Table of Contents

Security And Protection

How To

give different authority to different users

Most multi-user systems require that different categories of users have access to different features of the system. In ***AwareIM*** this is done by assigning different access levels to different users – see the "Access Level" section.

limit access to database tables

In ***AwareIM*** you are not dealing with database tables directly – instead you can limit access to business objects – see the "Setting Access to Business Objects" section for details. You can also limit access to business objects based on some conditions – see 1040_conditional_access.

limit access to records

In ***AwareIM*** you are not dealing with database records directly – instead you can limit access to attributes of business objects – see the "Setting Access to Business Objects" section for details. You can also limit access to attributes based on some conditions – see 1040_conditional_access.

limit access to operations

Most operations are invoked from forms - see How to use different forms for different users. You can also limit access to processes and services – see the "Adding/Editing Access Levels".

You can also define an "applicability condition" when defining an operation, for example:

```
Member.Status = 'Active'
```

The operation will only be shown for those members whose status is Active.

define access based on some conditions

If you want to limit access to business objects or its attributes based on some conditions you have to specify business rules that use the PROTECT action. See the description of the PROTECT action in the Rule Language Reference for details.

get end users to define their own access restrictions

Sometimes it may be necessary for end users (for example, system administrators) to override access restrictions supplied by developers. This could be the case if developers sell their application to many different customers each of them having their own unique access level requirements.

This can be done as part of the “User Defined Processes/Workflows” module. A developer needs to include a menu command of the “Configuration of User Defined Processes type”. This menu command brings up the user defined processes configuration module that allows among other things to define access restrictions to business objects and attributes.

implement multi-tenancy

It is often required that the same application is hosted for different “tenants”. A tenant logs into the system and only sees data that belongs to him, but not other tenants.

There are two approaches in ***AwareIM*** on how this can be implemented:

1. Each tenant has a separate business space allocated to the tenant and the same application is loaded into each business space
2. All tenants are managed by the same business space and the application itself is defined in such a way that each tenant’s data is protected from another tenant

There are pros and cons in both approaches, so the approach you should take depends on your specific requirements.

The advantage of the first approach is that each tenant is completely independent from each other because each business space is totally separate from another one. Moreover, you can allocate a separate database to each business space, so that the data of each tenant is stored in its own database. The application is simpler because there is nothing in it that refers to tenants. Data backup can be done on a tenant-by-tenant basis and a data problem in one tenant doesn’t affect another tenants.

The disadvantage of the first approach is that if there are thousands of tenants, existence of thousands of business spaces can significantly impact the performance of the server (however, you can add more servers to alleviate the problem). Another disadvantage is that if your applications needs functionality that works across different tenants – for example, if you want to collect statistics about your tenants, look at all registered tenants etc, it may be hard to do if all your tenants are independent applications.

Advantages and disadvantages of the second approach are reverse of that of the first one. The second approach may not depend as much on the larger number of tenants as the first approach and it’s also quite easy to provide functionality that looks across different tenants. But the application itself becomes more complicated and it is a lot more difficult to provide backup/restore on a tenant-by-tenant basis (but not impossible).

The following paragraph describes what needs to be done in the application to implement the second approach. This is what you need to do:

1. Define an object representing a Tenant

2. For all objects that belong to the tenant add a reference to the Tenant that it belongs to, for example `Account.Tenant`
3. The user object must have a reference to Tenant that it operates on behalf of - `LoggedInSystemUser.Tenant`
4. For each object that belongs to some tenant add rules to initialise tenant from the tenant of the logged in user that created the object, for example:

```
IF Account.Tenant IS UNDEFINED THEN
 Account.Tenant = LoggedInSystemUser.Tenant
```

5. Finally add protection rules to the object that belong to the tenant so that other tenants do not see it, for example:

```
IF Account.Tenant <> LoggedInSystemUser.Tenant THEN
   READ PROTECT Account FROM ALL EXCEPT System
```

set up SSL

AwareIM uses web server such as Tomcat or Weblogic to process web requests. The default web server that comes with ***AwareIM*** is Tomcat. If you want ***AwareIM*** to use SSL protocol for the web requests you can set up the web server to use SSL. Please refer to the documentation of the web server. See also the AwareIM Installation Guide.