MA definition, so if you export and import MAs, the run profiles go with them.

Naming

It is suggested that you name your run profiles meaningfully so that you can address them more easily under WMI and follow them in the operations tool.

Other Options

Log Files

For any step type involving an import or export you can create a log file. This drops an XML file of the raw data either received from the data source or sent to it in the MaData folder (under C:\Program Files\Microsoft Forefront Identity Manager\2010\Synchronization Service). This can be useful for auditing, but is especially useful for debugging.

Two further options are available only for the delta import, full import, and export step types:

1. **Create a log file and stop the run** – For import this drops the log file bit does not update the CS, while for export this drops the log file but does not update the data source.
2. **Resume** – Which in either case completes the import or export run.

One use of these options is that you can inject data errors into the file to see how the source or the MA (as configured by your rules) copes with the errors.

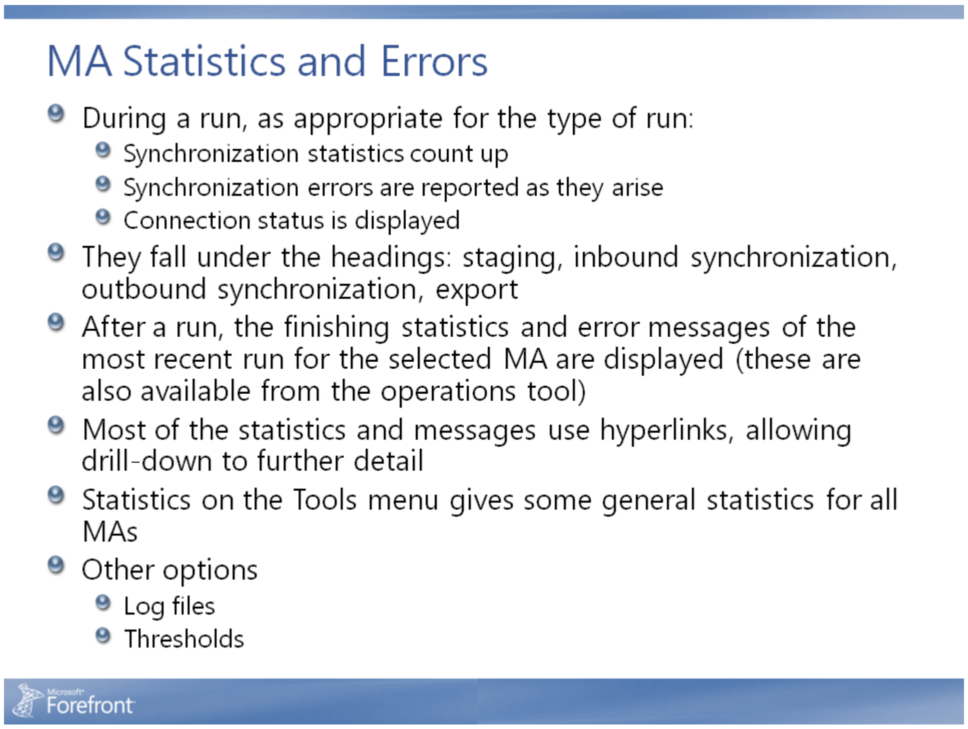
Another use is to keep files so that you can regression test future configurations against known conditions (this can require a lot of organization and careful naming).

Thresholds

You can specify a number of objects to process for a step. Note that for delta import or export (which is always delta), if you repeatedly run a step with this threshold, you gradually work through your objects (which may not be your intention!) For a full import or full synchronization, you just keep on processing the same objects over and over (assuming all other things are the same). For a delta synchronization you may work gradually through your objects, or you may have a lot of disconnectors that never go anywhere and keep being processed.

You can also specify the number of deletions to process. This tries to protect you against an unreasonable number of deletions coming through. Keep in mind that depending on the step type, and depending on the way your scripts are organized, you may find yourself gradually processing all deletes anyway. A better approach might be to use a script to count the number of deletes and make a decision. This is covered in Module 8.

MA Statistics and Errors



MA Statistics and Errors

- During a run, as appropriate for the type of run:
  - Synchronization statistics count up
  - Synchronization errors are reported as they arise
  - Connection status is displayed
- They fall under the headings: staging, inbound synchronization, outbound synchronization, export
- After a run, the finishing statistics and error messages of the most recent run for the selected MA are displayed (these are also available from the operations tool)
- Most of the statistics and messages use hyperlinks, allowing drill-down to further detail
- Statistics on the Tools menu gives some general statistics for all MAs
- Other options
  - Log files
  - Thresholds

FIM displays a lot of useful statistics.

Statistics and Messages

The Management Agents tool gradually displays available statistics as a run takes place. The information available includes:

- Step type – if there are multiple steps in a run profile, each step will be shown.
- Start time and end time for the run.
- Where multiple partitions exist (for example in multi-domain Active Directories), which partition was synchronized.
- Status – success or failure message.

In addition, there are statistics available for each phase of the synchronization process:

- **Staging** (this is the import from the data source to the CS) – How many of the CS objects were unchanged, added, updated, and deleted. Note that renames are only meaningful where distinguished names (DNs) are used.
- **Inbound synchronization** – How many CS objects were projected, joined, had attribute flow updates, were processed but did not have attribute flow updates, how many disconnectors, how many resulting metaverse deletes and so on; two that are less obvious are the filtered connectors—CS objects that were connectors and have just been disconnected by this synchronization (because they matched the connector filter), and filtered disconnectors—CS objects that were already disconnectors, and have just matched the connector filter.

- **Outbound Synchronization** – This reports the number of export flows and other outbound action (like provisioning and deprovisioning actions) triggered when MV objects are involved in the synchronization process and the rules are applied to it; an obvious case is when an inbound flow changes an attribute value that is part of an outbound flow.
- **Export** – This is the exports sent out in the categories of add, update, delete, and—where DNs are in use renames—delete-adds (a handy combination of a delete and an add that have been done in quick succession).

Errors

If errors occur during a step, an entry for each object in error will appear in the error pane (on the right). There is an error limit set at 5,000 objects (although this is configurable in the registry). If a step encounters 5,000 errors the process will stop.

Hyperlinks

Most error messages and statistics are in the form of hyperlinks to further details, including properties dialog boxes.

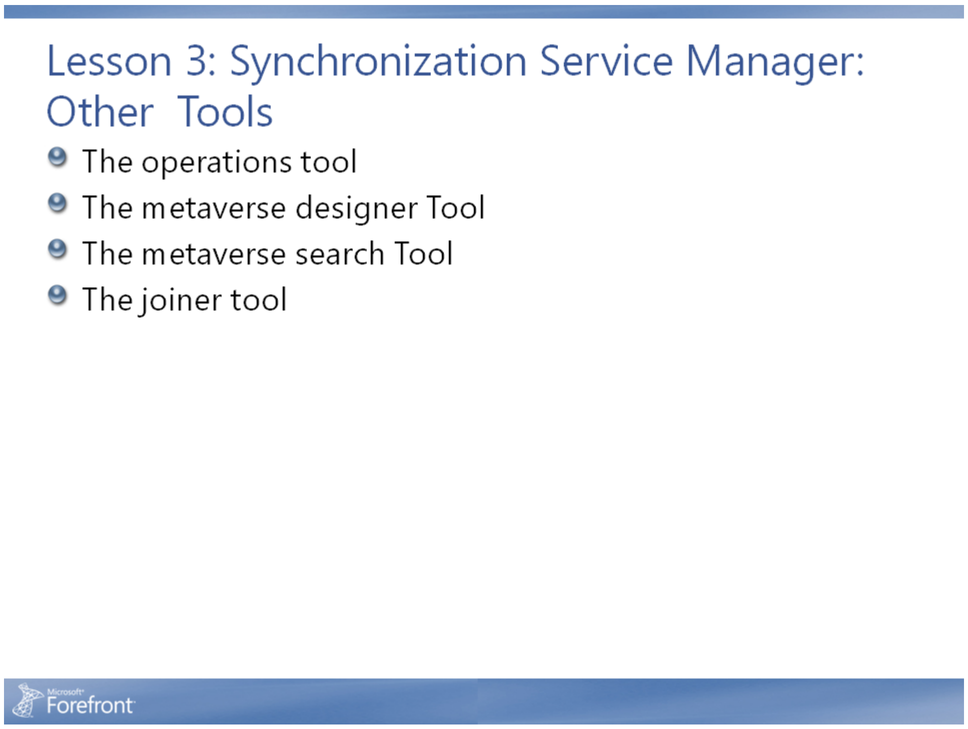
Run History

The MA statistics stay in place until another run by the same MA, but the operations tool can be used to view the statistics for previous runs.

Other Statistics

On the Tools menu there is a statistics option that gives overall statistics for all MAs, for example: Total objects, connectors, disconnectors, import/export, adds/updates/deletes, and so on.

Lesson 3: Synchronization Service Manager: Other Tools

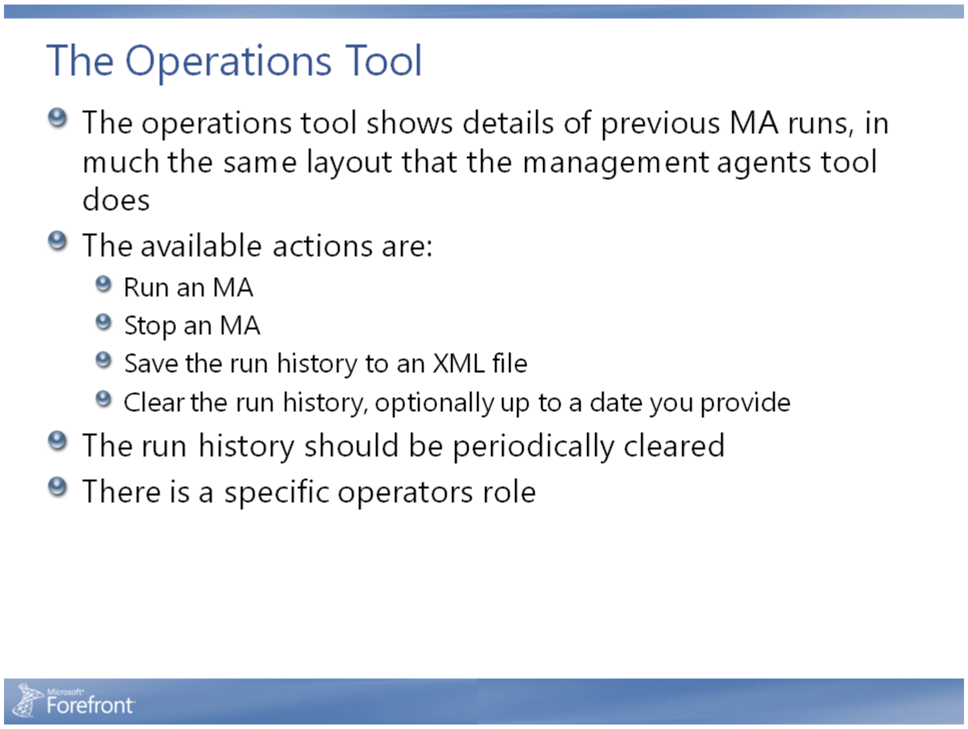


Lesson 3: Synchronization Service Manager: Other Tools

- The operations tool
- The metaverse designer Tool
- The metaverse search Tool
- The joiner tool

The management agent tool has been discussed in some detail because it is the arguably the most important. The four other tools will now be covered.

The Operations Tool



The Operations Tool

- The operations tool shows details of previous MA runs, in much the same layout that the management agents tool does
- The available actions are:
  - Run an MA
  - Stop an MA
  - Save the run history to an XML file
  - Clear the run history, optionally up to a date you provide
- The run history should be periodically cleared
- There is a specific operators role

Microsoft Forefront

MA Statistics

While the management agents tool shows the information about the most recent MA runs, the operations tool provides a history of previous runs. The information available for each run is very much as given in the management agents tool.

Other Actions

Additionally, from this tool you can:

- Run an MA.
- Stop an MA.
- Save the run history to an XML file.
- Clear runs (optionally up to a date you provide).

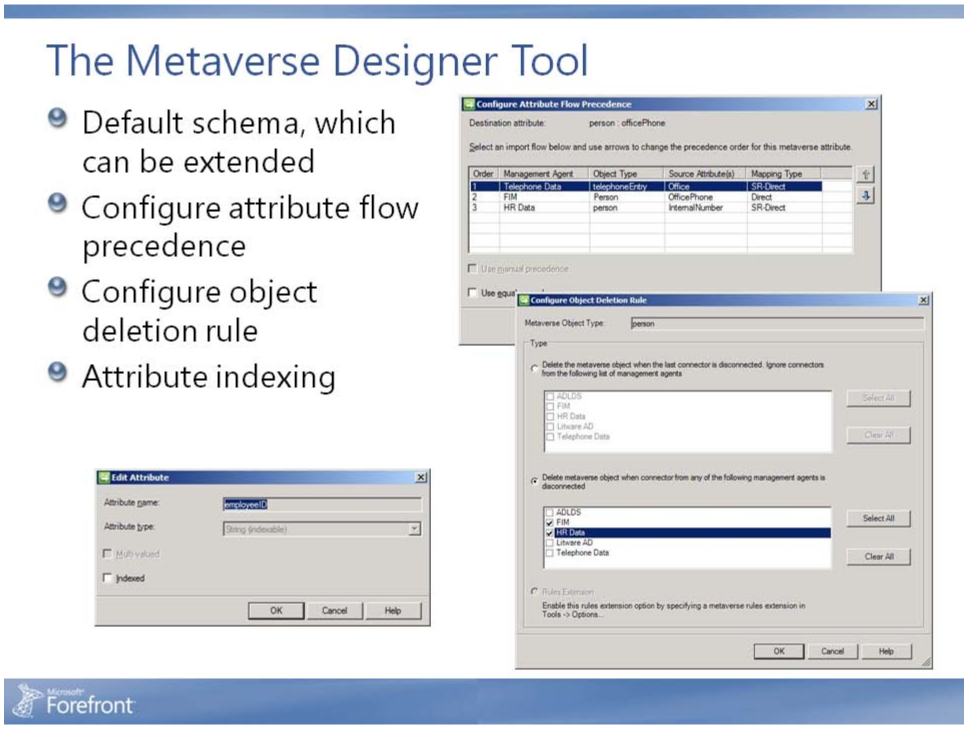
This last action is very important, because if you never delete the run history, you will eventually run out of disk space. It is usual to clear runs on a regular basis, whether here, or through a script.

> **Note:** If you view an older run of an MA, you will see the list of DNs; however the properties of the object are the current properties. If the object has changed since that run, only the current state is available—and the object could have since been changed or even deleted altogether.

The Operators Role

The FIMSyncOperators security group, created during the installation process, can be used to provide users with access specifically to this tool. Operators can run and stop MAs, review the run history, save a run to a file, and clear the history of runs.

The Metaverse Designer Tool



The Metaverse Designer Tool

- Default schema, which can be extended
- Configure attribute flow precedence
- Configure object deletion rule
- Attribute indexing

Configure Attribute Flow Precedence

Destination attribute: person : officePhone

Select an import flow below and use arrows to change the precedence order for this metaverse attribute.

Order	Management Agent	Object Type	Source Attribute(s)	Mapping Type
1	Telephone Data	telephoneEntry	Office	SR-Direct
2	FIM	Person	OfficePhone	Direct
3	HR Data	person	InternalNumber	SR-Direct

☐ Use manual precedence

☐ Use equal precedence

Configure Object Deletion Rule

Metaverse Object Type: person

Type

○ Delete the metaverse object when the last connector is disconnected. Ignore connectors from the following list of management agents

☐ ADLDS
☐ FIM
☐ HR Data
☐ Litware AD
☐ Telephone Data

● Delete metaverse object when connector from any of the following management agents is disconnected

☐ ADLDS
☑ FIM
☑ HR Data
☐ Litware AD
☐ Telephone Data

○ Rules Extension

Enable this rules extension option by specifying a metaverse rules extension in Tools -> Options...

Edit Attribute

Attribute name: employeeID

Attribute type: String (indexable)

☐ Multi-valued

☐ Indexed

The metaverse has a default schema that includes many commonly used object types and attributes. This schema can be extended, but that is beyond the scope of this course. However, here are some features worth covering.

Configure Attribute Flow Precedence

Precedence is quite a complicated subject. Sometimes more than one source can provide a value for an MV attribute, and you need a way to decide between them. For each combination of object type and attribute you can define an order, or choose equal precedence. There is also an option for manual precedence, for which you must provide (coded) rules extension rules.

Edit Attribute – Indexing

The only edit option for an existing attribute is to add an index. Indexes can affect performance—notably any MV attribute that is used for joining should be indexed.

Configure Object Deletion Rule

MV objects are only ever deleted as a result of a CS object being disconnected. Put another way, whenever a CS object disconnects from an MV object, the question arises: should the MV object be deleted? That question will be answered by looking at the MV object Deletion Rule.

Here are some reasons why a CS object might disconnect:

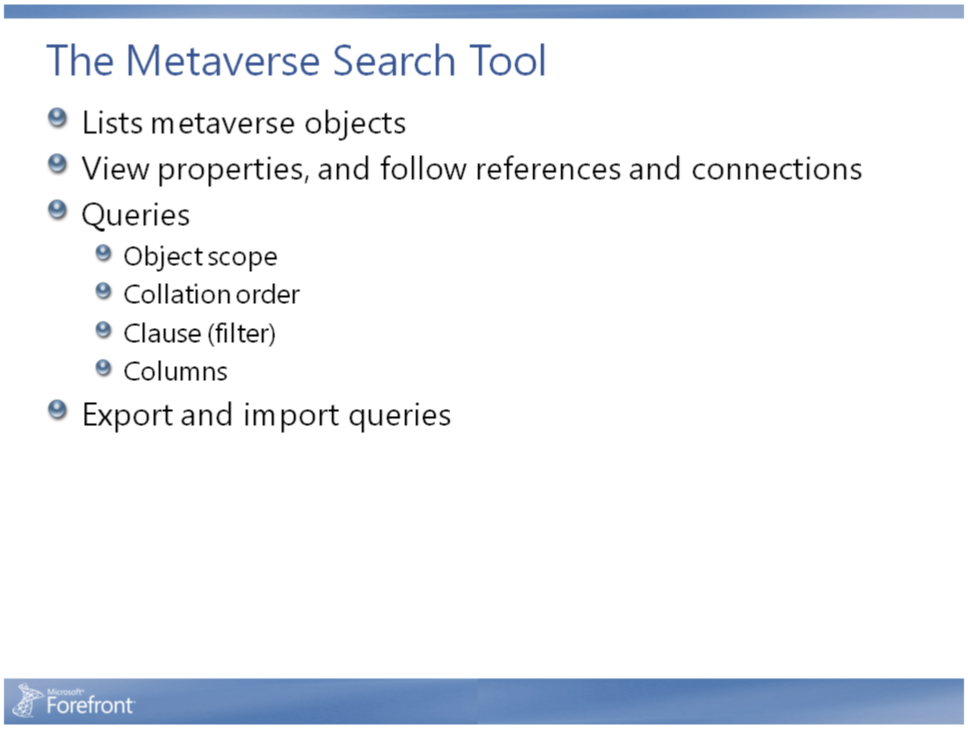
- Because the CS object got deleted (a deletion was imported from the data source).
- Because a filter action disconnected (for example, **filter status = terminated**).
- A manual disconnection (more unusual).

If the CS object originally projected this MV object (that is, it was authoritative for its creation) then it may well make sense for it to trigger deletion (that is, it should be authoritative for its deletion). This will not always be true, and MV deletion rules can be complicated—they can even be coded in a rules extension.

As an example, it may be that two systems can project new MV objects—FIM and HR Data. It would be typical for these to be selected as in the diagram.

Note: There is a default behavior in FIM that an MV object will be deleted when its last connector disconnects. Whichever of these options you take, you cannot switch off this default behavior which ensures that you do not end up with an orphaned MV object.

The Metaverse Search Tool



The Metaverse Search Tool

- Lists metaverse objects
- View properties, and follow references and connections
- Queries
  - Object scope
  - Collation order
  - Clause (filter)
  - Columns
- Export and import queries

Microsoft Forefront

The Metaverse Search Tool lists objects in the metaverse. Once you have your list you can view the properties of objects, including MV GUID, displayName, and object type. In fact, all non-null attributes will be shown, along with their value, which MA contributed it, the attribute type and when it was last modified.

References become hyperlinks that can be used to bring up the properties for referenced objects (such as a person's manager).

The connectors tab gives a list of connectors along with anchor value (or DN), the join method, and so on, and you can view the properties of these, too. In fact you can also disconnect them, deciding as you do so whether to make the CS Object a normal or explicit disconnector. Until you are very sure you know what you are doing, don't do this!

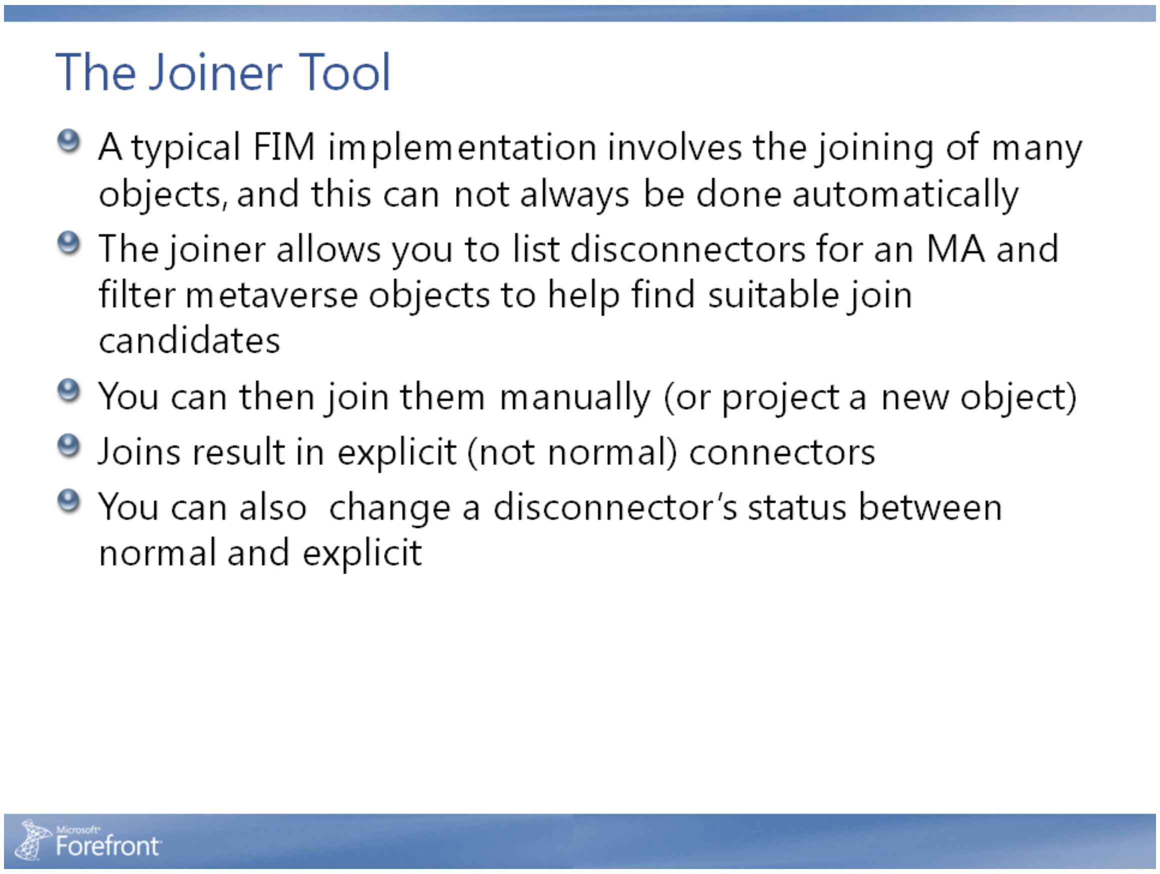
By default, the tool returns everything, and in a production environment, everything is unmanageable, so it is usual to define a query by configuring:

- An object scope – what object type do you want?
- Collation order.
- One or more clauses—a filter condition (such as givenName = Susan).
- Columns—which attributes and other properties you would like to see, in which order.

Then you can click **Search**.

You can export and import queries so that you can make use of them in future.

The Joiner Tool



The Joiner Tool

- A typical FIM implementation involves the joining of many objects, and this can not always be done automatically
- The joiner allows you to list disconnectors for an MA and filter metaverse objects to help find suitable join candidates
- You can then join them manually (or project a new object)
- Joins result in explicit (not normal) connectors
- You can also change a disconnector's status between normal and explicit

A FIM implementation involves the joining of objects from a variety of data sources, using an attribute (or attributes) that they have in common to identify which objects should be joined to each other.

Because real-life data is rarely perfect, the joining process may not be able to be performed fully automatically. Where there is ambiguity or missing data, you will be left with disconnectors—CS objects that are not joined to a corresponding record in the MV.

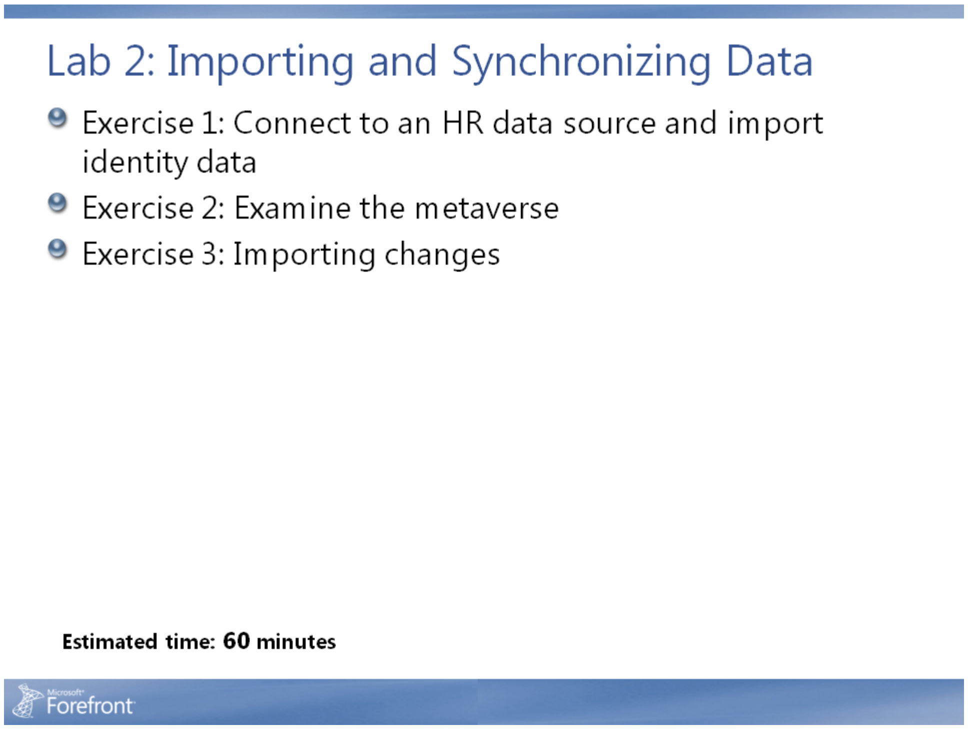
The joiner interface allows manual location of, and joining to, a suitable matching MV object for each disconnector. It allows you to list the CS objects for a particular MA, and define a filter to apply to the MV to help you find suitable MV objects candidates to join to. You can also choose to project a disconnector manually.

Note: If you make a manual join, the CS object involved does not become a normal connector, rather it is an explicit connector. This has the interesting property that a filter will not disconnect it (more of this in the next module).

One extra feature of the joiner—and this is the only place you can do this—is that it allows you to change the status of a disconnector between normal and explicit.

There is a specific security group for the users who perform this task—FIMSyncJoiners—which limits their access in Synchronization Service Manager to only the Joiner and Metaverse Search tools.

Lab 2: Importing and Synchronizing Data



Lab 2: Importing and Synchronizing Data

- Exercise 1: Connect to an HR data source and import identity data
- Exercise 2: Examine the metaverse
- Exercise 3: Importing changes

Estimated time: 60 minutes

Microsoft Forefront