$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

The OpenEAI Integration Analysis Template

Revision history: the OpenEAI project strongly recommends that you use a versioning system such as CVS, Subversion, or something similar to track changes to analysis artifacts, and that you clearly indicate the current revision of artifacts within them. For example, see the revision information in header on each page of this document.

Application Name:	**Enterprise Greeting Service**
Module Name (if appropriate):	**OpenEAI Example Enterprise**
Phase (if appropriate):	N/A
Integration Timeframe:	N/A
Target Template Completion Date:	N/A
Template Owner Name:	**Open Integration Incorporated**
Total Work Estimate:	N/A
Resources Named in Template:	N/A
Location of Detailed Project Plan:	N/A
Template Change Reviewers:	N/A

Template Status:

Step 1 ☒	Step 2 ☒	Step 3 ☒	Step 4 ☒
Step 5 ☒	Step 6 ☒	Step 7 ☒	Step 8 ☒
Step 9 ☒	Step 10 ☐	Step 11 ☐	Step 12 ☐
Step 13 ☐	Step 14 ☐	Step 15 ☐	Step 16 ☐

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Overview:

The integration analysis and design process consists of the following steps. **Steps 1 and 2 should be completed prior to the integration design meetings with EAI staff.**

Step 1: Describe Existing Integrations – Describe and define present application interfaces for the application named in this template. This information will be used to help define enterprise data objects for use in messages when design teams meet with integration staff. It is recommended that this section be completed prior to integration design meetings with integration staff.

Step 2: Describe Data Involved in Proposed Integrations – Define the data for which the application named in this template is authoritative. That is, which data in this application will other applications need. Also describe which data this application will need for which other applications are authoritative. This information will also be used to help define enterprise data objects for use in message when design teams meet with integration staff. It is recommended that this section be completed prior to integration design meetings with integration staff.

Step 3: Describe the Flow of Data in the Proposed Integrations – Define (at a high level) the flow of data between the application named in this template and other applications to support the interfaces defined in steps 1 and 2. This section should be completed during the integration design phase with integration staff.

Step 4: List Existing and New Enterprise Data Objects Required for the Integrations – List existing enterprise data objects that will be reused and new enterprise data objects that will be defined to support application interfaces for the application named in this template. This section should be completed during the integration design phase with integration staff.

Step 5: Name the Messages that will be used to Implement the Integrations – Name the messages that use the enterprise data objects to implement the integration flows. This section should be completed during the integration design phase with integration staff.

Step 6: Name the Existing and New Messaging Applications Required – Name the existing messaging applications that will be used or define the new messaging applications that must be implemented to produce, consume, transform, and route the messages listed in step 5. This section should be completed during the integration design phase with integration staff.

Step 7: Provide Technical Stories for the Primary Application – For the primary application named in this template, list the messages it must produce and consume and provide detailed stories describing the prescribed production and consumption logic. The “owner” of the application, the individual responsible for implementing message production and consumption, should prepare the detailed stories. This section should be completed during the integration design phase with integration staff.

Step 8: Provide Technical Stories for the other Applications Named in the Template – For each remaining application listed in step 7, list the new messages each application must produce and consume and provide brief stories describing the prescribed production and consumption logic. This section should be completed during the integration design phase with integration staff.

Step 9: Summarize all Outstanding Questions, Issues, and Action Items – Record all questions, issues and action items during the analysis and design sessions.

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 10: Perform Work Estimation and Scheduling – Identify resources and estimate work for each aspect of the project enumerated in the template and prepare a detailed project plan.

Step 11: Functional Approval of Analysis, Scope, and Scheduling – Functional team members must signoff to indicate completion of the analysis, design, and planning.

Step 12: Initiate Template Change Control Processes – Identify the integration change reviewers and begin the process of convening them to review every subsequent change to the template. These reviews should answer the following questions and ensure the appropriate follow-up actions are taken:

- Does the change impact the scope or timeline established for the integration?
- Does the change impact the overall design of the integration?
- Has the impact been properly reflected in the project plan and timeline?
- Do all approvers who initially approve this template also approve of this change?

Step 13: Provide Testing Specifications – Develop detailed testing specifications for the integration. This will usually consist of one or more OpenEAI test suites and some functional application use cases that can be used to certify the functionality of the integration. Typically, test specifications will also include a load testing strategy for the integration, which may again use OpenEAI test suites or involve testing the integration in a production-like environment. Finally, some production readiness test procedures should be specified. This testing can be performed in production after the integration is deployed to verify that it is properly deployed. The procedures can be reused whenever changes are made to the production deployment of this integration. Usually, production readiness testing is implemented with an OpenEAI test suite or application use-case scenarios that can be automated using monitoring or testing tools.

Step 14: Develop and Test Integrations – Develop the required message support for the gateways and applications using the technical stories and test specifications provided in the template. Most development processes are iterative, so changes in stories and requirements should be documented and trigger the template change control process. In this step, tests are often expanded and improved as a concrete picture of the integration functionality emerges.

Step 15: Prepare Implementation and Rollout Plan – Prepare a deployment diagram that visually depicts the deployment and complete an implementation rollout plan enumerating the steps required to implement the new integration in production.

Step 16: Implement in Production – Execute the rollout plan and production readiness testing.

The next sections of the document describe the details required for the steps listed above.

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 1: Describe Existing Integrations

Describe and define present application interfaces for the application named in this template. This information will be used to help define enterprise data objects for use in messages when design teams meet with integration staff. This section should be completed prior to integration design meetings with integration staff.

What data does this application store and operate on that it does not create itself? Typically, this data is acquired through batch extracts and feeds, remote procedure calls, or data replication. For example, a payroll history database and an employee change of status application may both maintain and in some cases update employee job data. This data in the employee change of status application may be synchronized with the payroll system by scheduled batch feeds from the payroll system to the employee change of status application. Changes made to this data in the employee change of status application may be updated in the payroll system through scheduled batch feeds from the employee change of status application back to the payroll system. Another way to ask this question is: What business events occur in other applications that the current application must know about and what business events occur in this application that other applications must know about?

1.1. Describe the current business processes that the primary application named in the template supports, how data is presently acquired, the timeline in the case that some of these existing integrations are being phased out, and the current flow of data among applications. This should be a high-level description in plain English prose not to exceed one page of text. Charts or diagrams are optional.

Description: The EnterpriseGreetingService sends Greeting.Create-Sync messages based on Greeting.Generate-Request messages sent by the GreetingApplication. This is the OpenEAI Project version of the classic Java “Hello, World!” programming example.

Timeline: Run on demand.

Flow: An input list is supplied to the GreetingApplication, the GreetingApplication sends Greeting.Generate-Request messages consumed by the EnterpriseGreetingService. The EnterpriseGreetingService then produces Greeting.Create-Sync messages.

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

1.2. List current application interfaces for the primary application named in the template which synchronize data changes made in other applications to the primary application.

[For each application, provide the name, description, source data structure, target data structure and how the source data is used to update the target. Use the structure below for the first application and repeat for other applications. If not applicable indicate below using **N/A.**]

List of interfaces: Input file greetees.txt

1.2.1. Source application details:

Source Application Name:	**GreetingApplication**			
Source Application Description:	From a supplied input list, produces Greeting.Generate-Request messages which are sent to the EnterpriseGreetingService			
Source Data Structure: (repeat for each table or filename related to this interface)	**Database Name:** N/A **Table Name:** N/A **File Name:** %CONNECTMAX_HOME%\configs\messaging\Environments\Examples\InputFiles\GreetingApplication\greetees.txt			
Source Field Name:	**Field type**	**Width**	**Nullable**	**Comments**
greetee.txt	N/A	Variable	N/A	This is a list of greetees, one per line

1.2.2. Target application details:

Target Application Name:	**EnterpriseGreetingService**			
Target Application Description:	The EnterpriseGreetingService sends Greeting.Create-Sync messages based on Greeting.Generate-Request messages sent by the GreetingApplication. This is the OpenEAI Project version of the classic Java “Hello, World!” programming example.			

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

1.3. List the interfaces that take data changes from the primary application named in the template to other existing applications.

N/A

[For each application, provide the name, description, source data structure, target data structure and how the source data is used to update the target. Use the structure below for the first application and repeat for other applications. If not applicable indicate above by **N/A.]**

List of interfaces: N/A

1.3.1. Source application details:

1.3.2. Target application details:

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 2: Describe Data Involved in Proposed Integrations

Describe and define the data that may be involved in interfaces with the application named in this template. This information will also be used to help define enterprise data objects for use in message when design teams meet with integration staff. **This section should be completed prior to integration design meetings with integration staff.**

2.1. Describe the proposed business processes that the application named in the template will support, how data will be acquired, the timeline and the proposed flow of data between applications. This should be a high-level description in plain English prose not to exceed one page of text. No charts or diagrams should be provided.

Description: The EnterpriseGreetingService sends Greeting.Create-Sync messages based on Greeting.Generate-Request messages sent by the GreetingApplication.

Timeline: Run on demand.

Flow: An input list is supplied to the GreetingApplication, the GreetingApplication sends Greeting.Generate-Request messages consumed by the EnterpriseGreetingService. The EnterpriseGreetingService then produces Greeting.Create-Sync messages and writes the greeting to standard output, for example:

OUTPUT>2006-05-09 14:11:27,875 INFO [Thread-30] - [ProcessInputLine.run] Greeting returned for 'Tod' is: Hello, Tod!

2.2. What data will this application require from other authoritative sources?

N/A

[Provide the name, description, authoritative source data structure, target data structure, and how the data will be used to update the application named in the template. Use the structure below for each authoritative source table and Target table and repeat as needed. If not applicable indicate above by **N/A.]**

2.2.1. Module details:

2.2.2. Target application details:

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

2.3. For what data will this application be the authoritative source?

[Provide the name, description, and how the inputs will used to update the application named in the template. Use the structure below for each source data structure. If not applicable indicate above by **N/A.]**

The Enterprise Greeting Service is authoritative for enterprise Greeting, that is, it produces Greeting synchronization messages (Greeting.Create-Sync).

2.3.1. Source application details:

Source Application Name:	**EnterpriseGreetingService**			
Source Application Description:	The EnterpriseGreetingService sends Greeting.Create-Sync messages based on Greeting.Generate-Request messages sent by the GreetingApplication. This is the OpenEAI Project version of the classic Java “Hello, World!” programming example.			

2.3.2. Target module details:

N/A

Target Application Name:	**GreetingApplication**			
Target Application Description:	The GreetingApplication sends requests to the GreetingService to generate Greetings for entities named in greetees.txt. The GreetingApplication consumes the greetings generated by the GreetingService.			

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 3: Describe the Flow of Data in the Proposed Integrations

Define (at a high level) the flow of messages between the application named in this template and other applications to support the interfaces defined in steps 1 and 2. **This section should be completed during the integration design phase with integration staff.**

3.1. Application 1: EnterpriseGreetingService

1. The EnterpriseGreetingService consumes Greeting.Generate-Request messages.
2. The EnterpriseGreetingService produces Greeting.Create-Sync messages.
3. The EnterpriseGreetingService produces Greeting.Response-Reply messages to report on the success or failure of Generate-Requests.

3.2. Application 2: GreetingApplication

1. The user inputs a list of the names of the entities to be greeted in greetees.txt. The GreetingApplication reads this file and produces Greeting.Generate-Request messages.
2. The GreetingApplication consumes the Greeting.Response-Reply messages produced by the EnterpriseGreetingService which report the success or failure of the Greeting.Generate-Request.

3.3. Application 3: N/A

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 4: List Existing and New Enterprise Data Objects Required for the Integrations

List existing enterprise data objects that will be reused and new enterprise data objects that will be defined to support application interfaces for the application named in this template. **This section should be completed during the integration design phase with integration staff.**

XML's Naming Convention:
? – Optional (0 or 1)
* – Optional (0 or more)
+ – At least 1 or more

Example:

```
<!ELEMENT ParentElement (ChildElementOne, ChildElementTwo?, ChildElementThree*, ChildElementFour+)>
<!ATTLIST ParentElement
        parentAttributeOne CDATA #REQUIRED
        parentAttributeTwo CDATA #IMPLIED
>
```

All attribute names begin with lowercase. All element names begin with uppercase. In this example, ChildElementOne is required, ChildElementTwo is either "null" or has a single value, ChildElementThree is either "null" or can have one or more values, and ChildElementFour cannot be null but must have one or more values. Attribute parentAttributeOne must be provided in the XML message, whereas parentAttributeTwo is optional.

Object Hierarchy Naming Convention:
The following naming conventions are equivalent:

ParentObject / ChildObject	ParentObject ChildObject	`ParentObject.ChildObject`
ParentObject @ parentAttribute	ParentObject parentAttribute	`ParentObject.parentAttribute`

[Existing enterprise data objects are found in the Segments.dtd files.] These are available as part of the integration documentation at:

[Example Enterprise Home Directory (ESB_HOME2, CONNECTMAX_HOME, etc.)]\message\releases\org\openeai\Resources

4.1 Existing enterprise data objects that will be reused or modified. This section should include the current definition of the enterprise objects as they appear in the enterprise message repository at the time the analysis was performed and any proposed changes.

[List the enterprise objects to be reused or state 'None' if no objects are available.]

None.

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

4.2 New enterprise data objects proposed. This section should include the proposed name and structure of the new data objects.

File	Name	Proposed Definition
Segments	Greetee	<!ELEMENT Greetee (FullName)>
Segments	Greeting	<!ELEMENT Greeting (Text)>

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 5: Name the Messages that Will Be Used to Implement the Integrations

Name the existing and proposed messages that use the enterprise data objects to implement the integration flows. **This section should be completed during the integration design phase with integration staff.**

5.1. Existing Messages

[List existing messages to be used in this interface or indicate that none are available for use.]

None.

5.2. Proposed Messages

[List proposed messages to be used in this interface or indicate that none are proposed for this interface.]

1. org.any-openeai-enterprise.CoreApplication.Greeting.Generate-Request
2. org.any-openeai-enterprise.CoreApplication.Greeting.Response-Reply
3. org.any-openeai-enterprise.CoreApplication.Greeting.Create-Sync

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 6: Name the Existing and New Messaging Applications Required

Name the existing messaging applications that will be used or define the new messaging applications that must be implemented to produce, consume, transform, and route the messages listed in step 5. **This section should be completed during the integration design phase with integration staff.**

6.1. Existing Messaging Components

[List existing components to be used in this interface or indicate that none are available for use]

Name	Type	Status	Responsible Unit
None			

6.2. New Messaging Components

Name	Type	Status	Responsible Unit
EnterpriseGreetingService	service	implemented	OpenEAI Project
GreetingApplication	application	implemented	OpenEAI Project

6.3. Other interface Components

Name	Type	Status	Responsible Unit
Request Proxy	infrastructure	deployed; needs new proxy rules	OpenEAI Project
Sync Router	infrastructure	deployed; needs new routing criteria	OpenEAI Project

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 7: Provide Technical Stories for the Primary Application

For the messaging component(s) of the application named in this template, list the messages that component must produce and consume and provide detailed stories describing the prescribed production and consumption logic. The owner of the application who is also responsible for implementing message production and consumption should prepare the detailed stories. This section should be completed during the integration design phase with integration staff.

7.1. Messages

Synchronization messages to publish

1. org.any-openeai-enterprise.CoreApplication.Greeting.Create-Sync

Synchronization messages to subscribe to

None.

Request messages to handle

1. org.any-openeai-enterprise.CoreApplication.Greeting.Generate-Request
2. org.any-openeai-enterprise.CoreApplication.Greeting.Response-Reply

7.2. Message Production Logic

[Describe the message production logic, the XML format for each message and mapping to source data that the application or gateway must produce.]

The EnterpriseGreetingService produces Greeting.Response-Reply and Greeting.Create-Sync messages (data area: Greeting.Text) in response to Greeting.Generate-Request messages sent by the GreetingApplication.

1. **org.any-openeai-enterprise.CoreApplication.Greeting.Create-Sync (DTD | Sample message)**

   Data Area element values:
   Greeting element (Segments file)

Enterprise Object Element / Attribute	Source Database Table	Source Database Field and transformations
Greeting	N/A	None
Text	N/A	None

Implementation resources: Names of resources

Work estimate in hours: XX hours

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

2. **org.any-openeai-enterprise.CoreApplication.Greeting.Response-Reply**
   (DTD | Sample message)

   Data Area element values:
   Greeting element (Segments file)

Enterprise Object Element / Attribute	Source Database Table	Source Database Field and transformations
Greeting	N/A	None
Text	N/A	None

Implementation resources: Names of resources

Work estimate in hours: XX hours

7.3. Message Consumption Logic

[Describe the message consumption logic, the XML format for each message and mapping to target data that the application or gateway must consume.]

The EnterpriseGreetingService consumes Greeting.Generate-Request messages sent by the GreetingApplication and uses the names in the request message to populate the Greeting element of the Greeting.Response-Reply and Greeting.Create-Sync messages it produces.

1. **org.any-openeai-enterprise.CoreApplication.Greeting.Generate-Request**
   (DTD | Sample message)

   Data Area element values:
   Greetee element (Segments file)

Enterprise Object Element / Attribute	Source Database Table	Source Database Field and transformations
Greetee	N/A	None
FullName	N/A	None

Implementation resources: Names of resources

Work estimate in hours: XX hours

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 8: Provide Technical Stories for the other Applications Named in the Template

For each remaining messaging component listed in step 6, list the new messages that component must produce and consume and provide brief stories describing the prescribed production and consumption logic. This section should be completed during the integration design phase with integration staff.

8.1. Messaging Component 1: GreetingApplication

8.1.1. Messages

1. org.any-openeai-enterprise.CoreApplication.Greeting.Generate-Request

8.1.2. Brief Story

The GreetingApplication reads the an input file, greetees.txt, and produces Greeting.Generate-Request messages, one for each Greetee listed in the greetees.txt input file.

The greetees.txt file consists of one greetee per line. Below is an example **greetees.txt** file:

World
Tod
Steve
Troye
David
Jon
Bob
Mike
John
Ziggy

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 9: Summarize all Outstanding Questions, Issues, and Action Items

Record all questions, issues and action items during the analysis and design sessions. Status will be indicated only when it is “closed”.

#	Question / Issues /Actions	Who?	Resolution	Date Resolved	Status
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 10: Perform Work Estimation and Scheduling

Review the work estimates provided with each story and messaging application. Sequence implementation work in a detailed project plan using your preferred project management tools.

10.1 List New Messaging Applications to Develop and Summary Work Estimates

Messaging Application	Resources	Total Hours
(Example) Payroll System Gateway	1. Chris Analyst, analysis and test suite development (24 hours) 2. Suzy Developer, BasicPerson message support (12 hours) 3. Joe Developer, BasicEmployee message support (12 hours) 4. Q.A. Masters, Test Suite Review and Certification (24 hours)	72

10.2 List New Message Support to Add to Existing Messaging Applications and Summary Work Estimates

Messaging Application	Resources	Total Hours
(Example) Identity Service	1. Chris Analyst, analysis and test suite development (36 hours) 2. Suzy Developer, InstitutionalIdentity synchronization message support (20 hours) 3. Q.A. Masters, Test Suite Review and Certification (24 hours)	80

10.3 Detailed Work Plan

Location of detailed implementation project plan: [provide link to location here]

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 11: Functional Approval of Analysis, Scope, and Scheduling

Functional team members must give approval to indicate completion of the analysis and design. An e-mail stating approval may be copied and pasted here instead of a signature on paper.

While many may consider this a pedantic step, experience teaches that this level of explicit approval is often required to support a strict scope management and change control process.

Project Role	Team member name	Signature	Approval Date

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 12: Initiate Template Change Control Processes

Application name:	**Payroll System Gateway** (Example)
Requestor name(s):	XXXXXXXX
Priority:	XXXXXXXX (Low, Medium, High. Please give priority details in later section)
Change Request Number:	**1** (Starts at 1 and increments by 1 for each application change request)
Brief change request description: XXXXXXXX	

Process for change requests: The EAI template should be completed during the period allocated to do the analysis and design work. Any changes subsequent to approval and acceptance of the template should follow the change request process. The change requests will be submitted by the requesting team to the EAI project manager who will forward the request to management for approval.

All items in this template must be filled out completely otherwise it may delay processing. For items that are not relevant to the change being requested please indicate that by typing in "Not Applicable".

The change requests must be packaged each week and delivered to the EAI project manager by Thursday noon. The review will be generally completed by the following Tuesday.

12.1 New requirements information: Please fill out the story and the details for the new functionality or services the gateway or application must support. This section must also be completed if there are new tables or fields in the requirements with or without any new functionality or services being requested.

[Provide story here]

12.2 Reference section in the existing EAI template which is impacted by the new request:

[Provide template section numbers or messages that are impacted by the new requirements]

12.3 Impact on other known integrations:

[example: Same change required in IdentityService template]

12.4 Describe priority (LOW, MEDIUM, HIGH):

[Describe the reasons behind the priority assessment in story format. If this requires a faster turnaround describe why and what dependencies it may share with other tasks]

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

12.5 New source system table and fields, if applicable:

[List the table and fields]

12.6 New target system table and fields, if applicable:

[List the table and fields]

12.7 New transformation requirements, if applicable:

Source table or file:	XXXXXX		
Target table or file:	XXXXXX		
Source field	**Target field**	**Transformation and reformatting rules**	**Message object**
Example Last-Name-10	Last_Name	None	BasicPerson/Name/LastName

12.8 Message Object changes:

[Enumerate message object definition changes.]

12.9 Work impact:

[Describe the anticipated impact on the project plan and timeline.]

12.10 Approval:

Management should sign and send documents back to the EAI team and the requestor. Approval implies acceptance of any changes in timeline.

Approval Request date:	**mm-dd-yyyy**			
Project Role	**Name**	**Signature**	**Signature date**	**Status (Approved or Denied)**
Project director				
Project manager				
Project manager				
Project manager				

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 13: Provide Testing Specifications

13.1 Test Stories

For every messaging application and gateway listed in this template, provide a brief description of how it will be tested for functional, performance, and production readiness certification.

(Example) Payroll System Gateway

Functional testing of the payroll system gateway will be accomplished by preparing an OpenEAI test suite to test the gateway's ability to handle BasicPerson and BasicEmployee query, create, update, and delete request messages as well as its ability to publish the corresponding synchronization messages. The goal of this test suite is to cover all actions on both BasicPerson and BasicEmployee objects using an appropriately diverse set of data.

Performance testing of the payroll system gateway will be accomplished by preparing an OpenEAI test suite with multiple series that can be run concurrently from one or more deployments of the test suite application to simulate the peak anticipated production load of concurrent requests. This suite will be used to test a deployment of the payroll system gateway to determine if the performance of deployment will handle the anticipated peak load. Additionally performance tests of web applications that send request messages to this gateway will be conducted independently of the gateway testing.

A self-contained production readiness OpenEAI test suite will be prepared that creates, queries for, updates, and deletes several bogus users and employees using realistic data. This test suite will be executed to certify the deployment in both non-production and production environments.

13.2 Links to Testing Artifacts Listed in the Test Stories and Preparation Status

Testing Artifact Description	Location	Preparation Status
(Example) Payroll System Gateway Functional Test Suite	CVS Repository Project/TestSuites PayrollSystemGateway.TestSuite1.xml	100%
(Example) Payroll System Gateway Performance Test Suite	CVS Repository Project/TestSuites PayrollSystemGateway.TestSuite2.xml	70%

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

Step 14: Develop and Test Integrations

At this point, development tasks for each messaging application and service should be enumerated in the integration project plan or master project plan. Developers and quality assurance staff should be developing and testing applications and gateways and reporting status on their assigned tasks.

A summary of tasks and their status could be maintained here in the integration template, but it is strongly encouraged to maintain this information and refer to it in a master project plan.

Step 15: Prepare Implementation and Rollout Plan

A rollout plan differs from the overall project plan in that it provides clear, step by step instructions for deployment and production readiness testing activities. This plan should be completed prior to production implementation and executed in a non-production environment as a practice or “dress rehearsal” run.

Step 16: Implement in Production

At this stage, production implementation should be anti-climactic. Execute the rollout plan, which should include production readiness testing, and celebrate.

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

The GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

(or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However,

$Revision: 1.1 $
$Date: 2006/08/14 21:24:37 $
$Source: /cvs/repositories/openii2/openii2/clients/general/training/OpenEaiAnalysisAndDevelopment/work/06-EnterpriseGreetingServiceTemplate.rtf,v $

parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.