Implementing Forefront Identity Manager 2010

Student Manual

Module 1: Introducing Forefront Identity Manager 2010

Table of Contents

Module 1: Introducing Forefront Identity Manager 2010

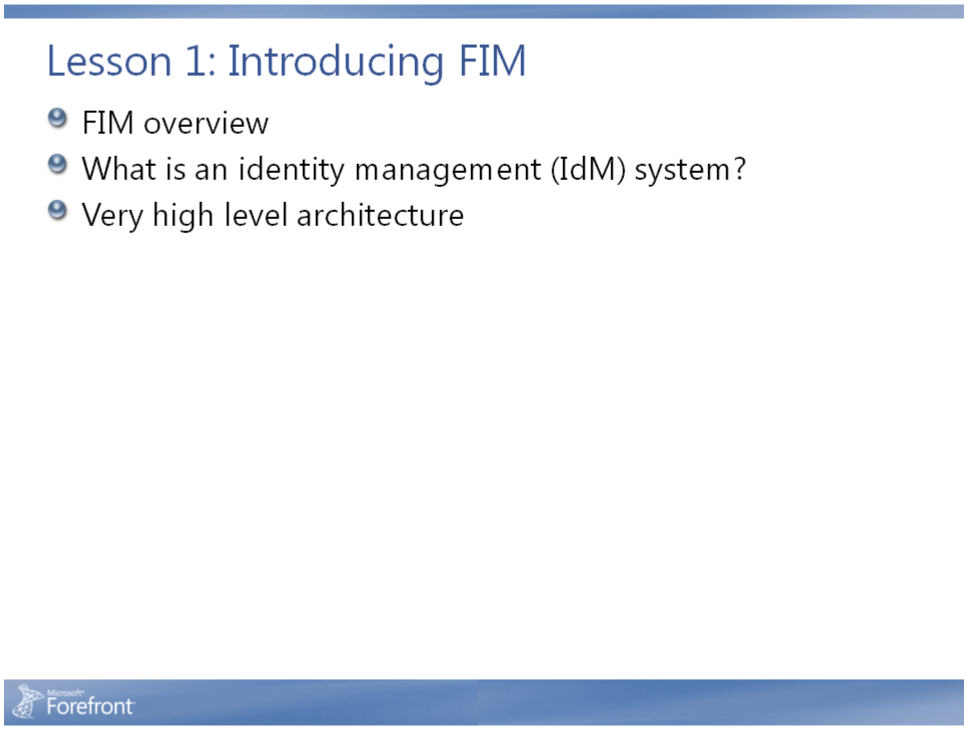
Module Overview

This module introduces Forefront Identity Manager (FIM) as an identity management (IdM) system developed from its predecessor Microsoft® Identity Lifecycle Manager (ILM). You'll learn what an IdM system is for and understand its high-level architecture.

The module goes on to cover the features of the synchronization service and the business problems it is trying to solve, before considering the other features of FIM (such as user management and policy management).

The module finishes with a walk-through of an already implemented FIM system, giving you the experience of managing users and groups through FIM.

Lesson 1: Introducing FIM



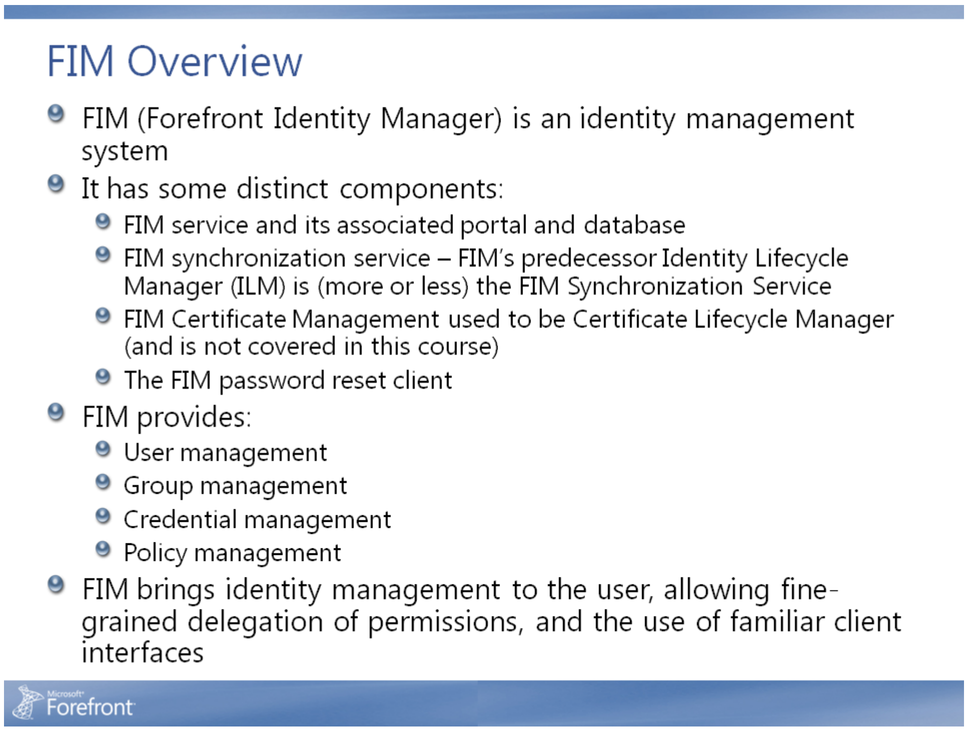
Lesson 1: Introducing FIM

- FIM overview
- What is an identity management (IdM) system?
- Very high level architecture

Microsoft Forefront

This lesson starts with an overview of FIM, its components, the services it provides, and the major features. It goes on to consider what an IdM system aims to achieve, and FIM's high-level architecture in relation to those aims.

FIM Overview



FIM Overview

- FIM (Forefront Identity Manager) is an identity management system
- It has some distinct components:
  - FIM service and its associated portal and database
  - FIM synchronization service – FIM's predecessor Identity Lifecycle Manager (ILM) is (more or less) the FIM Synchronization Service
  - FIM Certificate Management used to be Certificate Lifecycle Manager (and is not covered in this course)
  - The FIM password reset client
- FIM provides:
  - User management
  - Group management
  - Credential management
  - Policy management
- FIM brings identity management to the user, allowing fine-grained delegation of permissions, and the use of familiar client interfaces

FIM Is an Identity Management System

FIM is a development of Identity Lifecycle Manager (ILM), and in fact all of ILM is still there—it has now become the synchronization service (or sync engine). As a matter of interest, an existing ILM implementation should run just fine under FIM (though there are a few enhancements to the sync engine).

FIM Has Some Distinct Components

The FIM service, with it associated portal and application database, provides a management portal plus storage or identity and configuration data.

The FIM synchronization service can connect to a wide range of data sources, including directories and databases, and synchronize objects and attributes (FIM's predecessor ILM is more or less the FIM synchronization service.)

The FIM Certificate Management used to be Certificate Lifecycle Manager (CLM), and is still almost a separate product with its own portal, database, and workflows; it can be used, for example, to provision certificates to smart cards. This course does not cover FIM Certificate Management.

The FIM Password Reset Client, once rolled out to desktop, provides a self-service password reset facility—registering portal users by getting them to provide answers to questions that can later be used to authenticate them when they have forgotten their password.

FIM Provides a Number of Primary Solutions

User management – Users can be created, modified, and eventually deleted through the portal interface by administrators, delegated administrators (such as HR staff), and even the users themselves, but users can also be exported to the portal from other sources (such as HR applications).

Group management – Groups can be defined in various ways; criteria (query)-based groups can be populated automatically based on member attributes, and manual groups managed via user request and owner approval. Groups are particularly well aligned with Active Directory® groups, and it is easy to control access to Windows® resources based on user attributes.

Credential management – Self-service password reset is built-in, and smart cards and other certificate-based systems can be managed via FIM Certificate Management.

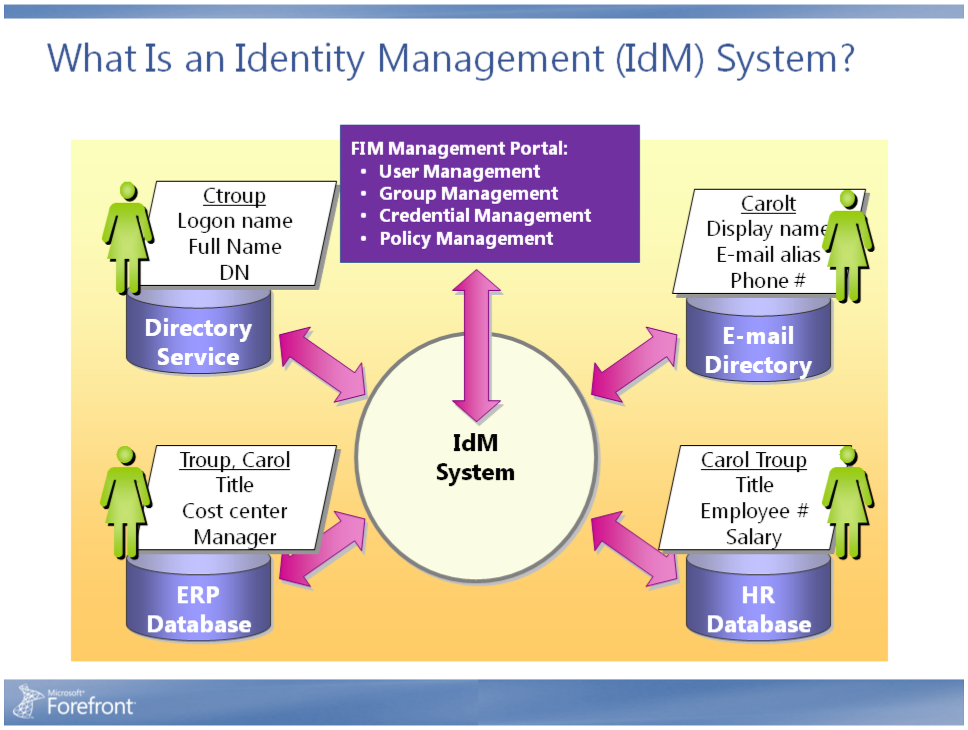
> **Note:** FIM Certificate Management is a substantial product component in its own right, with its own interface and database, and it deserves a training course of its own. This course does not cover it.

Policy management – This is really an overarching feature that applies to all of the above; policies can be defined to control permissions to data in the portal, to control how groups are managed (for example implementing owner approval), for provisioning and deprovisioning accounts in connect systems, and so on.

FIM Brings Identity Management to the User

ILM was invisible to most users, very much a server that worked in the background, whereas FIM is right out in front. FIM allows fine-grained delegation of permissions and the use of familiar client interfaces such as Microsoft® Outlook® 2010 and Windows® Internet Explorer®, making it very accessible to users to manage suitable attributes of their own, for example, in addition to their own group memberships and even their own distribution groups.

What Is an Identity Management (IdM) System?



Identity Management

Identity Management is the way we keep and manage information about things—usually people (but also computers, departments, printers, etc.)—and usually employees (but also contacts, customers, etc.) We are usually concerned about the access some things have to other things—primarily users' access to systems, hence Identity and Access Management.

What Is an Identity?

An identity is the summary of information about objects of interest, that is people, groups, or other resources such as computers and printers—or anything about which you wish to store data. This data is referred to as being contained in objects or resources. These objects are typically contained in different, and often incompatible, data stores—directories and databases—throughout an organization.

Identity Information Examples

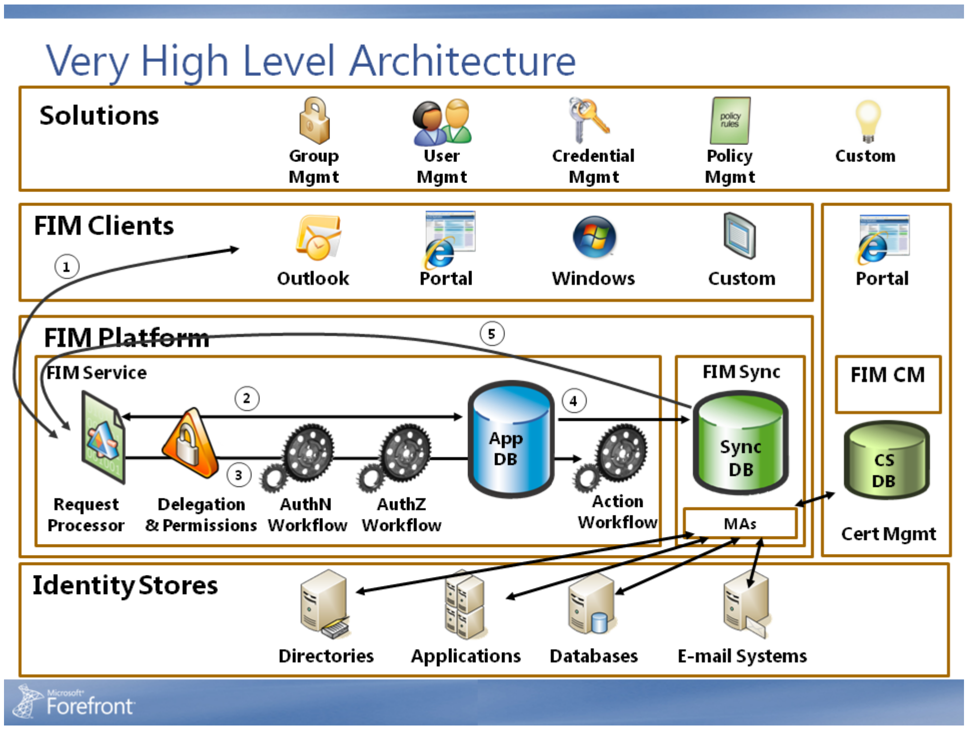
Examples of identity data attributes associated with person object types include names, e-mail addresses, telephone numbers, and job titles. A group object type might include a member list attribute; a computer object type might include a domain attribute; an application identity object type might include the network addresses where clients can find servers, or a list of services that applications can provide; a printer object type might include a location attribute, and so on.

As mentioned above, not all the objects an organization may be interested in managing are person object types. You can, in principle, represent any object type you require in this solution, be it users, computers, applications, and so on. It is certainly true that person object types are the predominant interest, and there are often different flavors of person objects, too, such as those that represent different clients or customers or perhaps even different roles in a single organization.

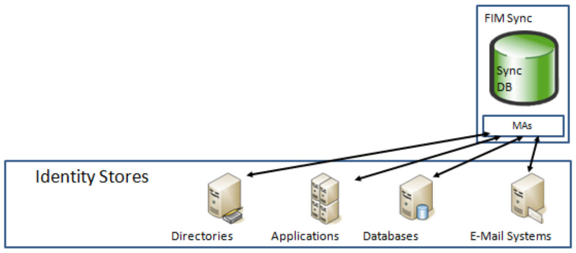
Identity Management System

An IdM system is a group of applications and/or services that coordinates information held in different data sources throughout an organization. In most organizations, this information is typically scattered in different directories, databases, and other data repositories throughout the Information Technology (IT) infrastructure. It is usually heavily redundant; it is likely to be inconsistent, giving rise to conflicts; it will generally be frustrating and expensive to administer; and it may be the cause of security holes. With the introduction of an IdM system, organizations can make their organizational data available in an accurate and timely manner, reducing costs through consolidation and process improvements, and reducing security problems caused by out-of-date access rights.

Very High Level Architecture

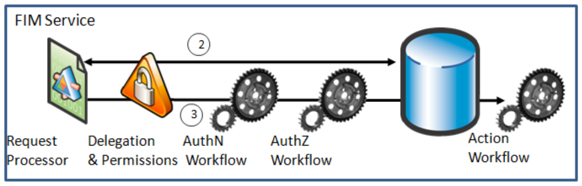


At this stage this diagram may seem daunting, but it's important to start somewhere, and when you return to it later, hopefully it will make more sense!



At the heart of FIM is the FIM synchronization service. This takes care of synchronizing heterogeneous data sources via various management agents (MAs). Note that it has a database, and that (as you will see) it maintains authoritative data about all the identities of interest gleaned from all sources.

FIM Certificate Management (formerly CLM) can be one such source, but in the diagram it is given a special status because although it is part of FIM, it has its own portal and database, and is not fully integrated with FIM's other data stores. In fact, you might correctly deduce from the diagram that it connects to the Synchronization Service in much the same way as any identity store does.



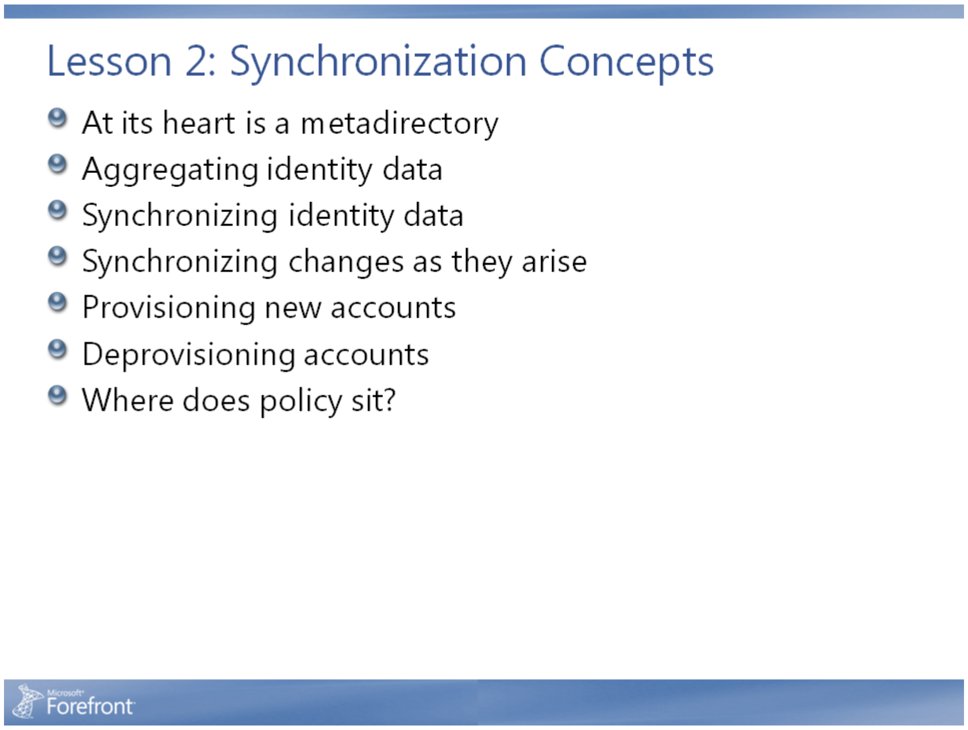
The other major component of FIM is the FIM service, with its application database. This database and the synchronization service database are closely coupled—it is okay to think of them at a high level as almost being the same thing, although at a detailed level each database can and does hold data that the other does not. Requests for access to the application database must pass through various gates; a read request (number 2 in the diagram) must have the correct permissions as defined in the service, and a write request (number 3 in the diagram) may have to go through additional authentication and authorization requirements, and after writing may trigger further actions (like provisioning actions).

Note that the synchronization service is privileged: it can read data directly (number 4 in the diagram), and although it writes via the service (number 5 in the diagram), it bypasses the authentication and authorization gates (not shown).

Solutions (such as user and group management) make use of familiar clients (such as Internet Explorer and Outlook 2010.) Whenever they read from and write to the application database they also do so via the FIM (Web) service in the form of a request (number 1 in the diagram).

Again, if that is a read request, they only need sufficient permission, but if it is a write request other authentications and authorizations may be required, and, once written to the database, such a request might trigger other workflows (such as notifications or provisioning actions). Thus the business rules can be enforced.

Lesson 2: Synchronization Concepts



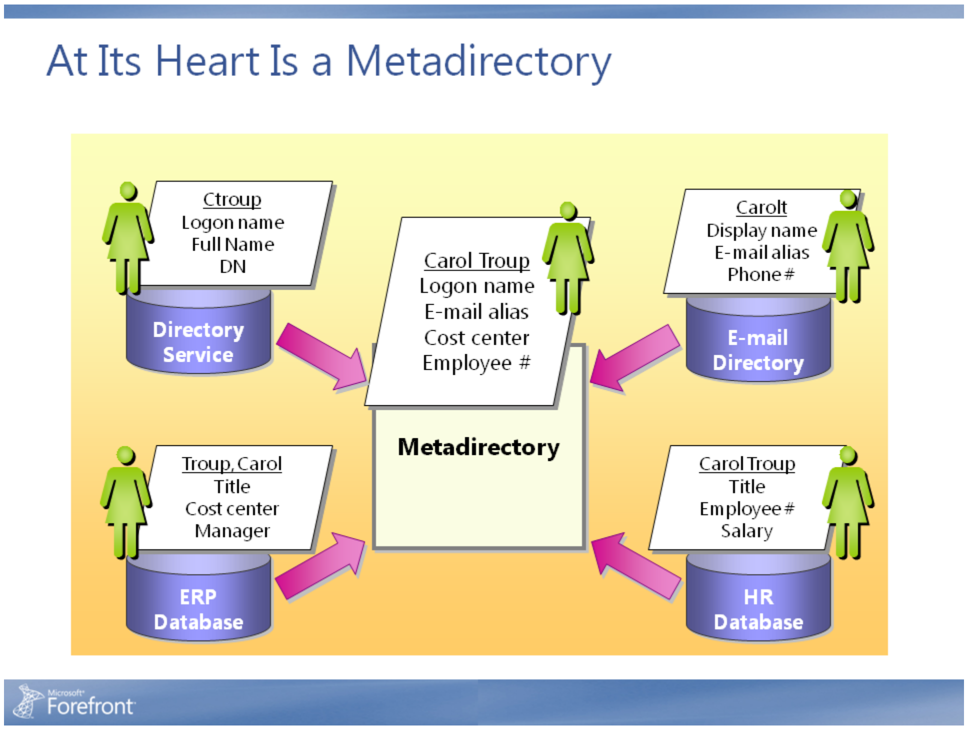
Lesson 2: Synchronization Concepts

- At its heart is a metadirectory
- Aggregating identity data
- Synchronizing identity data
- Synchronizing changes as they arise
- Provisioning new accounts
- Deprovisioning accounts
- Where does policy sit?

Microsoft Forefront

This lesson focuses on what is probably the most fundamental building block of FIM; without synchronization of data sources there would not be an identity management system. What does it do? What are the business problems it is trying to solve?

At Its Heart Is a Metadirectory

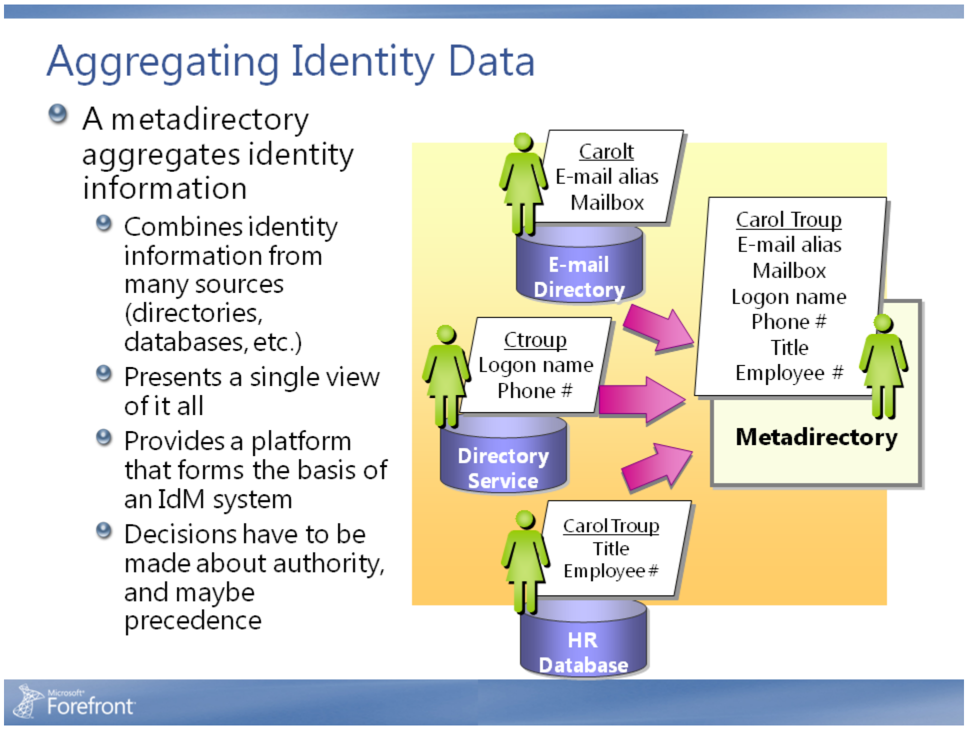


IdM systems can be designed in many different ways. FIM does not try to centralize all identity data in one single directory; neither does it try to take over lots of directories (for example by installing agents on each one). FIM is based around a metadirectory.

It is state-based and not event-driven: it imports the latest authoritative data from each directory and compares that data with what it last imported (it keeps a copy of the latest state in an area called the connector space), and is thus able to decide what is new, what has changed, and what has gone missing (presumed deleted). The latest data flows into the metaverse—the central part of the FIM data store which holds the combined authoritative data from all connected sources.

You will learn much more about the connector space and the metaverse later; the rest of this lesson focuses on the business problems that the Synchronization Service addresses.

Aggregating Identity Data



Aggregating Identity Data

- A metadirectory aggregates identity information
  - Combines identity information from many sources (directories, databases, etc.)
  - Presents a single view of it all
  - Provides a platform that forms the basis of an IdM system
  - Decisions have to be made about authority, and maybe precedence

Carolt
E-mail alias
Mailbox
E-mail Directory
Ctroup
Logon name
Phone #
Directory Service
Carol Troup
Title
Employee #
HR Database
Carol Troup
E-mail alias
Mailbox
Logon name
Phone #
Title
Employee #
Metadirectory
Microsoft Forefront

Aggregation

A metadirectory enables you to collect identity information from objects in different directories and then combine that information into a single object that represents the sum of all the authoritative identity information of interest.

In most organizations, identity information exists in many different data repositories, resulting in duplication of identity information; there is often no single reliable place to go for identity information about a person or resource.

Directories that hold identity information are often incompatible. These incompatibilities include different naming conventions, different directory schemas, different communication protocols, and different data formats.

Mapping Objects and Attributes

In combining the data for a specific person or resource, you have to identify the different objects (or accounts) and attributes (or fields) that correspond and create mappings between them. This is referred to as *joining objects* and *flowing attributes* (inbound).

The single, unified view will contain some or all of the attributes from the different directories, regardless of whether the directories are compatible.

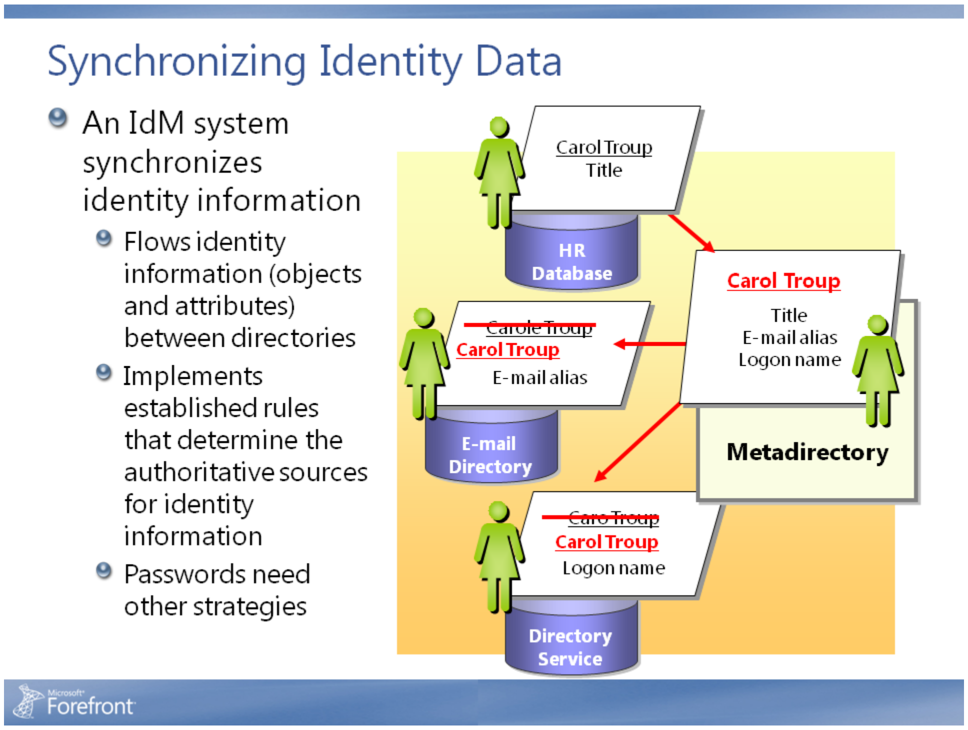
Authority and Precedence

You have created a platform that can become the basis of an IdM system—it contains the authoritative identity information for objects. In doing so you have identified which sources hold authoritative data.

Where there is more than one potentially authoritative source, you have to identify which is precedent—hopefully a formal order of precedence. As just discussed, you can aggregate identity information in the metadirectory.

At this point it is a short step to suggest that you also export data to (non-authoritative) sources that need it. See the next topic.

Synchronizing Identity Data



Synchronizing Identity Data

- An IdM system synchronizes identity information
  - Flows identity information (objects and attributes) between directories
  - Implements established rules that determine the authoritative sources for identity information
  - Passwords need other strategies

Outbound Flow Mappings

Once you have imported authoritative data, it makes sense to propagate it to non-authoritative sources (to the extent you want to, you can do so technically, and are allowed to do so by business policy). Much the same happens during export as happens during import, but in reverse—data is flowed out to where it is needed if it is different from what was last understood to be there (based on the copy in the connector space).

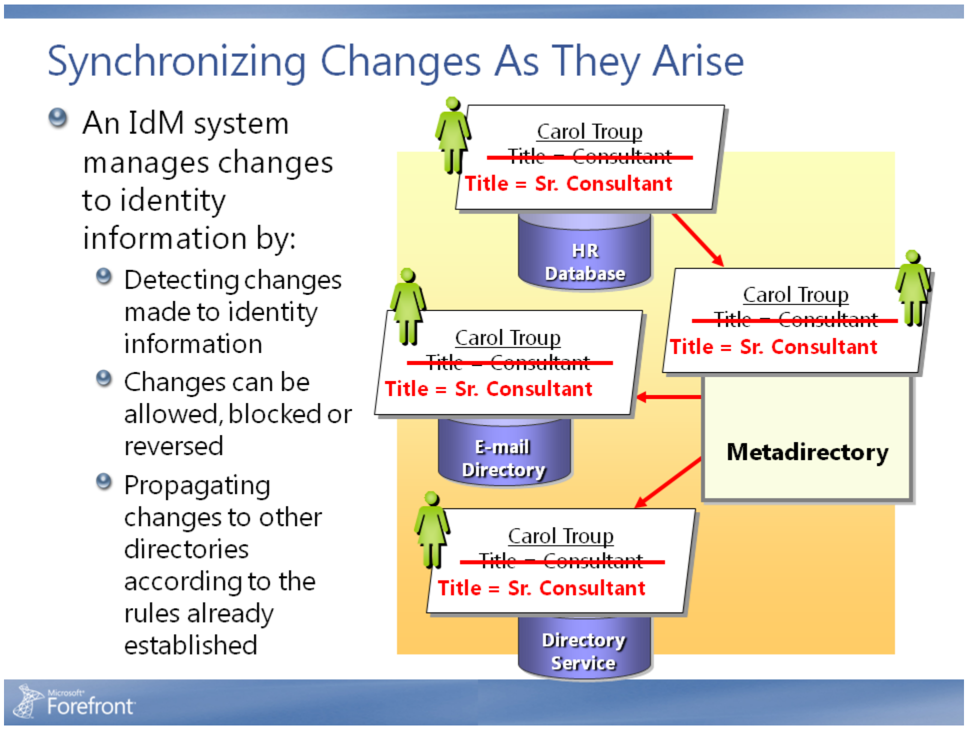
You must establish rules to determine how conflicts should be resolved, that is which data source out of a number of possible ones should take precedence, and update the other directories with that authoritative value.

Passwords

Note that passwords are a special case. It may be desirable to synchronize passwords, but you can't (or should not be able to) read them, so different strategies are required. This is covered in Module 6.

Presumably, if you can synchronize attributes across systems once, you can go on doing so as changes arise. See next topic.

Synchronizing Changes As They Arise



Synchronizing Changes As They Arise

- An IdM system manages changes to identity information by:
  - Detecting changes made to identity information
  - Changes can be allowed, blocked or reversed
  - Propagating changes to other directories according to the rules already established

As changes arise, of course you'll want the up-to-date information to be propagated to downstream (non-authoritative) systems.

Detecting Changes

One way to detect changes is to have some kind of agent sitting on a source, monitoring change. FIM is state-based, though, it is not event driven. It goes through a periodic process of importing the latest data and comparing it with a previously stored version (in the connector space). It then decides what has changed, is new, or has gone missing.

This concept of *delta* imports helps to optimize this process (in these cases, deletions are by way of an instruction to delete, rather than simply "missing, presumed deleted," which has the name *obsoletion*). This will be covered later in more detail.

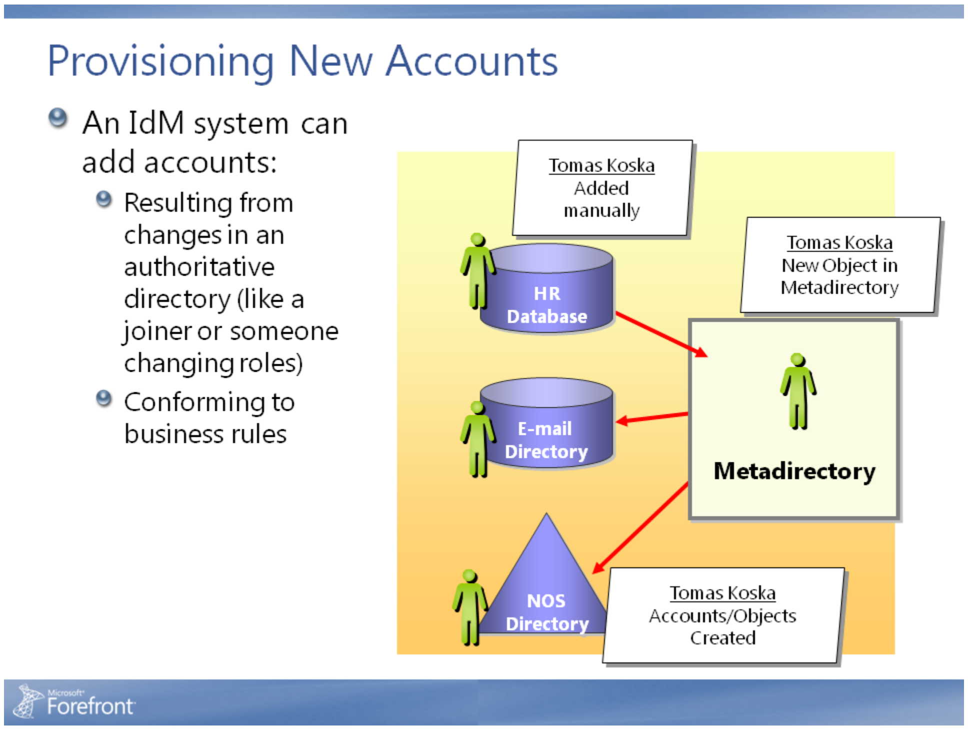
Changes can be allowed to flow into the metadirectory, can be blocked (leaving the source under local control with a non-authoritative value), or reversed (an authoritative value is forced out, reversing the change).

Propagating Changes

Changes can now propagate to other systems according to the rules already established.

So, for example, Carol Troup is promoted from Consultant to Senior Consultant. In the HR database, the value in the Title attribute is changed to Senior Consultant. When the metadirectory detects this change, the value in the Title attribute in the metadirectory is modified, and that change can be propagated to all other directories that contain a Title attribute. The rules you configure enforce integrity of identity information; you decide what is authoritative and should be allowed to flow in, what is not authoritative and should be overwritten, and what simply doesn't matter and can be left alone.

Provisioning New Accounts


Provisioning New Accounts

- An IdM system can add accounts:
  - Resulting from changes in an authoritative directory (like a joiner or someone changing roles)
  - Conforming to business rules

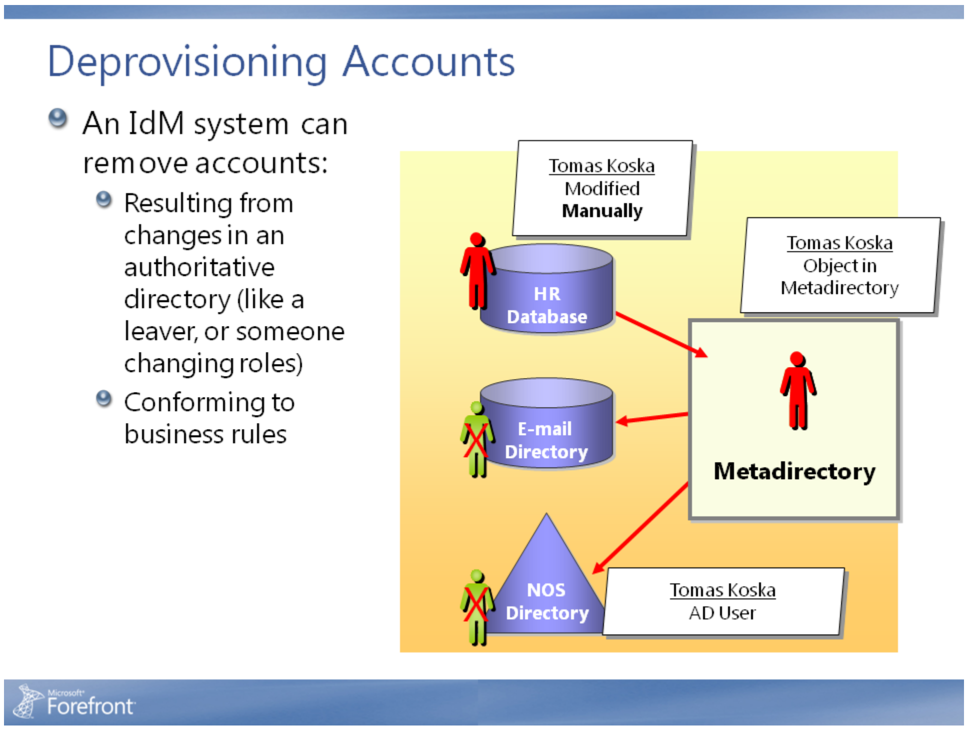
Once you have loaded all existing data and synchronized it, it is unlikely that you want to go on entering data in several places and then linking it up. It makes sense to identify one or a few places where you enter data and to treat these as authoritative sources for objects, which are imported.

Based on the rules you define, you can then make decisions about where to propagate objects—where to provision new objects and accounts. Once in place, you can flow attributes to them or from them (assuming that the target source generates some authoritative data of its own).

During the life of a particular identity, detected changes in attributes may require that you make further provisioning decisions. You may need new objects or accounts, or need to move them, or deprovision them (for more on this, see the next topic).

And of course it is just as important that you deprovision unwanted accounts when their useful life is over.

Deprovisioning Accounts



Deprovisioning Accounts

- An IdM system can remove accounts:
  - Resulting from changes in an authoritative directory (like a leaver, or someone changing roles)
  - Conforming to business rules

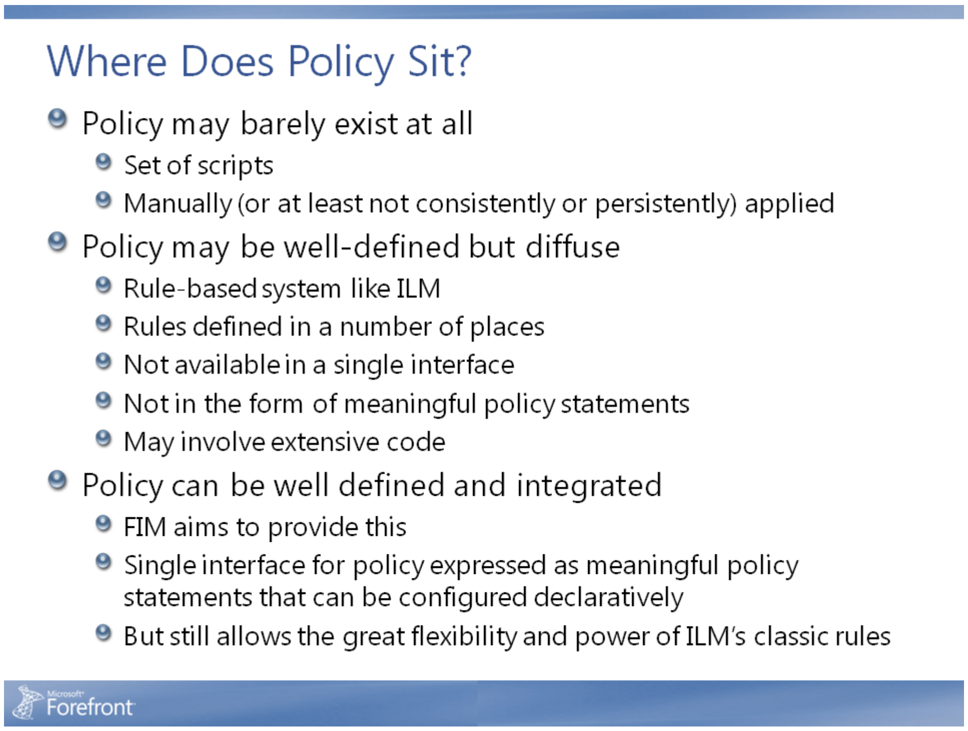
Deprovisioning is easy to overlook, but is just as important as provisioning. At the end of the life-cycle of an identity, it is likely that you'll want to remove some or all objects associated with it.

During the lifecycle, however, changes to that identity may dictate the need to deprovision and re-provision accounts many times. For example, an employee may changes roles, or leave and return, or take a sabbatical, or be suspended and reinstated.

The end of a life-cycle might not be a true end, in the sense that some representation of the object may stay in one or more systems for a long time. Based on the rules you define, you can make decisions about where to propagate deletions—where to deprovision objects and accounts.

All this must be done according to business policy. You must take business rules and turn them into identity management system rules.

Where Does Policy Sit?



Where Does Policy Sit?

- Policy may barely exist at all
  - Set of scripts
  - Manually (or at least not consistently or persistently) applied
- Policy may be well-defined but diffuse
  - Rule-based system like ILM
  - Rules defined in a number of places
  - Not available in a single interface
  - Not in the form of meaningful policy statements
  - May involve extensive code
- Policy can be well defined and integrated
  - FIM aims to provide this
  - Single interface for policy expressed as meaningful policy statements that can be configured declaratively
  - But still allows the great flexibility and power of ILM's classic rules

Synchronizing, provisioning, and deprovisioning must be done according to business policy. We must take the business policies or rules, and turn them into IdM system policies or rules. But where does that policy sit?

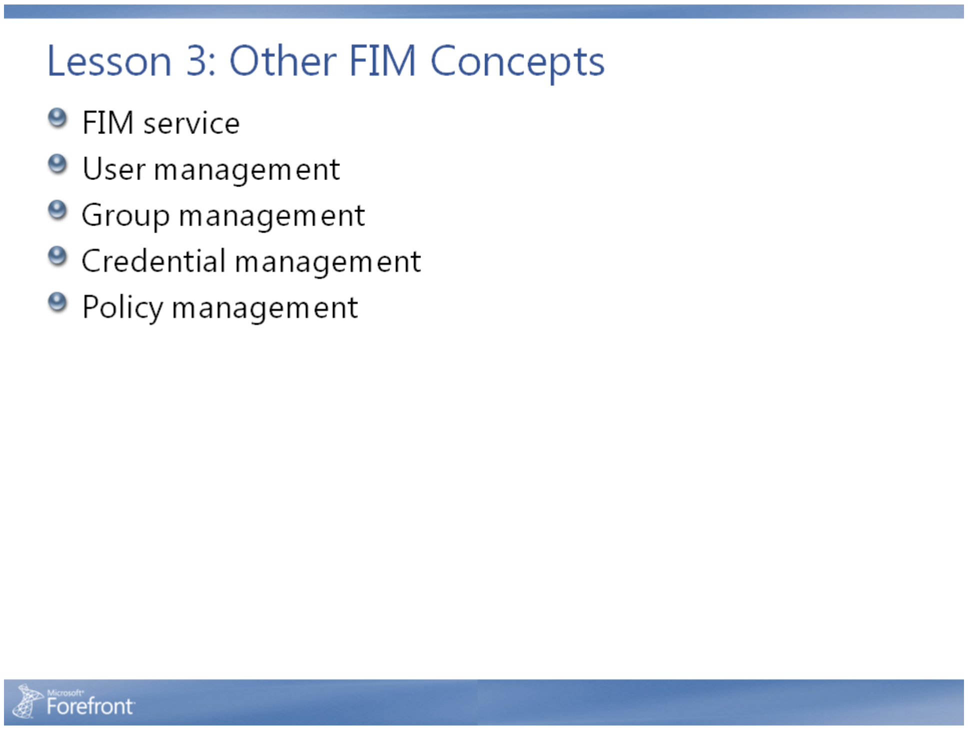
Homegrown systems often develop by adding scripts and point-to-point processes for synchronizing systems. This can be powerful, but also somewhat haphazard. Policies, if defined at all, are not rigorously and reliably enforced.

Straightforward synchronization systems—FIM's predecessor ILM might be seen as such, even though its design made it very powerful—tend to be configured as a set of rules that are not fully integrated into a central policy repository. These rules may not be in the form of meaningful policy statements that can be related back to business policies, and they may involve extensive code which is not accessible to non-programmers.

Policy can be well defined and integrated, and FIM, as you will see, aims for this. It has a single interface for policies that can be expressed as meaningful statements that relate to corresponding business requirements, and can be configured (for the most part) declaratively—in other words, in the Web UI and without resorting to script or code.

However, FIM maintains all the capabilities it had in its previous form (ILM) and can also make use of any Windows Foundation (WF) workflow. Although having two potential ways of doing things adds complication, the flexibility and power this brings makes this a worthwhile price.

Lesson 3: Other FIM Concepts

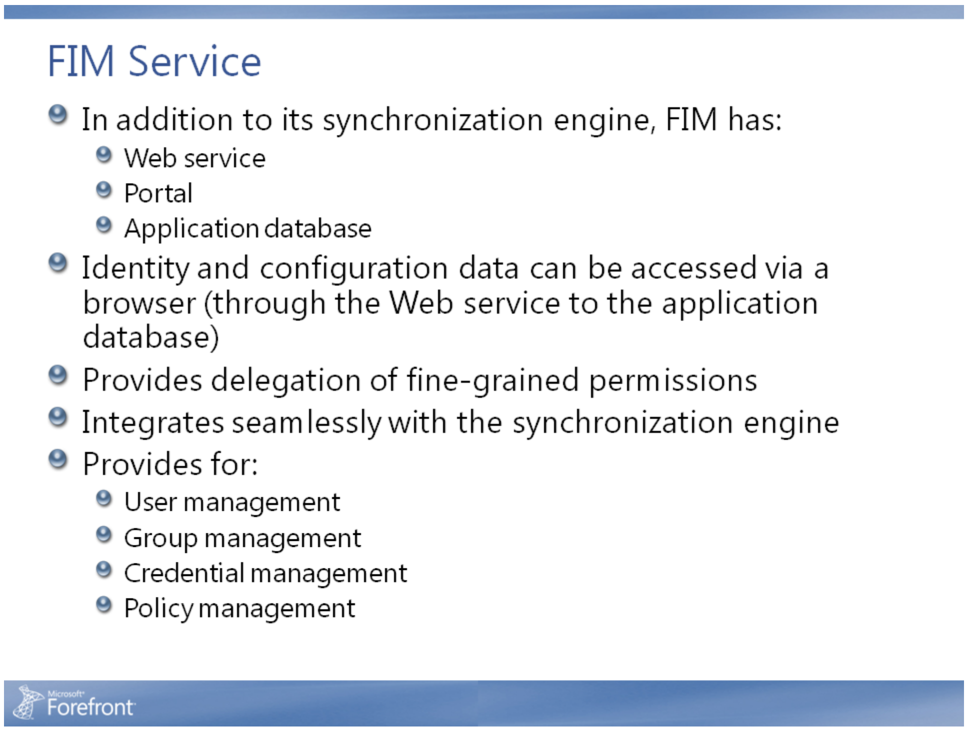


Lesson 3: Other FIM Concepts

- FIM service
- User management
- Group management
- Credential management
- Policy management

The last lesson covered the synchronization service—the bedrock of FIM as an IdM system. This lesson covers the other primary features of FIM—features that are provided mostly through the FIM service, but also through certificate management and the password client.

FIM Service



FIM Service

- In addition to its synchronization engine, FIM has:
  - Web service
  - Portal
  - Application database
- Identity and configuration data can be accessed via a browser (through the Web service to the application database)
- Provides delegation of fine-grained permissions
- Integrates seamlessly with the synchronization engine
- Provides for:
  - User management
  - Group management
  - Credential management
  - Policy management

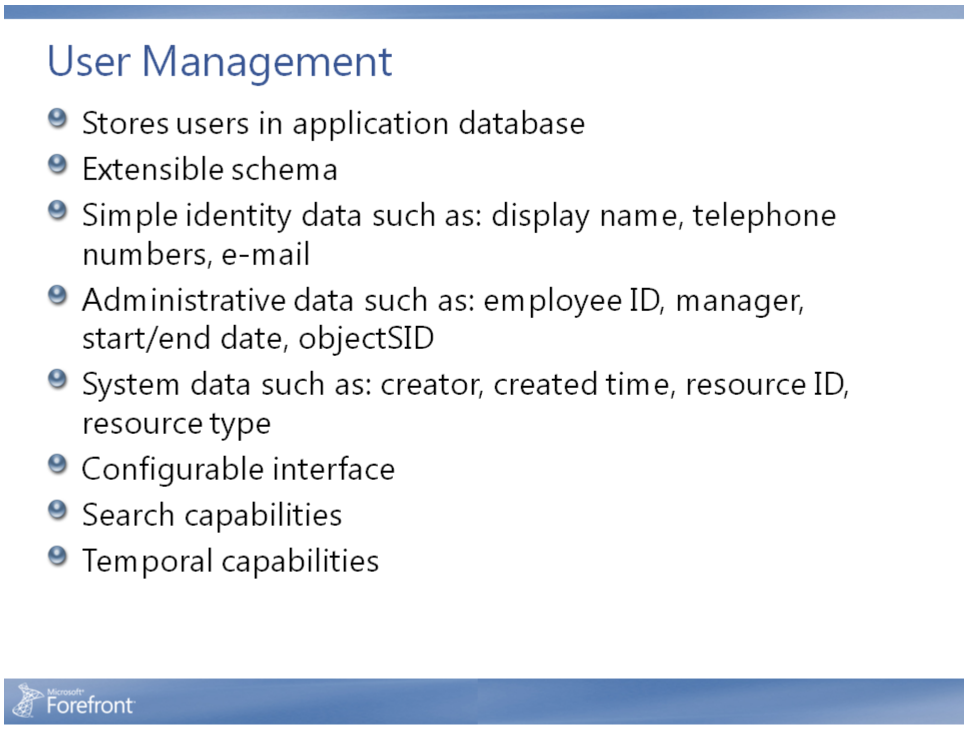
In addition to its synchronization engine, FIM has a Web service through which you access the application database. A portal, sitting on Microsoft® Windows® SharePoint® Services, provides the UI for accessing identity and configuration data.

The FIM Service has its own security token service (STS), and a powerful security model that controls user permissions down to attribute level so that you can allow users to change just single attributes of themselves and permit (and by omission deny) specific changes to specific attributes. For example, you could say that a particular set of users (perhaps those in the HR department) can read certain attributes of another set of users (perhaps full-time employees), and can modify a subset of those attributes, but that a particular attribute (perhaps the Employee Status attribute) can only be changed to specific values (perhaps full time to part time but not vice-versa; this level of control is arranged by creating sets of full time and part time employees and only allowing certain set transitions).

It integrates seamlessly with the synchronization engine. Identity data, and data that configures the synchronization service, is shuttled up and down between the application database and the synchronization service database—at a high level you can think of them as almost the same thing (though at a detailed level this is certainly not true).

The FIM Service, supported by the certificate management service and the Password Reset Client service, provides for user management, group management, credential management, and policy management, which are discussed in more detail in the following topics.

User Management



User Management

- Stores users in application database
- Extensible schema
- Simple identity data such as: display name, telephone numbers, e-mail
- Administrative data such as: employee ID, manager, start/end date, objectSID
- System data such as: creator, created time, resource ID, resource type
- Configurable interface
- Search capabilities
- Temporal capabilities

Microsoft Forefront

FIM provides a portal based on Windows SharePoint Services, which means that it already has some built-in functionality that is familiar to an increasing number of users. Data is stored in a Microsoft® SQL Server® Application Database.

A default schema is provided which includes simple identity data in the form of personal attributes, such as display name, telephone numbers, and e-mail; administrative data such as employee ID, manager, and start/end date; and system data such as: creator, created time, resource ID, and resource type. This schema can be extended.

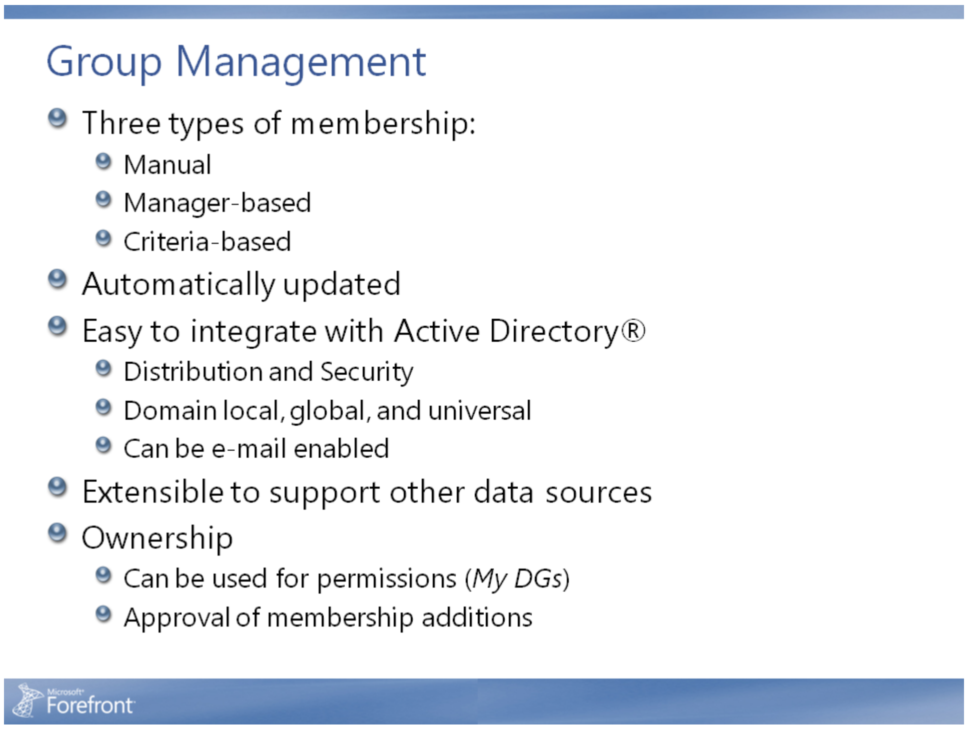
Some attributes are vital to the functioning of the portal. For example, the portal must have a user's Active Directory SID (objectSID attribute) in order for them to be able to use the portal. Many attributes are designed to line up nicely with Active Directory (AD) attributes so that management of AD accounts is facilitated.

> **Note:** Only the out-of-the box *user* resource type (also known as *person*) can be used to log on to the portal.

It has a default interface, which is also configurable. Entry forms can contain validation and other intelligent capabilities (such as lookup of values). The portal also has search capabilities built-in so that you can find users by display name, or account name, or whatever you configure.

It has temporal or time-based capabilities such as expiration of objects and warnings of impending events, and these can be extended, too, for delayed deprovisioning actions, for example.

Group Management



Group Management

- Three types of membership:
  - Manual
  - Manager-based
  - Criteria-based
- Automatically updated
- Easy to integrate with Active Directory®
  - Distribution and Security
  - Domain local, global, and universal
  - Can be e-mail enabled
- Extensible to support other data sources
- Ownership
  - Can be used for permissions (*My DGs*)
  - Approval of membership additions

Microsoft Forefront

Membership

There are three types of membership:

- **Manual** – Managed by the owner, or delegated user, possibly with owner approval.
- **Manager-based** – Automatically populated with a manager and their immediate reports; found by reference to data you have entered against each user (in other words it uses the manager attribute).
- **Criteria-based** – Automatically populated based on a query (actually an XPath query—more later). Groups are automatically updated as necessary when member (or potential member) data changes.

Types of Groups

There are three types of group resources available out-of-the-box:

- Distribution group (you would usually map these to Universal scope groups in AD, except in single domain situations).
- Security group (universal, global, or domain local scope).
- E-mail enabled security group (universal, global, or domain local scope).

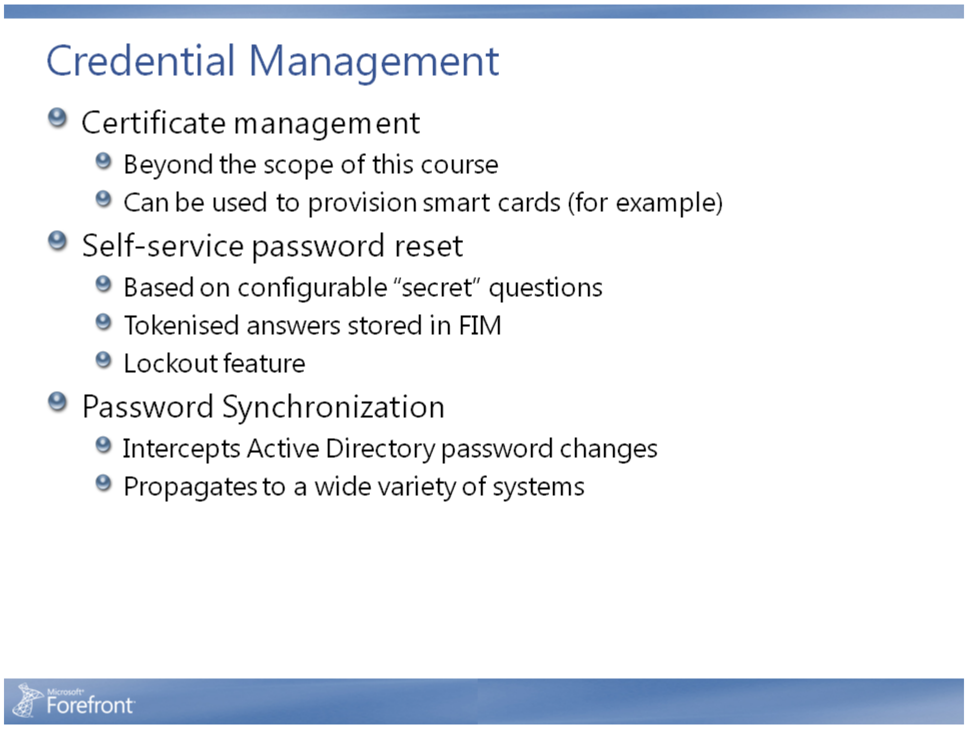
Integrates with Active Directory but Extensible to Support Other Data Sources

By default, group resources are oriented towards AD. It has been made easy to manage AD groups and Exchange distribution lists (DLs) without further modification. However the schema can easily be extended to provide additional attributes required by other connected systems, or even to create new group-like resource types. The forms used for accessing group data can also be modified to allow for different requirements.

Ownership

Groups have an owner. That ownership can be used to control permissions so that people can modify their own groups (for example). It can also be used for approval of membership additions, which can be done via the portal or via e-mail.

Credential Management



Credential Management

- Certificate management
  - Beyond the scope of this course
  - Can be used to provision smart cards (for example)
- Self-service password reset
  - Based on configurable "secret" questions
  - Tokenised answers stored in FIM
  - Lockout feature
- Password Synchronization
  - Intercepts Active Directory password changes
  - Propagates to a wide variety of systems

Credential management really applies to two areas: certificate management and password management (which itself splits into two).

Certificate Management

Certificate management is provided by FIM certificate management. This is a significant module in its own right, with its own interface, database, workflows, etc., and it deserves its own course. It is certainly beyond the scope of this course. (As matter of interest to some, this used to be CLM.)

FIM certificate management can be used to manage (provision, revoke, etc.) certificates of various kinds; notably it can manage smart cards.

Self-service Password Reset

FIM ships with the necessary workflows, other configuration, and password client to manage the registration process for self-service password reset. It does this via a series of configurable secret questions. It is not, of course, the questions that are secret, it is the answers, and these are stored in a tokenized form in a registration object associated with the user concerned.

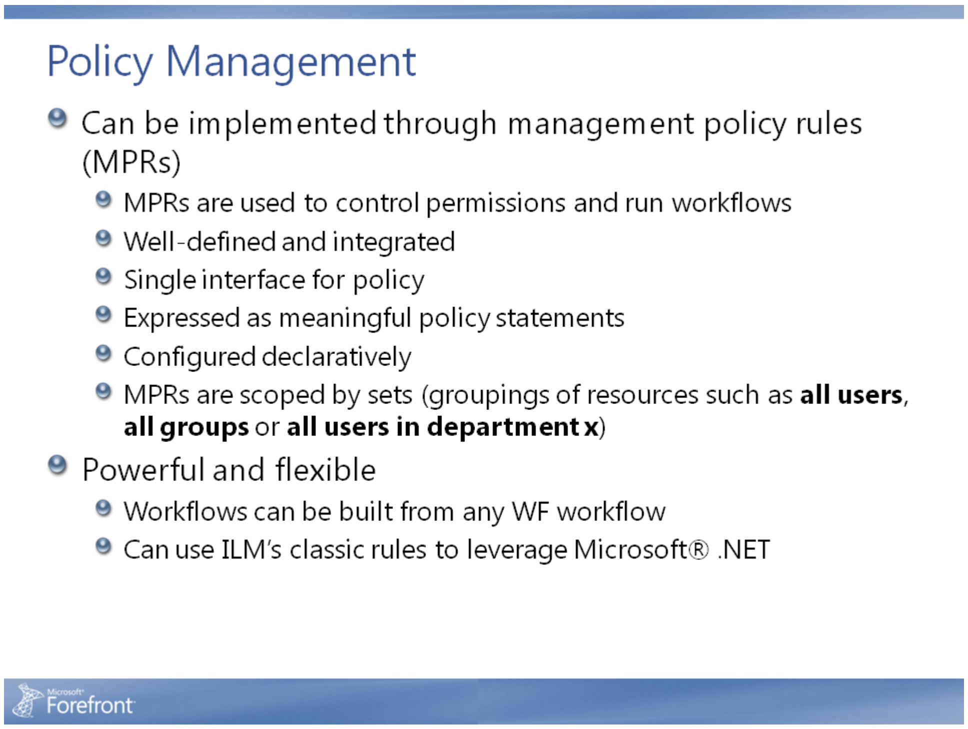
When a user forgets their password they can use the client (accessible via the Windows log on dialog) to access the questions, provide the answers, and (if they match) reset their password in AD.

FIM also supports a lockout feature that cuts in when a user responds to their questions incorrectly a number of times (which you can configure).

Password Synchronization

FIM also supports password synchronization (in exactly the way ILM did). It intercepts AD password changes, and propagates them to other data sources (as configured). It can send passwords to a wide variety of directories and other sources, and it is possible to code extensions for those sources not included out-of-the box.

Policy Management



Policy Management

- Can be implemented through management policy rules (MPRs)
  - MPRs are used to control permissions and run workflows
  - Well-defined and integrated
  - Single interface for policy
  - Expressed as meaningful policy statements
  - Configured declaratively
  - MPRs are scoped by sets (groupings of resources such as **all users**, **all groups** or **all users in department x**)
- Powerful and flexible
  - Workflows can be built from any WF workflow
  - Can use ILM's classic rules to leverage Microsoft® .NET

Microsoft Forefront

Policy Is Implemented through Management Policy Rules (MPRs)

You will be hearing a lot about MPRs on this course. Firstly, they are used to control permissions, and these permissions are cumulative.

For example, one MPR might give read permission to any requestor for all attributes of the target object, and another might give modify permission to some attributes provided the requestor is in the All HR Set, while yet a third might give modify permission to some other attributes provided the requestor is in the All Managers Set. So the net effect is that the requestor (an HR manager) may have both read and modify access to the various attributes mentioned. These permissions can also be made dependent on the target object being a member of a Set (and maybe staying as a member of that Set even after the operation).

Secondly, they can run workflows. These workflows can be of three types: authentication (when doing a password reset), authorization (when obtaining owner approval for a group membership), or action (a notification or provisioning action).

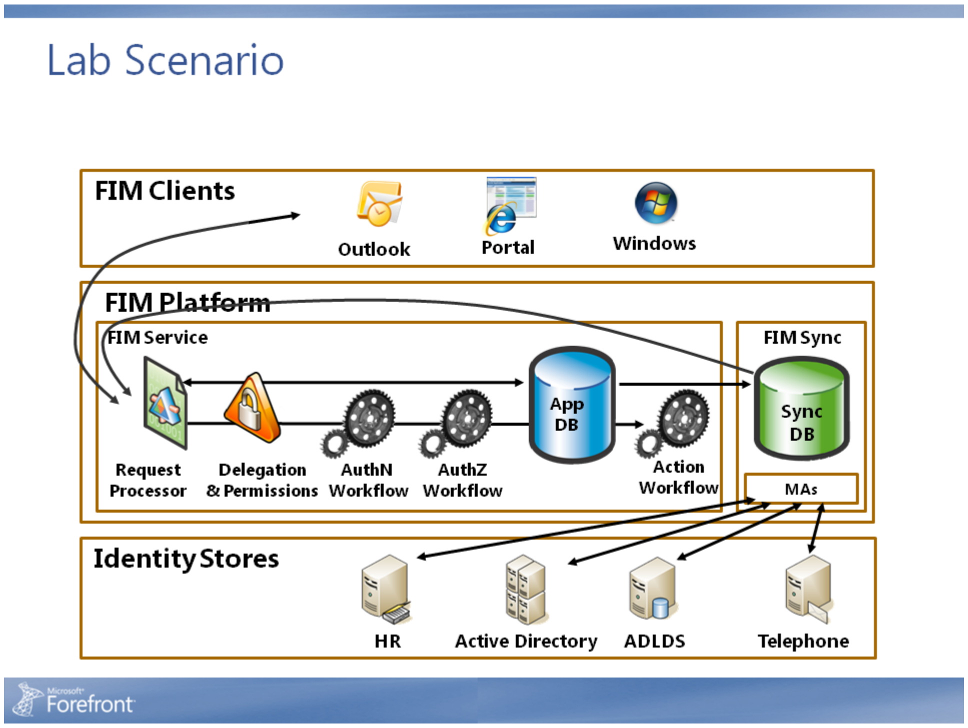
The portal provides an interface, search facilities, and so on for MPRs just as it does for users and groups. Thus you have a single interface for managing policies, which can be expressed as meaningful policy statements relating to actual business policies.

For the most part they can be configured declaratively, making heavy use of sets (groupings of resources like all users, all groups, or all users in department X.)

Powerful and Flexible

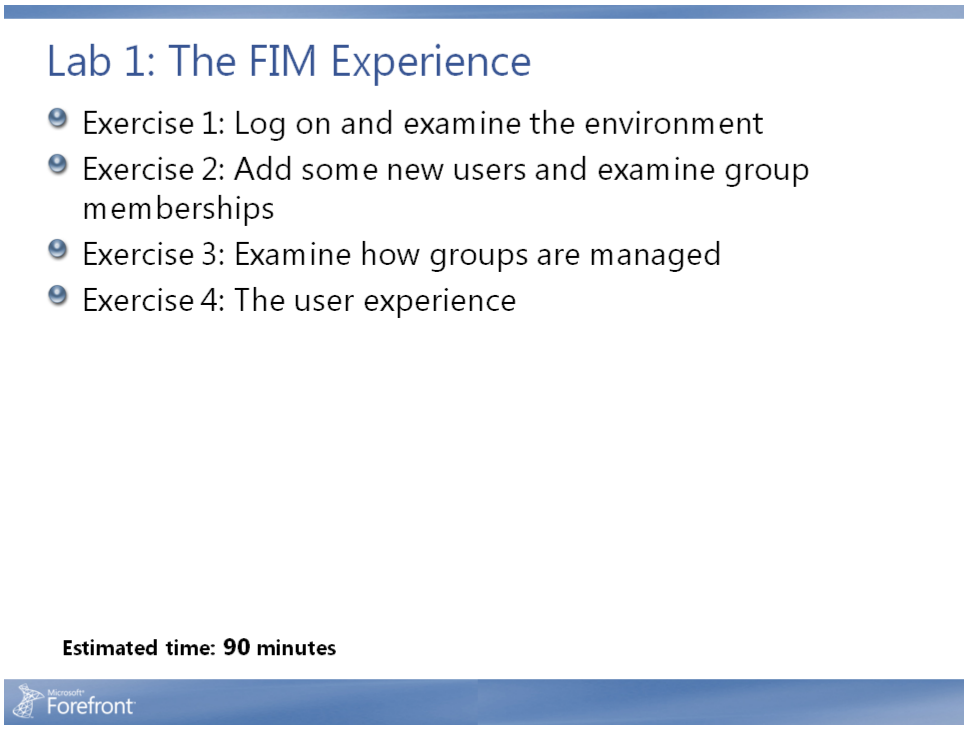
MPRs trigger workflows, and these workflows can be built from any WF workflow, which gives access to a huge range of possibilities. That range is further increased by the fact that you can also use synchronization rules defined directly in the synchronization engine, generally referred to as classic rules—and these can leverage the wider power of .NET programming and libraries.

Lab Scenario



Lab Scenario

Lab 1: The FIM Experience



Lab 1: The FIM Experience

- Exercise 1: Log on and examine the environment
- Exercise 2: Add some new users and examine group memberships
- Exercise 3: Examine how groups are managed
- Exercise 4: The user experience

Estimated time: 90 minutes

Microsoft Forefront