

# **ATG Dynamo and Eprise Participant Server Integration demo**

## ***A White Paper***

*Authored by:*  
*Frank DeBenedetti (Eprise)*  
*and*  
*Vivek Khadilkar (Infobase)*



Eprise Corporation  
200 Crossing Blvd.  
Framingham, MA 01702  
508-661-5200  
888-658-7037



39899 Balentine Dr., Ste. 185  
Newark, CA 94560  
(510)661-2000  
<http://www.infobaseusa.com>  
[info@infobaseusa.com](mailto:info@infobaseusa.com)

## **Introduction**

The ATG-driven Consolidated Financials demo, created by InfoBase for Eprise, is intended to demonstrate how Eprise Participant Server can be integrated with ATG Dynamo, thereby enabling both applications to be deployed in a manner consistent with their core strengths.

This white paper discusses implementation strategies for database driven websites in general, and includes specific examples of Eprise Participant Server and ATG Dynamo working together. The reader should be aware, however, that similar strategies might be taken with Eprise Participant Server and other applications and/or application servers including Microsoft Active Server Page, Cold Fusion, HAHT Commerce, BEA Web Logics, and IBM Web Sphere.

This white paper explains how to use XML/HTML File repositories for content delivery from ATG Dynamo.

## **Basics of ATG Dynamo Personalization**

ATG Dynamo personalization (ATG DPS) is driven by user profile data and business rules designed to deliver the right content to the right user (Readers familiar with both Eprise and ATG should recognize some overlap between Eprise & ATG DPS.) Leveraging the strengths of both applications, we designed a solution where Eprise manages the content, and ATG DPS delivers the content. ATG provides the targeted viewing as well, based upon user-specified preferences and user profile attributes.

This section outlines the key elements of the ATG DPS personalization system:

- User Profile Management
- Content Targeting
- Targeted E-Mail

### ***User Profile Management:***

When a person visits a website driven by ATG Dynamo Personalization Server (ATG DPS) web site for the first time, s/he is allowed to create his/her own user profile. Once created, DPS stores that user's **profile** in its database repository. This profile contains a list of properties that describe the person's characteristics, such as the name they entered in a registration form (a self-reported, or *explicit*, attribute) or the date of their last login (a system-generated, or *implicit*, attribute). ATG DPS uses this profile information stored in its database repository to provide targeted content to each user.

### ***Content Targeting:***

Targeting is the process of displaying content items to a particular user, at a particular time, in a particular context and on a particular rule set. In the DPS *rule-based* system, business managers create rule sets called **content targeters** that control how content is displayed on the Web site.

## Targeted E-mail

DPS includes a Targeted E-mail service for composing and delivering personalized e-mail using the same profile groups and targeting rules you use to deliver content on your Web site. (**Note:** If you have Dynamo Scenario Server installed, you can use scenarios to deliver targeted e-mail.) For example, you can use targeted e-mail to:

- Send a confirmation message to a new user who registers at your site.
- Notify frequent customers of special sales.
- Notify all users that have not logged in to your site in several months that their accounts will be closed soon.
- Send out a mass mailing with each message tailored to its recipient.

## How ATG Dynamo Personalization Server works with Eprise

This is a commonly asked question, as readers familiar with both products will realize that both Eprise and ATG DPS have the capability of personalizing content to various audiences. In the ATG-driven Consolidated Financials demo, all personalization for individual user experiences on the website is handled by ATG DPS. "Content management" components, such as the ability for a business user to edit content on the site, submit modified content into a workflow, approve content, and publish that content to the website, all without knowing any html or scripting, are provided by Eprise Participant Server. As a result, each page of content throughout the entire demo is rendered by ATG DPS, including the pages that display the Eprise "little blue edit buttons."

The screenshot shows a web browser window displaying the 'CF Tech Growth Fund Product Data Sheet'. The page features a navigation menu on the left, a header with 'consolidated financials WELCOME TO EMPLOYEE INTRANET', and a main content area. The main content area is divided into several sections:

- CF Tech Growth Fund**: Includes 'Manager Name' (Vivek) with an 'Edit' button, 'Commission Structure' (Level 1 - 5.00%, Level 2 - 5.75%, Level 3 - 6.25%) with an 'Edit' button, and 'How Has This Fund Performed?' with a line graph showing performance from 1997 to 2000.
- CFTGEX**: Includes 'Quick Stats' (NAV, Day Change, YTD Return, Morningstar Rating) and 'Inside Scoop' (text describing the fund's risk and performance).

Small blue 'Edit' buttons are visible next to the Manager Name, Commission Structure, and Inside Scoop sections.

The little blue edit buttons provide the ability to specify which users can edit various components on a particular web page. In the example shown above, taken from the Consolidated Financials demo, the business user has the ability to edit the Fund manager name, the Commission Structure, and the Inside Scoop components. Each

of these components are called blocks in Eprise. Once a business user clicks a little blue edit button, a request is made by ATG DPS to get the Eprise "edit quick form" page. ATG DPS passes the object id of the desired block with the request to Eprise via an Eprise Integration Agent. Eprise processes this request, first by determining whether the user is authorized to make the request, and then passes the results as a stream of html back to ATG DPS for rendering. For ATG-driven websites, Eprise Participant Server provides the customized Eprise "Quick Forms" that enable business users to contribute content and edit pages without knowing anything about html, or about Eprise Participant Server, or about ATG DPS. This enables the business user to edit content while viewing the site as it is deployed, served up by a Dynamo server.

To achieve the integration for the ATG-driven Consolidated Financials demo, we are using Scenario 1 from "[Eprise Participant Server Integration with ATG Dynamo White Paper](#)." In Scenario 1, once a page or piece of content is approved and published within the Eprise content repository, Eprise also pushes the new or modified document to the Eprise file system content repository using Eprise physical mapping feature. So, in this scenario, content managed by Eprise Participant server is first published (or pushed) into physical XML file system. As mentioned earlier, ATG DPS also supports XML/HTML file based repositories. ATG DPS periodically scans this file-based repository, extracting metadata from new content it discovers, then pulls that content into its database repository.

### **Eprise Content Separation Model**

To maximize performance and efficiency, we utilized a content separation approach, so that only content components where business users have edit capabilities are actually published to the file system repository in XML tags. These XML-based components are then included by the ATG DPS display page(s) that contain the remaining components such as header, footer, navigation, etc.

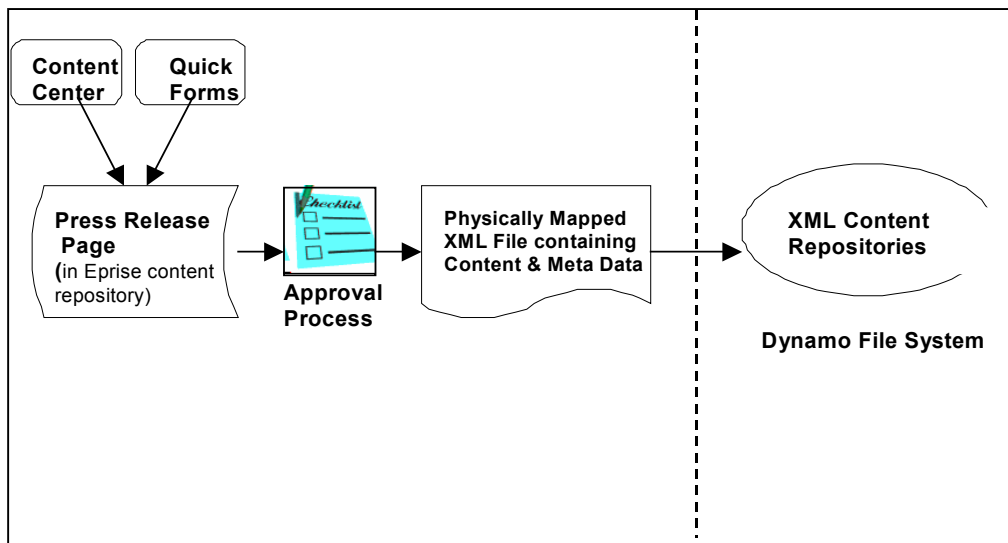
Eprise uses variables it calls "merge strings" to collect the parameters that are editable by the business user. These merge strings can be embedded in the html/XML code. Eprise also employs an approach known as content separation (aka component model) so that pages can be used for different purposes. A "content" page is used to collect data from the business user through the inclusion of merge strings, whereas other "display" page(s) are used to display or wrap what was collected by the content page with the appropriate tags for XML export or HTML look and feel.

In the Consolidated Financials demo described above, the CF Tech Growth Fund content page contains three blocks editable by business users. One for the Fund manager name, one for the Commission Structure (text), and one for the "Inside Scoop" blocks, each of these blocks are based upon elements containing merge strings that enable the business user to contribute and/or edit the content within that block.

When a business user clicks the little blue edit button for any of these three editable components, an Eprise Edit Quick Form page (called by an Eprise integration agent) containing the last published content is displayed by ATG Dynamo. If the business user modifies that content and submits those changes for approval, then the modifications are saved in Eprise for the purpose of passing the changes through a workflow and/or approval process but are not re-published to the file system until those changes are approved.

In the ATG-driven Consolidated Financials demo, content approvers use the Eprise User Interface for approving changes submitted for approval. Eprise offers an Approval Quickform so that it would be possible for business users to perform the approval without leaving the context of the website.

Once the page has been approved and published within Eprise, it is physically mapped to the file system, where the the ATG DPS API retrieves the updated page components from its XML repository and populates its database repository.



Using the following steps, a developer can better understand the process.

Steps:

1. Create CF Demo pages in Eprise
2. Configure physical mapping of the CF folders in Eprise Content Catalog by adding entries to map.cfg
3. Stop and restart the web server-related services
4. Republish the CF Demo content folder(s) in Eprise, which will in turn publish the content into the designated physical folder on the file system
5. After some specified time interval (e.g. 30 seconds), ATG DPS will refresh it's repository component which we have created for that specific folder
6. When the user refreshes the page or visits another page on the site, they will see the newly created/modified content.

## Frequently Asked Questions (FAQs)

***"I'm still not clear where exactly Eprise strategic content management played a role in the demo."***

The short answer is anytime a content editor is viewing an Edit quick form to edit a particular content page, or approving the content, Eprise is driving the user experience. ATG DPS is calling an Eprise Edit Quick Form through our integration agent, passing the objectid and/or instanceid of the desired content. Eprise returns the desired content as a stream of html to be rendered by ATG DPS.

Review the following steps to demonstrate:

1. User enters login name as ceditor and password as Eprise to login to the ATG-driven website.
2. Now logged in to the site as ceditor, you will see the link on the cf\_home page to "Edit this page" towards the bottom of the page, not displayed to anonymous viewers.
3. If the ceditor clicks on the "Edit this Page" link, the same page will be redisplayed by ATG DPS, this time, with little blue edit buttons.
4. The ceditor can then click on any one of the "little blue edit buttons" on a page still being rendered by ATG DPS.
5. Next, a Java bean calls an Eprise Quick form using Eprise Integration Agent, then hands over the control to Eprise to manage the content. This is the point where Eprise kicks in.
6. The quick form is displayed by Eprise and when the Submit for Approval and/or publish buttons are clicked, the content is first written into the Eprise Content repository. This allows changes to flow through the Eprise workflow and approval process prior to being published back into the ATG DPS repository. As soon as the content finishes its approval process within Eprise, and is published in the Eprise database repository and the Eprise file system repository, the content immediately becomes available to the ATG DPS file system repository as well.

***"So...what exactly is the Java bean doing?"***

This question (as well as an understanding of step 5) requires a conceptual understanding of the Eprise Participant Server event-driven architecture as well as Eprise Integration Agents. To start, since all Eprise functionality can be achieved through a browser, it should be clear that Eprise runs as an ISAPI or NSAPI plug-in to a web server. As such, Eprise dynamic link libraries (dll's) provide a suite of extended functionality to a web server, in as much as the 250 Eprise server-driven events enable business users to perform a whole host of content management functions using only a browser. These same events can also be called from an application server via Eprise Integration Agents. Thus, an application server can make a request for a particular event by passing one or more parameters to Eprise through our proprietary XML gateway. For example, an application server could request that Eprise create a particular page based upon a particular page style, stored in a particular file folder by passing that event name and the required parameters for the event to Eprise through our integration agent. The event is executed by Eprise, after evaluating the request against our business rules engine, which then returns a stream of html or event success parameters to the application server again through our xml gateway.

So one custom ATG DPS Java bean passes the authentication credentials of the user to Eprise, and another bean requests the appropriate event, page, or page style, then a third bean receives the html of the requested page back from Eprise for ATG rendering of the Edit Page Quick Form using the content separation model.

***"How many customized Java Beans do I need to achieve this sort of integration on my website (one per site, one per event, one per page, one per page style, etc)?"***

The count of Java beans is dependent on the requirement and the complexity of the web server. It is always better to have one single bean for authentication, a

separate bean for page rendering, and a separate bean for calling edit quick forms. Each separate edit function or called Eprise event would require its own Java Bean in the current architecture. However, the same bean can be used for displaying and/or calling any number of page styles, as well as for any number of pages. We currently have three beans in the demo, an authentication bean, a rendering bean, and a bean that calls and edit quick form.

***"How are the authentication credentials shared between ATG and Eprise? How do I keep the authentication credentials in synch if they are housed in separate repositories?"***

The authentication can be shared between ATG DPS and Eprise. We can use the same Eprise users for ATG DPS personalization. Eprise and ATG DPS both have their own internal authentication databases, and if it is desired that both authentication databases be maintained, then a process would be required to keep both in synch. However, both ATG DPS and Eprise can also share a common LDAP repository, as well as their authentication so that changes can occur in one repository that both Eprise Participant Server and ATG DPS can share.

***Why did you use Option 1 from the Eprise-ATG Integration White Paper?***

Option one was selected for the demo because it was the fastest option to deploy. We were under a timeline crunch to get the demo put together for the ATG Open. Since both ATG and Eprise support database driven repositories as well as the XML file based repositories, Eprise certainly could write the content into an ATG database repository instead. However, if XML-based repositories catch on, (see <http://www.content-wire.com/Home/Index.cfm?ccs=86&cs=279>) the approach used in the demo is right on track.

***How does the Eprise approach to integration with ATG compare with Interwoven's approach?***

Interwoven's approach is totally file based and as such does not allow the editing of content without switching applications. As a result, using the Interwoven approach, the business user can't edit content in ATG DPS from ATG DPS. They have to open the Interwoven application, modify the content in Interwoven, and then republish the content in Interwoven to the content in the shared file-based repository. Also, since Interwoven is only file-based, it does not support SQL repositories or LDAP.

***Why not use JSP by itself?***

Since we want to use ATG DPS we need to use JHTML, since ATG does not recommend the use of JSP scripts. Also, as JHTML separates the java code and the HTML code it is functionally easier for the project manager to separate the two teams (i.e. HTML developer team and the JAVA developer team).

***What are the basic steps in the integration process?***

- EPS writes content into XML files using approach described above.
- One or more repository components are created inside ATG DPS. While creating these components, you need to create one to one relation between the repository component's attributes and each node of the EPS-generated XML file repository. e.g.

```
<newsarticle>
<author>Robyn Hitchcock</author>
<headline>Globe of Frogs</headline>
```

```
<date>5/28/1999 00:00:00</date>
<targets>international</targets>
<content><![CDATA[ <p> It's a globe of frogs out there. </p>
]]></content> </newsarticle>
```

For the Consolidated Financials demo, we created an ATG DPS repository component with following attribute for the news article:

author  
headline  
date  
targets  
content

- Repeat the above step to create a repository component for each different type of XML file in that repository
- For each repository component created, specify how frequently you want to refresh the repository. Each time ATG DPS refreshes its cache from the EPS-generated repository, ATG DPS keeps the values in its cache and uses the cache for content delivery.
- After creating the repository component(s) you need to create targeter component(s). Using targeter components, ATG DPS can deliver the content to targeted groups of users.
- Next, you need to create the actual HTML (.jhtml) pages within ATG DPS that will be used to deliver this content.
- Include these created components within the actual HTML (.jhtml) pages
- XSL pages can also be used in either Eprise or in ATG DPS to provide personalized views of the xml content as well.

(For more information please refer to ATG Dynamo DPS manuals.)