OpenEAI Methodology
Version 1.0

May, 2005

by

Tod Jackson (tod@openeai.org)
Rupa Majumdar (rmajumd@uillinois.edu)
Steve Wheat (steve@openeai.org)

Copyright © 2005 OpenEAI Software Foundation.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Section being the first section entitled "Introduction", with no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is included in the section entitled "Appendix 1: GNU Free Documentation License".
Contents

Introduction
Methodology Overview
  Perform Analysis
  Define Messages
  Generate Messages Objects
  Develop, Document, and Test Messaging Applications
  Update Enterprise Documentation Artifacts
  Deploy in Production
  Tracking Progress along the Way
Methodology Details: The OpenEAI Integration Template
  Step 1: Describe Existing Integrations
  Step 2: Describe the Data Involved in the Proposed Integrations
  Step 3: Describe the Flow of Data in the Proposed Integrations
  Step 4: List Existing and New Enterprise Data Objects Required for the Integrations
  Step 5: Name the Messages that Will Be Used to Implement the Integrations
  Step 6: Name the Existing and New Messaging Applications Required
  Step 7: Provide Technical Stories for the Primary Application
  Step 8: Provide Technical Stories for the other Applications Named in the Template
  Step 9: Summarize all Outstanding Questions, Issues, and Action Items
  Step 10: Perform Work Estimation and Scheduling
  Step 11: Functional Approval of Analysis, Scope, and Scheduling
  Step 12: Initiate Template Change Control Processes
  Step 13: Provide Testing Specifications
  Step 14: Develop and Test Integrations
  Step 15: Prepare Implementation and Rollout Plan
  Step 16: Implement in Production

Appendix 1: Completed Analysis Template Example – Institutional Identity Service
Appendix 2: Completed Analysis Template Example – SunGard SCT Banner
Appendix 3: The GNU Free Documentation License
Introduction

The purpose of the OpenEAI methodology is to bring order, discipline, and transparency to the practices of integration analysis, development, testing, and project management. While the OpenEAI methodology, analysis template, and related artifacts are tailored around some of the specifics of the OpenEAI concepts and technology, many aspects of this methodology add value to any integration project using any concepts and technology. The OpenEAI project encourages organizations to adapt the methodology and artifacts to their practices and technologies and appreciates feedback and suggestions to make the OpenEAI methodology more generally applicable and useful in a wide range of applications.

Note that a number of different names can be used to refer to the defined discrete units of business data used to conduct enterprise messaging transactions: Enterprise Data Objects, Enterprise Message Objects, Enterprise Business Objects, and so on. In the introduction and overview, we'll simply use the term “enterprise objects” to refer to these business data objects.
Methodology Overview

Perform Analysis

Identify, in general, the systems that need to be integrated. Once those are identified, an analysis group comprised of business or functional experts and technical integration analysts complete the OpenEAI analysis template, or their organization’s customized version of the template, for each application that is to be integrated. Among other things, the template:

- documents general integration requirements
- identifies which new enterprise objects will be required
- specifies which existing enterprise objects will be used
- defines the structure of new enterprise objects
- specifies which message actions of the OpenEAI Message Protocol will be required for each enterprise object
- enumerates the messaging applications, gateways, and infrastructure that must be developed to implement the integrations, and
- enumerates detailed message production and consumption logic

Define Messages

Define any new enterprise objects needed and the message actions they require through the process of completing the OpenEAI analysis template. Technical integration analysts can then create XML message definitions for each of the new messages and their message actions and slot them into the organization’s message hierarchy, and provide a sample message for each definition. This process allows organizations to share message definitions and communicate about them effectively internally and with trading partners. For more background on defining and sharing OpenEAI message definitions and enterprise object documents, see the OpenEAI Message Definition Document.

Generate Message Objects and Related Artifacts

Next, the message definitions are implemented as Java objects. A Java object must be created for every complex enterprise object defined. These Java objects are automatically generated using the OpenEAI MoaGenApplication, which also generates their corresponding enterprise object XML documents. This step is typically performed by EAI analysts, who should be well-suited to complete the enterprise objects documents with the appropriate rules based on the outcome of their EAI analysis. For more information on the Java message object API (MOA), see the Java message object section of the OpenEAI API Introduction Document.

Develop, Document, and Test Messaging Applications

The details of this phase will vary from organization to organization: many organizations already have application development and testing practices established. However an organization chooses to implement them, the following guidelines should be considered.
1. Developers and analysts prepare detailed, technical stories for each messaging application and gateway listed in the completed analysis. These stories will draw heavily on the message production and consumption logic prepared by the subject matter experts and analysts included in the analysis template.

2. Developers implement the appropriate messaging applications and gateways listed in the template using OpenEAI foundation components, the message object API that was generated for the organization’s enterprise objects, and the enterprise object documents completed by the subject matter experts and analysts. When developing an OpenEAI-based application or gateway, this means developing the commands needed to support the processes defined in the analysis.

3. While steps one and two above are proceeding, integration analysis staff can prepare OpenEAI TestSuiteApplication test suite documents for testing the message gateways that are to be developed. Test suite documents are XML documents made up of messaging test cases. The OpenEAI TestSuiteApplication can take a test suite document and execute all its test cases very rapidly: it sends test messages to a target application and compares the replies received and sync messages published by the target application with expected results, produces a detailed report, and can even tear-down any messaging artifacts or database entries created as a result of the TestSuite execution. Subject matter experts and analysts are typically the best equipped to prepare these test suite documents, because they are the people who know the business logic and they also specified the integration requirements, message production logic, and message consumption logic. Developers can perform this work, but if they do, the subject matter experts and analysts should review the test cases and certify they represent the requirements.

   It is useful for developers to have these test cases available to them during the development process. As they implement message support they can iteratively execute the test suite using the OpenEAI TestSuiteApplication to check their progress. As developers make changes they can use the test suites to verify they do not break any required behavior.

   At this time, subject matter experts and analysts also prepare real-world online and batch scenarios to test the new messaging applications and gateways in combination with the rest of the messaging enterprise in a test environment.

4. All messaging applications and gateways pass both informal developer testing and all of the formal test suites executed using the TestSuiteApplication.

5. The new messaging applications and gateways are promoted from a development environment to a test environment for integration testing, and the real-world online and batch scenarios are executed until the subject matter experts and analysts are convinced the new applications are performing appropriately. Note that although the practice of preparing test suites and unit testing messaging applications can be very effective at ensuring that messaging applications perform as designed, it may not help validate that the design is correct. From time to time integration testing with the rest of the messaging enterprise turns up new requirements, such as identifying additional applications that need to know about data from a new authoritative source. For this reason, integration testing is a critical part of the process. It is also the part of the testing process that gives management a tangible level of confidence that it is appropriate to proceed with a production implementation.
Update Enterprise Documentation Artifacts

Practicing the OpenEAI methodology produces a number of documentation artifacts, including an analysis template for each application that interfaces with other applications, enterprise object definitions, message definitions, and javadoc for applications that implement support for each enterprise object. These artifacts should be posted in a web-accessible format. For example, message definitions and enterprise object documents should be posted on a web server so messages can be validated when necessary and so field-level business rules and translations can be applied by the Java implementation of the message objects at runtime. This practice allows you to build a web page to nicely document each messaging application or gateway, linking to and leveraging each of these artifacts that must be created anyway to support XML document validation and enterprise object validation. For more information on posting message object definitions, enterprise object documents, and a description of the application in an enterprise, and which enterprise objects these are authoritative for, see the OpenEAI Message Definition Document.

This document recommends a format for these web pages in the documentation section below. One real-world example of such documentation is available on the “Get Enterprise Data” page of the University of Illinois’ Administrative IT web site at http://www.ait.uillinois.edu/GetEnterpriseData.

Just as your organization may customize the analysis template or the OpenEAI methodology itself, you will likely choose to customize the web documentation template as well. This web documentation can be posted on your organizational web site or intranet. Posting this documentation helps managers, functional analysts, and technical analysts plan and prepare for new integrations. Additionally, many organizations have auditing or best-practice requirements that mandate the preparation of some type of formal documentation for each integration. Posting this documentation as a web page makes this information available to those who need it. In some cases, it can even be helpful to complete and post most of the template prior to or during the development phase.

Deploy in Production

There’s not much to say about this step from an overview perspective, since if you get to this point, most of the work has already been done. If you follow the recommended OpenEAI practices for testing in pre-production environments, deploying in production should be anticlimactic. The OpenEAI Deployment Patterns document provides details on the minimum number of recommended environments you should set up for a messaging enterprise and how and when to promote messaging applications and gateways from one environment to the next.

Tracking Progress along the Way

Two major risk categories inherent in integration projects are “real” and “perceived” failure. There are many ways to fail when integrating complex systems. The best way to avoid failure, or even the perception of failure, is to track progress and manage changing requirements, expectations, and timelines. The single most important benefit of a structured integration analysis template is that it breaks work into discrete tasks for which status can be recorded in the template and tracked by integration project managers in their overarching project plans. The OpenEAI integration analysis template supports this tracking and reporting by encouraging integrators and
analysts to identify resources, estimate work, and set goals for every step along the way. These
goals can be noted and tracked by project managers who can help escalate deficient resources
or accelerate lagging progress before they become blocking issues or high risks for the
integration project or the larger project under which this integration work is occurring.
Methodology Details: The OpenEAI Integration Template

Note: a standalone version of this template, separate and distinct from the Methodology Document, is available on the OpenEAI Project documentation page at:

http://www.openeai.org/openeai.xml?document=Documentation.xml

This standalone version can be a convenient starting point for customizing your organization’s own version of an integration process.

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.

<table>
<thead>
<tr>
<th>Application Name:</th>
<th>Xxxxxxxxxxxxxx</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module Name (if appropriate):</td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>Phase (if appropriate):</td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>Integration Timeframe:</td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>Target Template Completion Date:</td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>Template Owner Name:</td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>Total Work Estimate:</td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>Resources Named in Template:</td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>Location of Detailed Project Plan:</td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>Template Change Reviewers:</td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>Xxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>Template Status:</td>
<td></td>
</tr>
<tr>
<td>Step 1    ☐                        Step 2    ☐                        Step 3    ☐                        Step 4    ☐</td>
<td></td>
</tr>
<tr>
<td>Step 5    ☐                        Step 6    ☐                        Step 7    ☐                        Step 8    ☐</td>
<td></td>
</tr>
<tr>
<td>Step 9    ☐                        Step 10   ☐                        Step 11   ☐                        Step 12   ☐</td>
<td></td>
</tr>
<tr>
<td>Step 13   ☐                        Step 14   ☐                        Step 15   ☐                        Step 16   ☐</td>
<td></td>
</tr>
</tbody>
</table>

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, 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.

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.

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.
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 usually acquired by the application 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 is kept synchronized with the payroll system by scheduled batch feeds from the payroll system to these applications. Changes made to this data in these applications are updated in the payroll system through scheduled batch feeds from these applications 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 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: XXXXXXXXXXX

Timeline: XXXXXXXXXXX

Flow: XXXXXXXXXXX
1.2. List current application interfaces for the primary application named in the template that synchronizes data changes made in other applications to the primary application named in this template.

[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: XXXXXXXX, XXXXXXXX

1.2.1. Source application details:

<table>
<thead>
<tr>
<th>Source Application Name:</th>
<th>XXXXXXXX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Application Description:</td>
<td>XXXX XXXX (Brief description)</td>
</tr>
<tr>
<td>Source Data Structure: (repeat for each table or filename related to this interface)</td>
<td>Database Name: XXXXXXXX</td>
</tr>
<tr>
<td></td>
<td>Table Name: XXXXXXXX</td>
</tr>
<tr>
<td></td>
<td>File Name: XXXXXXXX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example TAPPOINTMENT.APPT_REF_NB</td>
<td>CHAR</td>
<td>1</td>
<td>NO</td>
<td>The appointment reference number in the employee change of status application</td>
</tr>
</tbody>
</table>

1.2.2. Target application details:

<table>
<thead>
<tr>
<th>Target Application Name:</th>
<th>This is the application named in the template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Application Description:</td>
<td>Description already given for the Application named in the template.</td>
</tr>
<tr>
<td>Target Data Structure: (repeat for each table or filename related to this interface)</td>
<td>Database Name: XXXXXXXX</td>
</tr>
<tr>
<td></td>
<td>Table Name: XXXXXXXX</td>
</tr>
<tr>
<td></td>
<td>File Name: XXXXXXXX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example APPOINTMENT-REF-NUM-60</td>
<td>VARCHAR</td>
<td>60</td>
<td>NO</td>
<td>The appointment reference number in the payroll system</td>
</tr>
</tbody>
</table>
1.3. List the interfaces that take data changes from the primary application named in the template to other existing applications.

[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: XXXXXXXX, XXXXXXXX

1.3.1. Source application details:

<table>
<thead>
<tr>
<th>Source Application Name:</th>
<th>This is the application named in the template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Application Description:</td>
<td>Description already given for the Application named in the template.</td>
</tr>
</tbody>
</table>

Source Data Structure: (repeat for each table or filename related to this interface)

<table>
<thead>
<tr>
<th>Database Name:</th>
<th>XXXXXXXX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table Name:</td>
<td>XXXXXXXX</td>
</tr>
<tr>
<td>File Name:</td>
<td>XXXXXXXX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example LAST-NAME-10</td>
<td>CHAR</td>
<td>23</td>
<td>NO</td>
<td>The last name of a person in the payroll system</td>
</tr>
</tbody>
</table>

1.3.2. Target application details:

<table>
<thead>
<tr>
<th>Target Application Name:</th>
<th>XXXXXXXX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Application Description:</td>
<td>XXXX XXXX (Brief description)</td>
</tr>
</tbody>
</table>

Target Data Structure: (repeat for each table or filename related to this interface)

<table>
<thead>
<tr>
<th>Database Name:</th>
<th>XXXXXXXX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table Name:</td>
<td>XXXXXXXX</td>
</tr>
<tr>
<td>File Name:</td>
<td>XXXXXXXX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example TPERSON.LAST_NAME</td>
<td>VARCHAR</td>
<td>30</td>
<td>NO</td>
<td>The last name of a person in the employee change of status system</td>
</tr>
</tbody>
</table>
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: XXXXXXXXX

Timeline: XXXXXXXX

Flow: XXXXXXXX

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

[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:

<table>
<thead>
<tr>
<th>Source Application Name:</th>
<th>(name of the other authoritative system)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Application Description:</td>
<td>(Briefly describe the authoritative source module or business function involved in this interface.)</td>
</tr>
<tr>
<td>Source Data Structure:</td>
<td>Database Name: XXXXX</td>
</tr>
<tr>
<td>(repeat for each table or filename related to this interface)</td>
<td>Table Name: XXXXXXXX</td>
</tr>
<tr>
<td>Source Field Name</td>
<td>Field type</td>
</tr>
<tr>
<td>Example GENDER</td>
<td>VARCHAR</td>
</tr>
</tbody>
</table>

2.2.2. Target application details:

<table>
<thead>
<tr>
<th>Target Application Name:</th>
<th>This is the application named in the template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Application Description:</td>
<td>Description already given for the application named in the template</td>
</tr>
</tbody>
</table>
### Target Data Structure:

- **Database Name**: XXXXXXXX
- **Table Name**: XXXXXXXX
- **File Name**: XXXXXXXX

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example LEGACY_GENDER</td>
<td>CHAR</td>
<td>1</td>
<td>NO</td>
<td>Gender code stored by the legacy system</td>
</tr>
</tbody>
</table>

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

[Provide the name, description, and how the inputs will be 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.]

#### 2.3.1. Source application details:

<table>
<thead>
<tr>
<th>Source Application Name:</th>
<th>(This is the application named in the template)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Application Description:</td>
<td>(Description already given for the Application named in the template.)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Data Structure:</th>
<th>(repeat for each table or filename related to this interface)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Name:</td>
<td>XXXXXXXX</td>
</tr>
<tr>
<td>Table Name:</td>
<td>XXXXXXXX</td>
</tr>
<tr>
<td>File Name:</td>
<td>XXXXXXXX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example LAST-NAME-10</td>
<td>CHAR</td>
<td>23</td>
<td>NO</td>
<td>The last name of a person in the legacy system</td>
</tr>
</tbody>
</table>

#### 2.3.2. Target module details:

<table>
<thead>
<tr>
<th>Target Application Name:</th>
<th>Other system that requires data from our legacy system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Application Description:</td>
<td>(Briefly describe the module involved in this interface.)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Target Data Structure:</th>
<th>(repeat for each table or filename related to this interface)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Name:</td>
<td>XXXX</td>
</tr>
<tr>
<td>Table Name:</td>
<td>XXXXXXXX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example LAST_NAME</td>
<td>VARCHAR</td>
<td>60</td>
<td>NO</td>
<td>The last name of a person in the target system</td>
</tr>
</tbody>
</table>
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: Payroll System (Example)

1. As basic person and basic employee information changes the Payroll System, the Payroll System publishes synchronization messages to keep the ERP and Identity Service up-to-date.
2. ...

3.2. Application 2: Identity Service (Example)

1. The Identity Service must handle incoming ID number query and generate requests and send the appropriate replies.
2. The Identity Service must handle incoming basic person and basic employee synchronization messages and apply any changes to the Identity Service database to keep its data current.
3. ...

3.3. Application 3: ERP (Example)

1. The ERP must consume basic person and basic employee synchronization messages from the legacy Payroll system to keep its data current.
2. ...
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:

```xml
<!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:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ParentObject @ parentAttribute</td>
<td>ParentObject parentAttribute</td>
<td>ParentObject.parentAttribute</td>
</tr>
</tbody>
</table>

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

[location to the Segment file(s)]

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.]

1. LightweightPerson (Example)
2. UnknownPerson (Example)

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

<table>
<thead>
<tr>
<th>File</th>
<th>Name</th>
<th>Proposed Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segments</td>
<td>InstitutionalIdentity</td>
<td><code>&lt;!ELEMENT InstitutionalIdentity (InstitutionalId, UnknownPerson)&gt;</code></td>
</tr>
</tbody>
</table>
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.]


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

1. com.sct.Person.BasicPerson.Create-Sync (Example)
2. com.sct.Person.BasicPerson.Delete-Sync (Example)
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]

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Status</th>
<th>Responsible Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Example) Identity Service</td>
<td>Gateway</td>
<td>Implemented</td>
<td>Unit responsible for supporting this service (both the data and the implementation)</td>
</tr>
</tbody>
</table>

### 6.2. New Messaging Components

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Status</th>
<th>Responsible Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Example) Payroll system</td>
<td>gateway</td>
<td>planned</td>
<td>Unit responsible for supporting this service (both data and the implementation)</td>
</tr>
<tr>
<td>(Example) ERP Gateway</td>
<td>gateway</td>
<td>In development</td>
<td></td>
</tr>
</tbody>
</table>

### 6.3. Other interface Components

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Status</th>
<th>Responsible Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Example) Payroll extract</td>
<td>application</td>
<td>planned</td>
<td></td>
</tr>
</tbody>
</table>
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

Messages to produce

Messages to consume

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.]

(Example) The legacy payroll system gateway must send InstitutionalIdentity.Query-Request and InstitutionalIdentity.Generate-Request messages to determine if each new employee is already known by the identity service and to generate a new identity for each new employee who is not already known by the identity service. This gateway must also publish BasicPerson and BasicEmployee synchronization messages as person and employee information changes. This process will occur whenever the legacy payroll system gateway is provided with new payroll data. Presently, new payroll files are produced nightly.


Data Area Element values:
(Example) UnknownPerson element (provide Segments link)
The UnknownPerson/SocialSecurityNumber element value is taken from the Payroll SSN-10 field.

<table>
<thead>
<tr>
<th>Enterprise Object Element /</th>
<th>Source Database Table</th>
<th>Source Database Field and transformations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>Example Paymaster Table</td>
<td>Example Paymaster Field</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UnknownPerson</th>
<th>Paymaster</th>
<th>SSN-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SocialSecurityNumber</td>
<td>Paymaster</td>
<td>This must be numeric</td>
</tr>
<tr>
<td>Name/FirstName</td>
<td>Paymaster</td>
<td>FIRST-NAME-10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Convert to mixed case</td>
</tr>
</tbody>
</table>
Implementation resources:

Work estimate in hours:

2. org.any-openeai-enterprise.CoreApplication.InstitutionalIdentity-Generate-Request
   (provide DTD link| Sample message link)

Data Area Element values:

<table>
<thead>
<tr>
<th>Enterprise Object Element / Attribute</th>
<th>Source Database Table</th>
<th>Source Database Field and transformations</th>
</tr>
</thead>
</table>

Implementation resources:

Work estimate in 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.]

(Example) The Payroll gateway will consume messages in response to each of the messages it produces and sends to other messaging components. The messages it must consume are:

3. org.any-openeai-enterprise.CoreApplication.InstitutionalIdentity-Create-Sync

Data Area Element values:

<table>
<thead>
<tr>
<th>Enterprise Object Element / Attribute</th>
<th>Target Database Table</th>
<th>Target Database Field and transformations</th>
</tr>
</thead>
</table>

Implementation resources:

Work estimate in hours:


Data Area Element values:
<table>
<thead>
<tr>
<th>Enterprise Object Element / Attribute</th>
<th>Target Database Table</th>
<th>Target Database Field and transformations</th>
</tr>
</thead>
</table>

Implementation resources:

Work estimate in hours
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: ERP Gateway (Example)

8.1.1. Messages

1. com.sct.Person.BasicPerson.Create-Sync (Example)
2. com.sct.Person.BasicPerson.Delete-Sync (Example)

8.1.2. Brief Story

(Example) The ERP Gateway must consume the BasicPerson and BasicEmployee synchronization messages.

8.2. Messaging Component 2: Identity Service (Example)

8.2.1. Messages

1. org.any-openeai-enterprise.CoreApplication.InstitutionalIdentity-Query-Request (Example)
2. org.any-openeai-enterprise.CoreApplication.InstitutionalIdentity-Generate-Request (Example)
3. org.any-openeai-enterprise.CoreApplication.InstitutionalIdentity-Provide-Reply (Example)

8.2.2 Brief Story

(Example) The Identity Service must field incoming InstitutionalIdentity query request and provide replies. It must also …..
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".

<table>
<thead>
<tr>
<th>#</th>
<th>Question / Issues /Actions</th>
<th>Who?</th>
<th>Resolution</th>
<th>Date Resolved</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Step 10: Perform Work Estimation and Scheduling

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

10.1 List New Messaging Applications to Develop and Summary Work Estimates

<table>
<thead>
<tr>
<th>Messaging Application</th>
<th>Resources</th>
<th>Total Hours</th>
</tr>
</thead>
</table>
| **(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

<table>
<thead>
<tr>
<th>Messaging Application</th>
<th>Resources</th>
<th>Total Hours</th>
</tr>
</thead>
</table>
| **(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]
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.

<table>
<thead>
<tr>
<th>Project Role</th>
<th>Team member name</th>
<th>Signature</th>
<th>Approval Date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Step 12: Initiate Template Change Control Processes

<table>
<thead>
<tr>
<th>Application name:</th>
<th>Payroll System Gateway (example)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requestor name(s):</td>
<td>xxxx xxxxxxx</td>
</tr>
<tr>
<td>Priority:</td>
<td>XXXX (Low, Medium, High. Please give details in later section for priority)</td>
</tr>
<tr>
<td>Change Request Number:</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>(starts at 1 and increments by 1 for each application change request)</td>
</tr>
<tr>
<td>Brief change request description:</td>
<td>xxxxxxxxxx</td>
</tr>
</tbody>
</table>

**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]
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:

<table>
<thead>
<tr>
<th>Source table or file:</th>
<th>XXXXXX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target table or file:</td>
<td>XXXXXX</td>
</tr>
<tr>
<td>Source field</td>
<td>Target field</td>
</tr>
<tr>
<td>Example Last-Name-10</td>
<td>Last_Name</td>
</tr>
</tbody>
</table>

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 will sign and send documents back to the EAI team and the requestor. Approval implies acceptance of the change in timeline.

<table>
<thead>
<tr>
<th>Approval Request date:</th>
<th>mm-dd-yyyy</th>
<th>Project Role</th>
<th>Name</th>
<th>Signature</th>
<th>Signature date</th>
<th>Status (Approved or Denied)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Project director</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Project manager</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Project manager</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Project manager</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Testing Artifact Description</th>
<th>Location</th>
<th>Preparation Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Example) Payroll System Gateway Functional Test Suite</td>
<td>CVS Repository Project/TestSuites PayrollSystemGateway.TestSuite1.xml</td>
<td>100%</td>
</tr>
<tr>
<td>(Example) Payroll System Gateway Performance Test Suite</td>
<td>CVS Repository Project/TestSuites PayrollSystemGateway.TestSuite2.xml</td>
<td>70%</td>
</tr>
</tbody>
</table>
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.
Appendix 1: Completed Analysis Template Example – Institutional Identity Service

This completed template is an example of an organization’s use of the OpenEAI template to perform the analysis for an identity service. The template was customized and curtailed for the organization’s needs. This completed template was given to the OpenEAI project by the University of Illinois.

Application Name: ICard
Module Name: ICard
Phase: 1
Timeframe: December 2001—June 2002
Target template completion date: 07-30-2001
Template Owner Name: Jeff Fletcher
Template Status: Step1 ☑ Step2 ☑ Step3 ☑ Step4 ☑ Step5 ☑ Step6 ☑ Step7 ☑ Step8 ☑

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 AITS staff.**

Step 1. 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 AITS integration staff. This section should be completed prior to integration design meetings with AITS integration staff.

Step 2. Describe and define Banner 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 messages when design teams meet with AITS integration staff. This section should be completed prior to integration design meetings with AITS integration staff.

Step 3. 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 AITS integration staff.

Step 4. 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 AITS integration staff.

Step 5. 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 AITS integration staff.

Step 6. Name the existing messaging components that will be used or define the new messaging components 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 AITS integration staff.
Step 7. For the messaging component(s) specifically for the application named in this template, list the new 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 AITS integration staff.

Step 8. 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 AITS integration staff.**

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

1. **Step 1**
   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 AITS integration staff. This section should be completed prior to integration design meetings with AITS integration staff.

What data does this application store and operate on that it does not create itself? Presently, this data is usually acquired by the application through batch extracts and feeds, remote procedure calls, or data replication. For example, applications such as Payroll History Database (PHD) and ECOS maintain and in some cases update appointment data. This data is kept synchronized with Paymaster by scheduled batch feeds from Paymaster to these applications. Changes made to this data in these applications are updated in Paymaster through scheduled batch feeds from these applications to Paymaster. Another way to look at 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 application named in the template supports, how data is presently acquired, the timeline, and the current 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:** ICard is the University's identification card and ID number generation and tracking system for all University employees, students, and other entities like vendors, student spouses, and persons and non-persons that have university affiliations. Data for students is acquired when admitted applicants register to become students at the University through feeds from student systems. Data for employees is acquired through a feed from the Paymaster system. Data for other entities is acquired through special interfaces like Intensive English Institute feed, University foundation, McKinley Health Services feed.

In the future, the ICard system will continue to provide unique identification numbers for University employees, students and other entities like vendors, student spouses, and persons and non-persons that have university affiliations; at first through interfaces with existing systems which record new employees and new students, and later through a direct interface with Banner which will begin recording new students in June 2002 and new employees in October 2003.

**Timeline:** The Paymaster data changes must continue to be propagated to ICard using current batch processes until December 2001. Starting December 2001 Paymaster changes must begin to be reflected in ICard using the messaging technology. In June 2002 when Banner admissions module is implemented and new applicants are added to Banner directly, there may be a new
interface between Banner and ICard to generate the ids for admitted students. The Paymaster interface will continue until December 2003 when Paymaster is decommissioned and Banner completely assumes support of all HR and payroll functions at which point there may be a new interface with ICard.

Flow: Basic person and employee information from Paymaster flows to ICard. ICard presently maintains its own copy of person and employee information that is in paymaster for ICard’s own purposes. For example, ICard client applications have interfaces that look up address and phone number information. Presently, data does not flow back from ICard to Paymaster. ICard receives similar feeds from campus student systems. Similarly, ICard maintains an image of that data as it appears in those systems.
1.2. List the current application interfaces for the application named in the template that synchronizes data changes made in other applications.

[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.]

Only information for the current Paymaster to ICard interface is provided here, since the feeds from campus students systems, Intensive English Institute, University Foundation, and McKinley Health Services will continue until those systems are modified or retired.

1.2.1. Source application details:

<table>
<thead>
<tr>
<th>Source Application Name:</th>
<th>Paymaster (This is the application named in the template)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Application Description:</td>
<td>The Paymaster system provides the essential HR and payroll functions for the University.</td>
</tr>
<tr>
<td>Source Data Structure: (repeat for each table or filename related to this interface)</td>
<td>Database Name: XXXXXXXX</td>
</tr>
<tr>
<td></td>
<td>Table Name: XXXXXXXX</td>
</tr>
<tr>
<td></td>
<td>File Name: PAYROLL_DAT</td>
</tr>
<tr>
<td>Source Field Name</td>
<td>Field type</td>
</tr>
<tr>
<td>Example LAST-NAME-10</td>
<td>CHAR</td>
</tr>
</tbody>
</table>
1.2.2. Target application details:

<table>
<thead>
<tr>
<th>Target Application Name:</th>
<th>ICARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Application Description:</td>
<td>Description already given for the Application named in the template.</td>
</tr>
<tr>
<td>Target Data Structure: (repeat for each table or filename related to this interface)</td>
<td>Database Name: ICARD</td>
</tr>
<tr>
<td></td>
<td>Table Name: DEMOGRAPH</td>
</tr>
</tbody>
</table>
### Target Field Name | Source Field Name | Transformation, formatting, and comments
---|---|---
SOURCE | | |
EXT_ID_TYP* | | |
EXTERN_ID* | | |
CAMPUS* | | |
NAME | | |
LAST_NAME | | |
FIRST_NAME | | |
MID_NAME | | |
PREFIX_NAME | | |
SUFFIX_NAME | | |
SEX | | |
BIRTHDATE | | |
TIME_STAMP | | |
REC_UPDATE | | |

### Target Field Name | Field type | Width | Nullable | Comments
---|---|---|---|---
SOURCE | Char(10) | 10 | NO | |
EXT_ID_TYP* | Char(10) | 10 | NO | |
EXTERN_ID* | Char(16) | 16 | NO | |
CAMPUS* | Char(1) | 1 | NO | |
NAME | Varchar(31) | 31 | NO | |
LAST_NAME | Varchar(30) | 30 | NO | |
FIRST_NAME | Varchar(30) | 30 | NO | |
MID_NAME | Varchar(35) | 35 | NO | |
PREFIX_NAME | Varchar(10) | 10 | NO | |
SUFFIX_NAME | Varchar(10) | 10 | NO | |
SEX | Char(1) | 1 | NO | |
BIRTHDATE | Char(8) | 8 | NO | | YYYYMMDD
TIME_STAMP | Timestamp | | NO | |
REC_UPDATE | Datetime | | NO | |

### Target Field Name | Field type | Width | Nullable | Comments
---|---|---|---|---
SOURCE* | Char(10) | 10 | NO | |
EXT_ID_TYP* | Char(10) | 10 | NO | |
EXTERN_ID* | Char(16) | 16 | NO | |
CAMPUS* | Char(1) | 1 | NO | |
ADDR_TYPE* | Char(10) | 10 | NO | | Home, Local, Work
ADDRESS_1 | Varchar(30) | 30 | NO | |
ADDRESS_2 | Varchar(30) | 30 | NO | |
ADDRESS_3 | Varchar(30) | 30 | NO | |
CITY | Varchar(20) | 20 | NO | |
STATE | Char(3) | 3 | NO | |
ZIP_CODE | Varchar(10) | 10 | NO | |
MAIL_CODE | Varchar(6) | 6 | NO | |
COUNTRY | Char(3) | 3 | NO | |
PHONE | Varchar(25) | 25 | NO | |
SUPP_ADDR | Char(1) | 1 | NO | |
SUPP_PHONE | Char(1) | 1 | NO | |
TIME_STAMP | Timestamp | | NO | |
REC_UPDATE | Datetime | | NO | |

### Target Field Name | Field type | Width | Nullable | Comments
---|---|---|---|---
SOURCE | Char(10) | 10 | NO | |
EXT_ID_TYP* | Char(10) | 10 | NO | |
EXTERN_ID* | Char(16) | 16 | NO | |
CAMPUS* | Char(1) | 1 | NO | |
HOME_DEPT | Char(4) | 4 | NO | |
EMP_GROUP | Char(1) | 1 | NO | |
EMP_STATUS | Char(1) | 1 | NO | |
APPT_TYPE | Char(1) | 1 | NO | |
TIME_STAMP | Timestamp | | NO | |
REC_UPDATE | Datetime | | NO | |

### 1.2.3. How are the source inputs used to update the target application?
<table>
<thead>
<tr>
<th>Table</th>
<th>business rules for target</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMOGRAPH</td>
<td></td>
</tr>
<tr>
<td><strong>SOURCE</strong></td>
<td>Value “PAYROLL”</td>
</tr>
<tr>
<td>EXT_ID_TYP*</td>
<td></td>
</tr>
<tr>
<td>EXTERN_ID*</td>
<td>EXTERN_ID (SSN)</td>
</tr>
<tr>
<td>CAMPUS*</td>
<td></td>
</tr>
<tr>
<td>NAME</td>
<td></td>
</tr>
<tr>
<td>LAST_NAME</td>
<td>DEMOGRAPH.LAST_NAME</td>
</tr>
<tr>
<td>FIRST_NAME</td>
<td>DEMOGRAPH.FRST_NAME</td>
</tr>
<tr>
<td>MID_NAME</td>
<td>DEMOGRAPH.MID_NAME</td>
</tr>
<tr>
<td>PREFERENCES</td>
<td>DEMOGRAPH.SEX</td>
</tr>
<tr>
<td>SUFFIX_NAME</td>
<td>DEMOGRAPH.BIRTHDATE</td>
</tr>
<tr>
<td>SEX</td>
<td></td>
</tr>
<tr>
<td>BIRTHDATE</td>
<td></td>
</tr>
<tr>
<td>TIME_STAMP</td>
<td></td>
</tr>
<tr>
<td>REC_UPDATE</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>Value “PAYROLL”</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SOURCE</strong></td>
<td></td>
</tr>
<tr>
<td>EXT_ID_TYP*</td>
<td></td>
</tr>
<tr>
<td>EXTERN_ID*</td>
<td>EXTERN_ID (SSN)</td>
</tr>
<tr>
<td>CAMPUS*</td>
<td></td>
</tr>
<tr>
<td>ADDR_TYPE*</td>
<td>ADDRESS.ADDRESS_1</td>
</tr>
<tr>
<td>ADDRESS_1</td>
<td>ADDRESS.ADDRESS_2</td>
</tr>
<tr>
<td>ADDRESS_2</td>
<td>ADDRESS.CITY ADDRESS.STATE</td>
</tr>
<tr>
<td>ADDRESS_3</td>
<td>ADDRESS.ZIP_CODE</td>
</tr>
<tr>
<td>CITY</td>
<td>ADDRESS.MAIL_CODE</td>
</tr>
<tr>
<td>STATE</td>
<td>ADDRESS.PHONE</td>
</tr>
<tr>
<td>ZIP_CODE</td>
<td>ADDRESS.SUPP_ADDR</td>
</tr>
<tr>
<td>MAIL_CODE</td>
<td>ADDRESS.SUPP_PHONE</td>
</tr>
<tr>
<td>COUNTRY</td>
<td></td>
</tr>
<tr>
<td>PHONE</td>
<td></td>
</tr>
<tr>
<td>SUPP_ADDR</td>
<td></td>
</tr>
<tr>
<td>SUPP_PHONE</td>
<td></td>
</tr>
<tr>
<td>TIME_STAMP</td>
<td></td>
</tr>
<tr>
<td>REC_UPDATE</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMPLOYEE</th>
<th>Value “PAYROLL”</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SOURCE</strong></td>
<td></td>
</tr>
<tr>
<td>EXT_ID_TYP*</td>
<td></td>
</tr>
<tr>
<td>EXTERN_ID*</td>
<td></td>
</tr>
<tr>
<td>CAMPUS*</td>
<td></td>
</tr>
<tr>
<td>HOME_DEPT</td>
<td>EMPLOYEE.HOME_DEPT</td>
</tr>
<tr>
<td>EMP_GROUP</td>
<td>EMPLOYEE.EMP_GROUP</td>
</tr>
<tr>
<td>EMP_STATUS</td>
<td>EMPLOYEE.EMP_STATUS</td>
</tr>
<tr>
<td>APPT_TYPE</td>
<td>EMPLOYEE.APPT_TYPE</td>
</tr>
<tr>
<td>TIME_STAMP</td>
<td></td>
</tr>
<tr>
<td>REC_UPDATE</td>
<td></td>
</tr>
</tbody>
</table>
1.3. List the interfaces that take data changes from the application named in the template to other existing applications.

[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: None**

1.3.1. Source application details:

<table>
<thead>
<tr>
<th>Source Application Name:</th>
<th>This is the application named in the template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Application Description:</td>
<td>Description already given for the Application named in the template.</td>
</tr>
<tr>
<td>Source Data Structure: (repeat for each table or filename related to this interface)</td>
<td>Database Name:</td>
</tr>
<tr>
<td>Source Field Name</td>
<td>Field type</td>
</tr>
</tbody>
</table>

1.3.2. Target application details:

<table>
<thead>
<tr>
<th>Target Application Name:</th>
<th>(Brief description)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Application Description:</td>
<td>Database Name:</td>
</tr>
<tr>
<td>Target Data Structure: (repeat for each table or filename related to this interface)</td>
<td>Table Name:</td>
</tr>
<tr>
<td>Target Field Name</td>
<td>Field type</td>
</tr>
</tbody>
</table>

1.3.3. How are the source inputs used to update the target application?

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Source Field Name</th>
<th>Transformation, formatting, and business rules for target</th>
</tr>
</thead>
</table>
2. Step 2
Describe and define Banner 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 AITS integration staff. This section should be completed prior to integration design meetings with AITS integration staff.

2.1. Describe the proposed business processes that the application named in the template will supports, 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: With the implementation of Banner, the ICard system will continue to provide unique identification numbers for University employees and students; at first through interfaces with existing systems which record new employees and new students, and later through a direct interface with Banner which will begin recording new students in June 2002 and new employees in October 2003. Basic person and admitted applicant information from Banner will flow to ICard to generate ids. Generated id information will flow from ICard to Banner.

Timeline: ICard to Banner interface will begin in June 2002 when Banner begins recording new students and the interface will continue indefinitely.

Flow: Basic person and admitted applicant information from Banner will flow to ICard to generate university ids. Generated id information will flow from ICard to Banner.

2.2. What Banner data will this application require?
[Provide the name, description, Banner data structure, target data structure, and how the Banner data will used to update the application named in the template. Use the structure below for each Banner table and Target table and repeat as needed. If not applicable indicate above by N/A.]

N/A

3. Step 3
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 AITS integration staff.

3.1. Application 1: ICard
4. ICard must field incoming University identity query and generate requests and send the appropriate replies.
5. ICard must consume basic person and basic employee synchronization messages to keep its support data current.

3.2. Application 2: Paymaster
3. As basic person and basic employee information changes in Paymaster, Paymaster issues synchronization messages to keep ICard up-to-date.
4. In order to produce basic person and basic employee synchronization messages, Paymaster must send University Identity query requests to ICard and handle the replies. For cases of new employees, Paymaster must send University ID number generation requests and handle the replies.

4. Step 4
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 AITS integration staff.

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

SctSegments and UiSegments

4.1 Existing enterprise object XML definitions that will be reused or modified. This section should include the current definition of the enterprise objects as they appear in the SCT and University of Illinois 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]

<table>
<thead>
<tr>
<th>File</th>
<th>Name</th>
<th>Proposed Definition</th>
</tr>
</thead>
</table>
| SctSegments| UnknownPerson       | Original (as of 3/20/2001):<![ELEMENT UnknownPerson (SocialSecurityNumber?, Name, BirthDate, Gender)]>  
Proposed:  
![ELEMENT UnknownPerson (SocialSecurityNumber?, Name, BirthDate)]  
![ATTLIST UnknownPerson  
gender (Male | Female | Unknown) #REQUIRED  >  
Note: we need to make SSN optional in UnknownPerson to support sync from ICard. |
| BasicPerson | Original (as of 3/12/2001):<![ELEMENT BasicPerson (InstitutionalId, Name, SocialSecurityNumber?, BirthDate?, Gender, DeceasedDate?, Ethnicity*, Address*, Phone*, Email*)]>  
<![ATTLIST BasicPerson...>  
Note: we need to make SSN optional in UnknownPerson to support sync from ICard. |
<table>
<thead>
<tr>
<th>Proposed:</th>
</tr>
</thead>
</table>
| ```xml
<!ELEMENT BasicPerson (InstitutionalId, Name, SocialSecurityNumber?, BirthDate?, DeceasedDate?, Ethnicity*, Address*, Phone*, Email*)>
<!ATTLIST BasicPerson
citizen (Yes | No) #IMPLIED
citizenshipType CDATA #IMPLIED
disabledVeteran (Yes | No) #IMPLIED
deceased (Yes | No) #IMPLIED
gender (Male | Female | Unknown) #REQUIRED
maritalStatus (Married | Single | Divorced) #IMPLIED
vietnamService (Yes | No) #IMPLIED
>`

<table>
<thead>
<tr>
<th>Address Original (as of 8/15/2001):</th>
</tr>
</thead>
</table>
| ```xml
<!ELEMENT Address (Street1?, Street2?, Street3?, CityOrLocality, County?, State?, Nation?, Zip?, EffectiveDate, TerminationDate?)>
<!ATTLIST Address
type CDATA #REQUIRED
>`

<table>
<thead>
<tr>
<th>Proposed:</th>
</tr>
</thead>
</table>
| ```xml
<!ELEMENT Address (Street1?, Street2?, Street3?, CityOrLocality, County?, State?, Country?, Zip?, EffectiveDate, TerminationDate?, Phone?)>
<!ATTLIST Address
type (Home | Office | Extension | Other) #REQUIRED
directory (Suppress | Publish) #REQUIRED
>`

<table>
<thead>
<tr>
<th>Phone Original (as of 12/15/2000):</th>
</tr>
</thead>
</table>
| ```xml
<!ELEMENT Phone (CountryCode?, PhoneArea?, PhoneNumber?, PhoneExtension?)>
<!ATTLIST Phone
type CDATA #REQUIRED
phoneAddressType CDATA #IMPLIED
>`

<table>
<thead>
<tr>
<th>Proposed:</th>
</tr>
</thead>
</table>
| ```xml
<!ELEMENT Phone (CountryCode?, PhoneArea?, PhoneNumber?, PhoneExtension?)>
<!ATTLIST Phone
type (Home | Office) #REQUIRED
directory (Suppress | Publish) #REQUIRED
>`

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

<table>
<thead>
<tr>
<th>File</th>
<th>Name</th>
<th>Proposed Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SctSegments</td>
<td>InstitutionalIdentity</td>
<td>`&lt;ELEMEN[T InstitutionalIdentity (InstitutionalId, UnknownPerson)&gt; Note that this definition will be extended in the future to include other entities besides people, for example UnknownVendor.</td>
</tr>
</tbody>
</table>

5. Step 5
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 AITS integration staff.

5.1. Proposed Messages
[List proposed messages to be used in this interface or indicate that none are proposed for this interface.]
1. Person-InstitutionalIdentity-Create-Sync
2. Person-InstitutionalIdentity-Update-Sync
3. Person-InstitutionalIdentity-Delete-Sync
4. Person-InstitutionalIdentity-Provide-Reply
5. Person-InstitutionalIdentity-Response-Reply

5.2. Existing Messages
[List existing messages to be used in this interface or indicate that none are available for use.]
6. CoreMessaging-Sync-Error-Sync
7. Generic-Response-Reply
8. Person-InstitutionalIdentity-Generate-Request
9. Person-InstitutionalIdentity-Query-Request
10. Person-BasicPerson-Create-Sync
11. Person-BasicPerson-Update-Sync
12. Person-BasicPerson-Delete-Sync
13. Employee-BasicEmployee-Create-Sync
14. Employee-BasicEmployee-Update-Sync
15. Employee-BasicEmployee-Delete-Sync
6. Step 6
Name the existing messaging components that will be used or define the new messaging components 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 AITS integration staff.

6.1. Existing Messaging Components
[List existing components to be used in this interface or indicate that none are available for use.]

6.2. New Messaging Components

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Status</th>
<th>Responsible Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICard Gateway</td>
<td>gateway</td>
<td>planned</td>
<td>OBFS and AITS</td>
</tr>
<tr>
<td>Paymaster Gateway</td>
<td>gateway</td>
<td>planned</td>
<td>AITS</td>
</tr>
</tbody>
</table>

7. Step 7.
For the messaging component(s) of the application named in this template, list the new 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 AITS integration staff.

7.1. Messages

1. Person-InstitutionalIdentity-Create-Sync
2. Person-InstitutionalIdentity-Update-Sync
3. Person-InstitutionalIdentity-Delete-Sync
4. Person-InstitutionalIdentity-Provide-Reply
5. Person-InstitutionalIdentity-Response-Reply
6. CoreMessaging-Sync-Error-Sync
7. Generic-Response-Reply
8. Person-InstitutionalIdentity-Generate-Request
9. Person-InstitutionalIdentity-Query-Request
10. Person-BasicPerson-Create-Sync
11. Person-BasicPerson-Update-Sync
12. Person-BasicPerson-Delete-Sync
13. Employee-BasicEmployee-Create-Sync
14. Employee-BasicEmployee-Update-Sync
15. Employee-BasicEmployee-Delete-Sync

7.2. Message Production Logic
[Describe the message production logic and the XML format for each message that the application or gateway must produce.]

ICard modifications
The following stories assume that ICard data structures will be modified to store a complete, authoritative set of data that comprises an institutional identity. Presently, ICard stores almost all of this information required in its MAIN table with the following structure:
TABLE [dbo].[MAIN] (  
[ID_NUM] [char] (16) NOT NULL ,  
[CREAT_DATE] [datetime] NOT NULL ,  
[DATA_CHANGED] [datetime] NOT NULL ,  
[DATA_DEACT] [datetime] NULL ,  
[CARD_ISSUE] [datetime] NULL ,  
[CARD_CHANGED] [datetime] NULL ,  
[CARD_EXPIRED] [datetime] NULL ,  
[CARD_SEQ] [char] (2) NULL ,  
[PASS_NUM] [char] (4) NULL ,  
[BAR_CODE] [char] (14) NULL ,  
[MAG_STRIPE] [char] (28) NULL ,  
[IMAGE_PATH] [char] (8) NULL ,  
[BADGE_TYPE] [char] (2) NULL ,  
[TIME_STAMP] [timestamp] NOT NULL ,  
[LOGNTRIES] [int] NOT NULL ,  
[UI_TEMP_PIN] [varchar] (255) NULL ,  
[UI_PERM_PIN] [varchar] (255) NULL ,  
[UI_PERM_PASSWORD] [varchar] (255) NULL ,  
[FIRST_NAME] [varchar] (30) NOT NULL ,  
[MID_NAME] [varchar] (35) NOT NULL ,  
[LAST_NAME] [varchar] (30) NOT NULL ,  
[BIRTHDATE] [datetime] (8) NULL ,  
[OCP_QUESTION_A] [int] NULL ,  
[OCP_ANSWER_A] [varchar] (255) NULL ,  
[OCP_QUESTION_B] [int] NULL ,  
[OCP_ANSWER_B] [varchar] (255) NULL
)

Of these attributes, the following are part of an institutional identity:

[ID_NUM] [char] (16) NOT NULL ,  
[FIRST_NAME] [varchar] (30) NOT NULL ,  
[MID_NAME] [varchar] (35) NOT NULL ,  
[LAST_NAME] [varchar] (30) NOT NULL ,  
[BIRTHDATE] [datetime] (8) NULL

The following three attributes complete an institutional identity and are presently lacking in the
ICard MAIN table. These will be added to the MAIN table or a related table, and ICard will be
modified to populate and maintain these attributes with appropriate data.

[CURRENT_SSN] [char] (16) NULL,
[GENDER] [char] (1) NULL,
[NAME_PREFIX] [varchar] (10) NULL,
[NAME_SUFFIX] [varchar] (10) NULL

Once these modifications have been made to ICard, all message consumption and production
logic should be able to operate on the authoritative copy of the institutional identity information in
the MAIN table.

Summary
The ICard gateway must produce InstitutionalId provide replies and InstitutionalId response replies. It must also produce InstitutionalId create, update, and delete synchronization messages whenever a new institutional identity is generated, updated, or deleted by the ICard system. Like any application that consumes synchronization messages, the ICard gateway must also produce synchronization error messages to a standard error topic whenever it encounters errors applying changes from sync messages.

The ICard gateway will produce messages in XML format for each of the messages.

1. **Person-InstitutionalIdentity-Create-Sync** *(Message DTD | Sample Message)*

The ICard Gateway will generate this sync message whenever the ICard system generates a new institutional identity. Presently, new Institutional Identities are generated based on feeds from various sources including the Paymaster and Student systems and interactively at the ID centers. In the future new institutional IDs will also be generated when ICard consumes Person-InstitutionalId-Generate-Requests.

ICard will produce synchronization messages using the RDBMS Queue Table method in conjunction with a polling message producer.

The ICard Gateway must produce this message in the XML format specified in the message DTD.

**Data Area Element values:** (see `SctSegments` for all the elements)

The `DataArea/NewData element is a complex object with the DTD`

```xml
<!ELEMENT NewData (InstitutionalIdentity)>  
```

The `DataArea/NewData/InstitutionalIdentity/InstitutionalId element value is taken from the ICard MAIN.ID_NUM field. In ICard the ID_NUM field is a 16 character field but always contains a 9 digit numeric id in the first 9 positions and no translation is needed.`

The `DataArea/NewData/InstitutionalIdentity/UnknownPerson element is a complex object with the DTD`

```xml
<!ELEMENT UnknownPerson (SocialSecurityNumber, Name, BirthDate, Gender)>  
```

The `UnknownPerson/SocialSecurityNumber element value is taken from the ICard MAIN.CURRENT_SSN field. The SocialSecurityNumber element must be nine numeric digits. MAIN.CURRENT_SSN is optional in ICard.`

The `UnknownPerson/Name element is a complex object with the DTD`

```xml
<!ELEMENT Name (LastName, FirstName?, MiddleInitial?, Prefix?, Suffix?)>  
```

`UnknownPerson/Name@current will take the value “Yes”.`

UnknownPerson/Name/LastName value is taken from the ICard MAIN.LAST_NAME field. In ICard this data should always be present. If no LAST_NAME is present in ICard, then a message cannot be produced since UnknownPerson/Name is required. In ICard it may be formatted in all upper case or in mixed case. In the enterprise message, LastName should be formatted in mixed case using the scrubbing algorithm implemented as `edu.uillinois.aits.enterprise.config.NameScrubber`.

UnknownPerson/Name/FirstName value is taken from the ICard MAIN.FIRST_NAME field. In ICard this data should always be present. If no FIRST_NAME is present in ICard the message will not include this element, as it is optional. In ICard it may be formatted in all upper case or in
mixed case. In the enterprise message, FirstName should be formatted in mixed case using the scrubbing algorithm implemented as `edu.uillinois.aits.enterprise.config.NameScrubber`.

UnknownPerson/Name/MiddleName value is taken from the ICard MAIN.MID_NAME field. In ICard this data is optional. If no MID_NAME is present in ICard the message will not include this element, since it is optional in the message as well. In ICard it may be formatted in all upper case or in mixed case. In the enterprise message, MiddleName should be formatted in mixed case using the scrubbing algorithm implemented as `edu.uillinois.aits.enterprise.config.NameScrubber`.

UnknownPerson/Name/Prefix value is taken from ICard MAIN.NAME_PREFIX. In ICard this data is optional. If no NAME_PREFIX is present in ICard, the message will not include this element, since it is optional in the message as well. In ICard it may be formatted in all upper or in mixed case. In the enterprise message, Prefix should be formatted in mixed case using the scrubbing algorithm implemented as `edu.uillinois.aits.enterprise.config.NameScrubber`.

**Current Suffixes in ICard**

An SQL query of the ICard database provided the following list of suffixes:

- MR
- MR
- MS
- MRS
- DR
- JR
- JR
- SR
- II
- III
- IV
- V
- VI

Possible additional suffixes the NameScrubber should handle are:

- MS
- MRS
- DR
- SR

UnknownPerson/Name/Suffix value may be taken from the ICard MAIN.NAME_SUFFIX field. If no value for MAIN.NAME_SUFFIX is present in ICard, then the message will not include this element since it is optional. In ICard this data may be present and take values like Jr, Sr, etc. In ICard it may be formatted in all upper case or in mixed case. In the enterprise message, Suffix should be translated using valid values defined in EnterpriseFields.

The UnknownPerson/BirthDate element value is taken from the ICard MAIN.BIRTHDATE field. BIRTHDATE appears in date format. If the birthdate is unknown, MAIN.BIRTHDATE will be null in ICard. In the enterprise message, the BirthDate should appear as an appropriately parsed date.
XML date with a month in the range from 1-12, and a day in the range from 1-31 and a four digit year.

UnknownPerson/Gender value will be taken from the ICard MAIN.GENDER field. In ICard this data is generally present and takes values of Male, Female, Unknown, or null. In other words, since MAIN.GENDER is a new attribute in ICard, it will have the enterprise values.

2. **Person-InstitutionalIdentity-Update-Sync** *(Message DTD | Sample Message)*

The ICard Gateway will generate this sync message whenever the ICard system updates an institutional identity. Presently, Institutional Identities information is updated by backend matching cleanup processes, when name changes and such occur in Paymaster or students systems that feed ICard, and through on-line modification of ICard data using ICard clients. See message production logic for item 1 above. Note that the update sync message must include a Baseline InstitutionalIdentity and a new InstitutionalIdentity object.

3. **Person-InstitutionalIdentity-Delete-Sync** *(Message DTD | Sample Message)*

The ICard Gateway will generate this sync message whenever the ICard system deletes an institutional identity. Presently, Institutional Identities are deleted by backend matching cleanup processes and through on-line modification of ICard data using ICard clients. See message production logic for item 1 above.

4. **Person-InstitutionalIdentity-Provide-Reply** *(Message DTD | Sample Message)*

The ICard Gateway must produce this message in the XML format specified in the message DTD when it receives a query request.

**Data Area Element values:** (see SctSegments for all elements)

The DataArea element is a complex object with the DTD

```xml
<!ELEMENT DataArea (InstitutionalIdentity?)>
```

The DataArea/InstitutionalIdentity/InstitutionalId element value is taken from the ICard MAIN.ID_NUM field. In ICard the ID_NUM field is a 16 character field but always contains a 9 digit numeric id in the first 9 positions and no translation is needed. See the production logic for /DataArea/InstitutionalIdentity/UnknownPerson in item 1 above.

5. **Person-InstitutionalIdentity-Response-Reply** *(Message DTD | Sample Message)*

The ICard Gateway must produce this message in the XML format specified in the message DTD in response to a Person-InstitutionalIdentity-Generate-Request.

**Data Area Element values:** (see SctSegments for all elements)

The DataArea element is a complex object with the DTD

```xml
<!ELEMENT DataArea (InstitutionalIdentity?)>
```

The DataArea/InstitutionalIdentity/InstitutionalId element value is taken from the ICard MAIN.ID_NUM field. In ICard the ID_NUM field is a 16 character field but always contains a 9 digit numeric id in the first 9 positions and no translation is needed. See the production logic for /DataArea/InstitutionalIdentity/UnknownPerson in item 1 above.

6. **CoreMessaging-Sync-Error-Sync** *(Message DTD | Sample Message)*

Whenever the ICard gateway or application encounters an error consuming a synchronization message from Paymaster, it must produce a CoreMessagingSyncErrorSync with the appropriate error information.

7. **Generic-Response-Reply** *(Message DTD | Sample Message)*

ICard may produce this message in an error situation in response to a query or generate request from Paymaster (see page 56 of Analysis-Paymaster.doc). It XML format is:

**Data Area Element values:** (see SctSegments for all elements)

The DataArea for this message is a complex object with the DTD
7.3. Message Consumption Logic

[Describe the message consumption logic and the XML format for each message that the application or gateway must consume.]

The ICard gateway must consume the InstitutionalIdentity generate and query requests and BasicPerson and BasicEmployee synchronization messages.

1. Person-InstitutionalIdentity-Generate-Request (Message DTD | Sample Message)

The ICard Gateway must consume this message in the XML format specified in the message DTD.

Data Area Element values: (see SctSegments for all elements)

The DataArea/InstitutionalIdentity element is a complex object with the DTD

```
<!ELEMENT InstitutionalIdentity (InstitutionalId, UnknownPerson)>
```

The DataArea/InstitutionalIdentity/InstitutionalId element contains the institutional ID number. For the purposes of the generate request, this will have a values of “Requested”.

The DataArea/InstitutionalIdentity/UnknownPerson object is a complex object with the DTD

```
<!ELEMENT UnknownPerson (SocialSecurityNumber?, Name, BirthDate, Gender)>
```

The formatting for UnknownPerson is the same as described above for message Person-InstitutionalIdentity-Create-Sync.

Consumption logic:

We will use a standard enterprise consumer to invoke application logic encapsulated in stored procedures.

Procedure Name: MSG_UIN_GEN_REQUEST

Arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURCE</td>
<td>CHAR(10)</td>
</tr>
<tr>
<td>SSN</td>
<td>CHAR(9)</td>
</tr>
<tr>
<td>LASTNAME</td>
<td>VARCHAR(30)</td>
</tr>
<tr>
<td>FIRSTNAME</td>
<td>VARCHAR(30)</td>
</tr>
<tr>
<td>MIDNAME</td>
<td>VARCHAR(35)</td>
</tr>
<tr>
<td>PREFIX</td>
<td>VARCHAR(10)</td>
</tr>
<tr>
<td>SUFFIX</td>
<td>VARCHAR(10)</td>
</tr>
<tr>
<td>BIRTHDATE</td>
<td>DATE</td>
</tr>
<tr>
<td>GENDER</td>
<td>CHAR(1)</td>
</tr>
</tbody>
</table>

Result Set:

<table>
<thead>
<tr>
<th>Result</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATUS</td>
<td>text string with value success or failure</td>
</tr>
<tr>
<td>APP_ERROR_CODE</td>
<td>text string with app-specific values to be defined</td>
</tr>
<tr>
<td>APP_ERROR_DESCRIPTION</td>
<td>text string with app-specific values to be defined</td>
</tr>
<tr>
<td>INSTITUTIONALID</td>
<td>CHAR(9)</td>
</tr>
</tbody>
</table>

ICard should check to see if MAIN.ID_NUM for the UnknownPerson/SocialSecurityNumber exists for the message. If it does then it is an application error and failure. It returns an error telling that it cannot generate since the institutional id already exists for this SocialSecurityNumber.

If no MAIN.ID_NUM exists for the UnknownPerson/SocialSecurityNumber then ICard will generate a new InstitutionalIdentity for the UnknownPerson. Note that this behavior must be
modified prior to phase 2. This approach works for phase I because PersonInstitutionalIdentityGenerateRequests are only being produced for Employees for whom we have SSNs.

Then ICard must:

1. Generate the institutional ID number and provide the result set.

2. Update the ICard database by storing the generated institutional ID number in MAIN.ID_NUM. It must store the SocialSecurityNumber in EXTERN_ID.EXTERN_ID and MAIN.CURRENT_SSN. It must store the other fields in the MAIN table as SocialSecurityNumber in MAIN.EXTERN_ID, LastName in LAST_NAME, FirstName in FIRST_NAME, MiddleName in MID_NAME, Prefix in PREF_NAME, Suffix in SUFF_NAME, BirthDate in BIRTHDATE and Gender in SEX.

3. For every new Identity generation it must also generate a Person-InstitutionalId-Create-Sync message (detailed description is in another section). The sync message must be added to the message queue table for this integration.

2. Person-InstitutionalIdentity-Query-Request (Message DTD | Sample Message)
The ICard Gateway must consume this message in the XML format specified in the message DTD.

Data Area Element values: (see SctSegments for all elements)
The DataArea element is a complex object with the DTD

```
<!ELEMENT DataArea (UnknownPerson)>
```

The DataArea/UnknownPerson object is a complex object with the DTD

```
<!ELEMENT UnknownPerson (SocialSecurityNumber, Name, BirthDate, Gender)>
```

The formatting for UnknownPerson is the same as described above for message Person-InstitutionalIdentity-Create-Sync.

Consumption logic:
We will use a standard enterprise consumer to invoke application logic encapsulated in stored procedures.

Procedure Name: MSG_UIN_QUERY_REQUEST
Arguments:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURCE</td>
<td>CHAR(10)</td>
</tr>
<tr>
<td>SSN</td>
<td>CHAR(9)</td>
</tr>
<tr>
<td>LASTNAME</td>
<td>VARCHAR(30)</td>
</tr>
<tr>
<td>FIRSTNAME</td>
<td>VARCHAR(30)</td>
</tr>
<tr>
<td>MIDNAME</td>
<td>VARCHAR(35)</td>
</tr>
<tr>
<td>PREFIX</td>
<td>VARCHAR(10)</td>
</tr>
<tr>
<td>SUFFIX</td>
<td>VARCHAR(10)</td>
</tr>
<tr>
<td>BIRTHDATE</td>
<td>DATE</td>
</tr>
<tr>
<td>GENDER</td>
<td>CHAR(1)</td>
</tr>
</tbody>
</table>

Result Set:

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATUS</td>
<td>text string with value success or failure</td>
</tr>
<tr>
<td>APP_ERROR_CODE</td>
<td>text string with app-specific values to be defined</td>
</tr>
<tr>
<td>APP_ERROR_DESCRIPTION</td>
<td>text string with app-specific values to be defined</td>
</tr>
<tr>
<td>INSTITUTIONALID</td>
<td>CHAR(9)</td>
</tr>
</tbody>
</table>
ICard should check to see if MAIN.ID_NUM for the UnknownPerson/SocialSecurityNumber exists for the message. If it does then a match has been found and it must produce a Person-InstitutionalId-Provide-Reply message (detailed description is in another section).

If no MAIN.ID_NUM exists for the Unknown/SocialSecurityNumber then ICard it is an application error and failure. It returns an error telling that it cannot provide the institutional id for this SocialSecurityNumber. Note that this behavior must be modified prior to phase 2. This approach works for phase I because PersonInstitutionalIdentityGenerateRequests are only being produced for Employees for whom we have SSNs.

3. Person-BasicPerson-Create-Sync (Message DTD | Sample Message)
The ICard Gateway must consume this message in the XML format specified in the message DTD.

Data Area Element values: (see SctSegments for all elements)
The BasicPerson element is a complex object with the DTD

```xml
<!ELEMENT BasicPerson (InstitutionalId, Name, SocialSecurityNumber?, BirthDate?, Gender, DeceasedDate?, Ethnicity*, Address*, Phone*, Email*)>
<!ATTLIST BasicPerson
citizen CDATA #IMPLIED
citizenshipType CDATA #IMPLIED
disabledVeteran CDATA #IMPLIED
deceased CDATA #IMPLIED
maritalStatus CDATA #IMPLIED
vietnamService CDATA #IMPLIED
```

Consumption logic:
We will use a standard enterprise consumer to invoke application logic encapsulated in stored procedures MSG_PERSON_CREATE_SYNC, MSG_ADDRESS_SYNC, and MSG_PHONE_SYNC.

Procedure Name: MSG_PERSON_CREATE_SYNC
Arguments:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURCE</td>
<td>CHAR(10)</td>
</tr>
<tr>
<td>INSTITUTIONALID</td>
<td>CHAR(16)</td>
</tr>
<tr>
<td>LASTNAME</td>
<td>VARCHAR(30)</td>
</tr>
<tr>
<td>FIRSTNAME</td>
<td>VARCHAR(30)</td>
</tr>
<tr>
<td>MIDNAME</td>
<td>VARCHAR(35)</td>
</tr>
<tr>
<td>PREFIX</td>
<td>VARCHAR(10)</td>
</tr>
<tr>
<td>SUFFIX</td>
<td>VARCHAR(10)</td>
</tr>
<tr>
<td>BIRTHDATE</td>
<td>DATE</td>
</tr>
<tr>
<td>GENDER</td>
<td>CHAR(1)</td>
</tr>
</tbody>
</table>

Result Set:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATUS</td>
<td>text string with value success or failure</td>
</tr>
<tr>
<td>APP_ERROR_CODE</td>
<td>text string with app-specific values to be defined</td>
</tr>
<tr>
<td>APP_ERROR_DESCRIPTION</td>
<td>text string with app-specific values to be defined</td>
</tr>
</tbody>
</table>

Procedure Name: MSG_ADDRESS_CREATE_SYNC
Arguments:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURCE</td>
<td>CHAR(10)</td>
</tr>
<tr>
<td>INSTITUTIONALID</td>
<td>CHAR(16)</td>
</tr>
<tr>
<td>TYPE</td>
<td>CHAR(10)</td>
</tr>
<tr>
<td>STREET1</td>
<td>VARCHAR(30)</td>
</tr>
<tr>
<td>STREET2</td>
<td>VARCHAR(30)</td>
</tr>
</tbody>
</table>
STREET3 VARCHAR(30)
CITY VARCHAR(20)
STATE CHAR(3)
ZIP VARCHAR(10)
COUNTRY CHAR(3)

Result Set:
STATUS text string with value success or failure
APP_ERROR_CODE text string with app-specific values to be defined
APP_ERROR_DESCRIPTION text string with app-specific values to be defined

Procedure Name: MSG_PHONE_CREATE_SYNC
Arguments:
SOURCE CHAR(10)
INSTITUTIONALID CHAR(16)
PHONE_TYPE CHAR(10)
PHONE VARCHAR(25)

Result Set:
STATUS text string with value success or failure
APP_ERROR_CODE text string with app-specific values to be defined
APP_ERROR_DESCRIPTION text string with app-specific values to be defined

4. Person-BasicPerson-Update-Sync (Message DTD | Sample Message)
The ICard Gateway must consume this message in the XML format specified in the message DTD.

Data Area Element values: (see SctSegments for all elements)
The BasicPerson element is a complex object with the DTD

```xml
<!ELEMENT BasicPerson (InstitutionalId, Name, SocialSecurityNumber?, BirthDate?, Gender, DeceasedDate?, Ethnicity*, Address*, Phone*, Email*)>
<!ATTLIST BasicPerson
citizen CDATA #IMPLIED
citizenshipType CDATA #IMPLIED
disabledVeteran CDATA #IMPLIED
deceased CDATA #IMPLIED
maritalStatus CDATA #IMPLIED
vietnamService CDATA #IMPLIED>
```

Consumption logic:
We will use a standard enterprise consumer to invoke application logic encapsulated in stored procedures.

Procedure Name: MSG_PERSON_UPDATE_SYNC
Arguments:
SOURCE CHAR(10)
INSTITUTIONALID_BASELINE CHAR(16)
INSTITUTIONALID CHAR(16)
LASTNAME VARCHAR(30)
FIRSTNAME VARCHAR(30)
MIDNAME VARCHAR(35)
PREFIX VARCHAR(10)
SUFFIX VARCHAR(10)
BIRTHDATE DATE
GENDER CHAR(1)
Result Set:

<table>
<thead>
<tr>
<th>STATUS</th>
<th>text string with value success or failure</th>
</tr>
</thead>
<tbody>
<tr>
<td>APP_ERROR_CODE</td>
<td>text string with app-specific values to be defined</td>
</tr>
<tr>
<td>APP_ERROR_DESCRIPTION</td>
<td>text string with app-specific values to be defined</td>
</tr>
</tbody>
</table>

Procedure Name: MSG_ADDRESS_UPDATE_SYNC

Arguments:

<table>
<thead>
<tr>
<th>SOURCE</th>
<th>CHAR(10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSTITUTIONALID_BASELINE</td>
<td>CHAR(16)</td>
</tr>
<tr>
<td>INSTITUTIONALID</td>
<td>CHAR(16)</td>
</tr>
<tr>
<td>TYPE</td>
<td>CHAR(10)</td>
</tr>
<tr>
<td>STREET1</td>
<td>VARCHAR(30)</td>
</tr>
<tr>
<td>STREET2</td>
<td>VARCHAR(30)</td>
</tr>
<tr>
<td>STREET3</td>
<td>VARCHAR(30)</td>
</tr>
<tr>
<td>CITY</td>
<td>VARCHAR(20)</td>
</tr>
<tr>
<td>STATE</td>
<td>CHAR(3)</td>
</tr>
<tr>
<td>ZIP</td>
<td>VARCHAR(10)</td>
</tr>
<tr>
<td>COUNTRY</td>
<td>CHAR(3)</td>
</tr>
</tbody>
</table>

Result Set:

<table>
<thead>
<tr>
<th>STATUS</th>
<th>text string with value success or failure</th>
</tr>
</thead>
<tbody>
<tr>
<td>APP_ERROR_CODE</td>
<td>text string with app-specific values to be defined</td>
</tr>
<tr>
<td>APP_ERROR_DESCRIPTION</td>
<td>text string with app-specific values to be defined</td>
</tr>
</tbody>
</table>

Procedure Name: MSG_PHONE_UPDATE_SYNC

Arguments:

<table>
<thead>
<tr>
<th>SOURCE</th>
<th>CHAR(10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSTITUTIONALID_BASELINE</td>
<td>CHAR(16)</td>
</tr>
<tr>
<td>INSTITUTIONALID</td>
<td>CHAR(16)</td>
</tr>
<tr>
<td>PHONE_TYPE</td>
<td>CHAR(10)</td>
</tr>
<tr>
<td>PHONE</td>
<td>VARCHAR(25)</td>
</tr>
</tbody>
</table>

Result Set:

<table>
<thead>
<tr>
<th>STATUS</th>
<th>text string with value success or failure</th>
</tr>
</thead>
<tbody>
<tr>
<td>APP_ERROR_CODE</td>
<td>text string with app-specific values to be defined</td>
</tr>
<tr>
<td>APP_ERROR_DESCRIPTION</td>
<td>text string with app-specific values to be defined</td>
</tr>
</tbody>
</table>

5. Person-BasicPerson-Delete-Sync (Message DTD | Sample Message)

The ICard Gateway must consume this message in the XML format specified in the message DTD.

Data Area Element values: (see SctSegments for all elements)

The DeleteData element is an object with the DTD

```xml
<!ELEMENT DeleteAction EMPTY>
<!ATTLIST DeleteAction
type (Delete | Purge) #REQUIRED>
```

The DeleteData/DeleteAction@type will take a value of “Delete” or “Purge”.

Consumption logic:

We will use a standard enterprise consumer to invoke application logic encapsulated in stored procedures.
Procedure Name: MSG_PERSON_DELETE_SYNC  
Arguments:  
  SOURCE   CHAR(10)  
  INSTITUTIONALID   CHAR(16)  

Result Set:  
  STATUS   text string with value success or failure  
  APP_ERROR_CODE   text string with app-specific values to be defined  
  APP_ERROR_DESCRIPTION   text string with app-specific values to be defined  

6. Employee-BasicEmployee-Create-Sync  (Message DTD | Sample Message)  
The ICard Gateway must consume this message in the XML format specified in the message DTD.  

Data Area Element values:  (see SctSegments for all elements)  
The BasicEmployee element is a complex object with the DTD  

```
<!ELEMENT BasicEmployee (InstitutionalId, HomeChartOfAccounts?, HomeOrganization,  
DistributionChartOfAccounts?, DistributionOrganization, EmployeeClass, LeaveCategory,  
BenefitCategory, EmploymentDates, EmploymentLeave?, EmploymentTermination?, DisabilityCode*)>  
<!ATTLIST BasicEmployee  
group CDATA #IMPLIED  
status CDATA #REQUIRED  
flsa CDATA #REQUIRED  
timeStatus CDATA #REQUIRED  
payCampus CDATA #REQUIRED  
>  
```

Consumption logic:  
We will use a standard enterprise consumer to invoke application logic encapsulated in stored procedures.  

Procedure Name: MSG_EMPLOYEE_CREATE_SYNC  
Arguments:  
  SOURCE   CHAR(10)  
  INSTITUTIONALID   CHAR(16)  
  CAMPUS   CHAR(1)  
  HOME-DEPT   CHAR(4)  
  EML-GROUP   CHAR(1)  
  EML-STATUS   CHAR(1)  
  APPT-TYPE   CHAR(1)  

Result Set:  
  STATUS   text string with value success or failure  
  APP_ERROR_CODE   text string with app-specific values to be defined  
  APP_ERROR_DESCRIPTION   text string with app-specific values to be defined  

7. Employee-BasicEmployee-Update-Sync  (Message DTD | Sample Message)  
The ICard Gateway must consume this message in the XML format specified in the message DTD.  

Data Area Element values:  (see SctSegments for all elements)  
The BasicEmployee element is a complex object with the DTD  

```
<!ELEMENT BasicEmployee (InstitutionalId, HomeChartOfAccounts?, HomeOrganization,  
DistributionChartOfAccounts?, DistributionOrganization, EmployeeClass, LeaveCategory,  
BenefitCategory, EmploymentDates, EmploymentLeave?, EmploymentTermination?, DisabilityCode*)>  
<!ATTLIST BasicEmployee  
group CDATA #IMPLIED  
status CDATA #REQUIRED  
flsa CDATA #REQUIRED  
>  
```
Consumption logic:
We will use a standard enterprise consumer to invoke application logic encapsulated in stored procedures.

Procedure Name: MSG_EMPLOYEE_UPDATE_SYNC
Arguments:

SOURCE CHAR(10)
INSTITUTIONALID_BASELINE CHAR(16)
INSTITUTIONALID CHAR(16)
CAMPUS CHAR(1)
HOME-DEPT CHAR(4)
EMPL-GROUP CHAR(1)
EMPL-STATUS CHAR(1)
APPT-TYPE CHAR(1)

Result Set:
STATUS text string with value success or failure
APP_ERROR_CODE text string with app-specific values to be defined
APP_ERROR_DESCRIPTION text string with app-specific values to be defined

8. Employee-BasicEmployee-Delete-Sync (Message DTD | Sample Message)
The ICard Gateway must consume this message in the XML format specified in the message DTD.

Data Area Element values: (see SctSegments for all elements)
The DeleteData element is an object with the DTD

The DeleteData/DeleteAction@type will take a value of “Delete” or “Purge”.

Consumption logic:
We will use a standard enterprise consumer to invoke application logic encapsulated in stored procedures.

Procedure Name: MSG_EMPLOYEE_DELETE_SYNC
Arguments:

SOURCE CHAR(10)
INSTITUTIONALID CHAR(16)

Result Set:
STATUS text string with value success or failure
APP_ERROR_CODE text string with app-specific values to be defined
APP_ERROR_DESCRIPTION text string with app-specific values to be defined

8. Step 8
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 AITS integration staff.

8.1. Messaging Component 1: Paymaster

8.1.1. Messages
1. Person-InstitutionalIdentity-Create-Sync
2. Person-InstitutionalIdentity-Update-Sync
3. Person-InstitutionalIdentity-Delete-Sync
4. Person-InstitutionalIdentity-Provide-Reply
5. Person-InstitutionalIdentity-Response-Reply
6. CoreMessaging-Sync-Error-Sync
7. Generic-Response-Reply
8. Person-InstitutionalIdentity-Generate-Request
9. Person-InstitutionalIdentity-Query-Request
10. Person-BasicPerson-Create-Sync
11. Person-BasicPerson-Update-Sync
12. Person-BasicPerson-Delete-Sync
13. Employee-BasicEmployee-Create-Sync
14. Employee-BasicEmployee-Update-Sync
15. Employee-BasicEmployee-Delete-Sync

8.1.2. Brief Story

The Paymaster gateway must produce the BasicPerson and BasicEmployee synchronization messages as well as the query and generate requests for InstitutionalId. It must consume the InstitutionalId reply messages in response to the query and generate requests. In addition, it must consume the InstitutionalId synchronization messages generated by ICard.

8.2. Messaging Component 2: Banner (Phase 2)

8.2.1. Messages

1. [To be completed]

8.2.2 Brief Story

The interface to Banner will begin in June 2002 when Banner starts recording new students information. Banner will provide basic person and admitted applicant data to ICard, which will generate university identification data and send the data back to Banner.
Appendix 2: Completed Analysis Template Example – SunGard SCT Banner

This completed template is an example of an organization’s use of the OpenEAI template to perform the analysis for the SunGard SCT Banner system. The template was customized and curtailed for the organization’s needs. This completed template was given to the OpenEAI project by the University of Illinois and SunGard SCT Banner.

Application Name: Banner
Module Name: Person Biographic Demographic
Phase: 1
Timeframe: December 2001—June 2002

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 AITS staff.**

Step 1. 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 AITS integration staff. This section should be completed prior to integration design meetings with AITS integration staff.

Step 2. Describe and define Banner 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 AITS integration staff. This section should be completed prior to integration design meetings with AITS integration staff.

Step 3. 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 AITS integration staff.

Step 4. 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 AITS integration staff.

Step 5. 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 AITS integration staff.

Step 6. Name the existing messaging components that will be used or define the new messaging components 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 AITS integration staff.

Step 7. For the messaging component(s) specifically for the application named in this template, list the new 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 AITS integration staff.

Step 8. For each remaining messaging component listed in step 6, list the new messages that component must produce and consume and provide brief stories describing of the prescribed production and consumption logic. This section should be completed during the integration design phase with AITS integration staff.

The next sections of the document describe the details required for the steps listed above.
1. Step 1
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 AITS integration staff. This section should be completed prior to integration design meetings with AITS integration staff.

What data does this application store and operate on that it does not create itself? Presently, this data is usually acquired by the application through batch extracts and feeds, remote procedure calls, or data replication. For example, applications such as Payroll History Database (PHD) and ECOS maintain and in some cases update appointment data. This data is kept synchronized with Paymaster by scheduled batch feeds from Paymaster to these applications. Changes made to this data in these applications are updated in Paymaster through scheduled batch feeds from these applications to Paymaster. Another way to look at 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 application named in the template supports, how data is presently acquired, the timeline, and the current 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: XXXXXXXX

Timeline: XXXXXXXX

Flow: XXXXXXXX

Description:
Banner is the ERP software product developed by SCT Corporation. It has built-in Student Administration, Human Resources, and Finance functionalities. Specifically, it runs on Oracle’s Object/Relational database management system and supports the full complement of Oracle tools.

The University of Illinois implements (or will implement) the following components in Banner:
- Student (recruiting and admissions, course catalogs, class schedules, student registration, academic records, transfer articulation, graduation processing, faculty workload analysis, enrollment management, curriculum advising, and program planning); Financial Aid (funds management, requirements tracking, disbursement, budgeting, loan processing, need analysis, verification, student employment, award history/transcripts, reporting, awards packaging), and Accounts Receivable (charge and payment information, invoice and statement generation, cashiering)
- Human Resources (HR biographic/demographic information, employee relations, position management, applicant tracking, benefits administration, compensation administration, employment administration, personnel services budgeting, time & attendance reporting, payroll calculation, payroll adjustment & history)
- Finance (chart of accounts, budgets, general ledger, accounts payable, purchasing, fixed assets, grants/contracts)

Timeline: Implementation will be done in phases. Refer to the most up-to-date UI2 project schedule at the UI2 website. This particular integration dealing with HR Bio-Demo will be implemented in December 2001.
Flow: Data will flow one way only, from legacy applications to Banner.
1.2. List the current application interfaces for the application named in the template that synchronizes data changes made in other applications.

[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 \textbf{N/A}.]

\textbf{1.2.1. Source application details:}

\textbf{1.2.2. Target application details:}

<table>
<thead>
<tr>
<th>Target Application Name:</th>
<th>This is the application named in the template (i.e. Banner)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Application Description</td>
<td>Banner general person, Employee, Address, Phone and email tables will be updated using Paymaster new employee data.</td>
</tr>
<tr>
<td>Target Data Structure:</td>
<td>Database Name: Banner Table Name: SPRIDEN</td>
</tr>
<tr>
<td>(repeat for each table or filename related to this interface)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPRIDEN_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number of the person.</td>
</tr>
<tr>
<td>SPRIDEN_ID</td>
<td>VARCHAR2</td>
<td>9</td>
<td>NO</td>
<td>This field defines the identification number used to access person on-line.</td>
</tr>
<tr>
<td>SPRIDEN_LAST_NAME</td>
<td>VARCHAR2</td>
<td>60</td>
<td>NO</td>
<td>This field defines the last name of person.</td>
</tr>
<tr>
<td>SPRIDEN_FIRST_NAME</td>
<td>VARCHAR2</td>
<td>15</td>
<td>YES</td>
<td>This field identifies the first name of person.</td>
</tr>
<tr>
<td>SPRIDEN_MI</td>
<td>VARCHAR2</td>
<td>15</td>
<td>YES</td>
<td>This field identifies the middle name of person.</td>
</tr>
<tr>
<td>SPRIDEN_CHANGE_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>This field identifies whether type of change made to the record was an ID number change or a name change. Valid values: I - ID change, N - name change.</td>
</tr>
<tr>
<td>SPRIDEN_ENTITY_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>This field identifies whether record is person or non-person record. It does not display on the form. Valid values: P - person, C - non-person.</td>
</tr>
<tr>
<td>SPRIDEN_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>This field defines the most current date record is created or changed.</td>
</tr>
<tr>
<td>SPRIDEN_USER</td>
<td>VARCHAR2</td>
<td>30</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>SPRIDEN_ORIGIN</td>
<td>VARCHAR2</td>
<td>30</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>SPRIDEN_SEARCH_LAST_NAME</td>
<td>VARCHAR2</td>
<td>60</td>
<td>YES</td>
<td>The Last Name field with all spaces and punctuation removed and all letters capitalized.</td>
</tr>
<tr>
<td>SPRIDEN_SEARCH_FIRST_NAME</td>
<td>VARCHAR2</td>
<td>15</td>
<td>YES</td>
<td>The First Name field with all spaces and punctuation removed and all letters capitalized.</td>
</tr>
<tr>
<td>SPRIDEN_SEARCH_MI</td>
<td>VARCHAR2</td>
<td>15</td>
<td>YES</td>
<td>The MI (Middle Initial) field with all spaces and punctuation removed and all letters capitalized.</td>
</tr>
<tr>
<td>SPRIDEN_SOUNDEX_LAST_NAME</td>
<td>CHAR</td>
<td>4</td>
<td>YES</td>
<td>The Last Name field in SOUNDEX phonetic format.</td>
</tr>
<tr>
<td>SPRIDEN_SOUNDEX_FIRST_NAME</td>
<td>CHAR</td>
<td>4</td>
<td>YES</td>
<td>The First Name field in SOUNDEX phonetic format.</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------</td>
<td>---</td>
<td>-----</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>SPRIDEN_NTYP_CODE</td>
<td>VARCHAR2</td>
<td>4</td>
<td>YES</td>
<td></td>
</tr>
</tbody>
</table>
**Target Data Structure:**
(repeat for each table or filename related to this interface)

**Database Name:** Banner  
**Table Name:** SPBPERS

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPBPERS_INCAR_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>The indication of the individual's incarceration.</td>
</tr>
<tr>
<td>SPBPERS_WEBID</td>
<td>RAW</td>
<td>1</td>
<td>YES</td>
<td>Web identification session token is now obsolete.</td>
</tr>
<tr>
<td>SPBPERS_WEB_LAST_ACCESS</td>
<td>RAW</td>
<td>1</td>
<td>YES</td>
<td>Web last access time stamp is now obsolete.</td>
</tr>
<tr>
<td>SPBPERS_PIN_DISABLED_IND</td>
<td>RAW</td>
<td>1</td>
<td>YES</td>
<td>Indicates whether or not the PIN number is disabled. This column is obsolete, now gobtpac_pin_disabled_ind</td>
</tr>
<tr>
<td>SPBPERS_ITIN</td>
<td>NUMBER</td>
<td>9</td>
<td>YES</td>
<td>The international tax id number.</td>
</tr>
<tr>
<td>SPBPERS_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal Identification Number of Person.</td>
</tr>
<tr>
<td>SPBPERS_SSN</td>
<td>VARCHAR2</td>
<td>9</td>
<td>YES</td>
<td>This field maintains person social security number.</td>
</tr>
<tr>
<td>SPBPERS_BIRTH_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>This field maintains person birth date.</td>
</tr>
<tr>
<td>SPBPERS_LGCY_CODE</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>This field maintains legacy code associated with person.</td>
</tr>
<tr>
<td>SPBPERS_ETHN_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>YES</td>
<td>This field maintains ethnicity code associated with person.</td>
</tr>
<tr>
<td>SPBPERS_MRTL_CODE</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>This field maintains marital status associated with person.</td>
</tr>
<tr>
<td>SPBPERS_RELG_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>YES</td>
<td>This field maintains religious affiliation associated with person.</td>
</tr>
<tr>
<td>SPBPERS_SEX</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>This field maintains the sex of person. Valid values are: M - Male, F - Female, N - Unknown.</td>
</tr>
<tr>
<td>SPBPERS_CONFID_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>This field identifies if a person record is confidential. Valid value is: Y - confidential.</td>
</tr>
<tr>
<td>SPBPERS_DEAD_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>This field indicates if a person is deceased. Valid value is: Y - deceased.</td>
</tr>
<tr>
<td>SPBPERS_VETC_FILE_NUMBER</td>
<td>VARCHAR2</td>
<td>10</td>
<td>YES</td>
<td>This field maintains veteran identification number associated with person.</td>
</tr>
<tr>
<td>SPBPERS_LEGAL_NAME</td>
<td>VARCHAR2</td>
<td>60</td>
<td>YES</td>
<td>This field maintains legal name associated with person.</td>
</tr>
<tr>
<td>SPBPERS_PREF_FIRST_NAME</td>
<td>VARCHAR2</td>
<td>15</td>
<td>YES</td>
<td>This field maintains the preferred first name associated with person.</td>
</tr>
<tr>
<td>SPBPERS_NAME_PREFIX</td>
<td>VARCHAR2</td>
<td>20</td>
<td>YES</td>
<td>This field maintains the prefix (Mr, Mrs, etc) used before person name.</td>
</tr>
<tr>
<td>SPBPERS_NAME_SUFFIX</td>
<td>VARCHAR2</td>
<td>20</td>
<td>YES</td>
<td>This field maintains the suffix (Jr, Sr, etc) used after person name.</td>
</tr>
<tr>
<td>SPBPERS_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>This field defines the most current date a record is added or changed.</td>
</tr>
<tr>
<td>SPBPERS_VERA_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Vietnam Service era Indicator.</td>
</tr>
<tr>
<td>SPBPERS_CITZ_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Citizen Indicator.</td>
</tr>
<tr>
<td>SPBPERS_DEAD_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>Person Deceased Date.</td>
</tr>
<tr>
<td>SPBPERS_PIN</td>
<td>RAW</td>
<td>1</td>
<td>YES</td>
<td>Person Identification Number. This column is obsolete, now gobtpac_pin</td>
</tr>
<tr>
<td>SPBPERS_CITZ_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>YES</td>
<td>Person Citizen Type</td>
</tr>
<tr>
<td>SPBPERS_HAIR_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>YES</td>
<td>The hair color of the person being defined.</td>
</tr>
<tr>
<td>SPBPERS_EYES_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>YES</td>
<td>The eye color of the person being defined.</td>
</tr>
<tr>
<td>SPBPERS_CITY_BIRTH</td>
<td>VARCHAR2</td>
<td>20</td>
<td>YES</td>
<td>The city where the person was born.</td>
</tr>
<tr>
<td>SPBPERS_STAT_CODE_BIRTH</td>
<td>VARCHAR2</td>
<td>3</td>
<td>YES</td>
<td>The state in which the person was born.</td>
</tr>
<tr>
<td>SPBPERS_DRIVER_LICENSE</td>
<td>VARCHAR2</td>
<td>20</td>
<td>YES</td>
<td>The Driver License Number as it appears on the actual license.</td>
</tr>
<tr>
<td>Target Data Structure: (repeat for each table or filename related to this interface)</td>
<td>Database Name: Banner</td>
<td>Table Name: SPBADDR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Target Field Name</td>
<td>Field type</td>
<td>Width</td>
<td>Nullable</td>
<td>Comments</td>
</tr>
<tr>
<td>SPRADDR_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number of person.</td>
</tr>
<tr>
<td>SPRADDR_ATYP_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>NO</td>
<td>This field identifies the type of address associated with person.</td>
</tr>
<tr>
<td>SPRADDR_SEQNO</td>
<td>NUMBER</td>
<td>2</td>
<td>NO</td>
<td>This field assigns an internal sequence number to each address type associated with person. This field does not display on screen.</td>
</tr>
<tr>
<td>SPRADDR_FROM_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>This field maintains the effective start date of address associated with person.</td>
</tr>
<tr>
<td>SPRADDR_TO_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>This field maintains the effective end date of address associated with person.</td>
</tr>
<tr>
<td>SPRADDR_STREET_LINE1</td>
<td>VARCHAR2</td>
<td>30</td>
<td>YES</td>
<td>This field maintains the first line of the address associated with person.</td>
</tr>
<tr>
<td>SPRADDR_STREET_LINE2</td>
<td>VARCHAR2</td>
<td>30</td>
<td>YES</td>
<td>This field maintains the second line of the address associated with person.</td>
</tr>
<tr>
<td>SPRADDR_STREET_LINE3</td>
<td>VARCHAR2</td>
<td>30</td>
<td>YES</td>
<td>This field maintains the third line of the address associated with person.</td>
</tr>
<tr>
<td>SPRADDR_CITY</td>
<td>VARCHAR2</td>
<td>20</td>
<td>NO</td>
<td>This field maintains the city associated with the address of person.</td>
</tr>
<tr>
<td>SPRADDR_STAT_CODE</td>
<td>VARCHAR2</td>
<td>3</td>
<td>YES</td>
<td>This field maintains the state associated with the address of person.</td>
</tr>
<tr>
<td>SPRADDR_ZIP</td>
<td>VARCHAR2</td>
<td>10</td>
<td>YES</td>
<td>This field maintains the zip code associated with the address of person.</td>
</tr>
<tr>
<td>SPRADDR_CNTY_CODE</td>
<td>VARCHAR2</td>
<td>5</td>
<td>YES</td>
<td>This field maintains the county associated with the address of person.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Type</td>
<td>Width</td>
<td>Nullable</td>
<td>Comments</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------</td>
<td>-------</td>
<td>----------</td>
<td>-----------------------------------------------------------</td>
</tr>
<tr>
<td>SPRADDR_NATN_CODE</td>
<td>VARCHAR2</td>
<td>5</td>
<td>YES</td>
<td>This field maintains the nation/country associated with</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>the address of person.</td>
</tr>
<tr>
<td>SPRADDR_PHONE_AREA</td>
<td>VARCHAR2</td>
<td>3</td>
<td>YES</td>
<td>This field maintains the area code of the phone</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>number associated with address of person.</td>
</tr>
<tr>
<td>SPRADDR_PHONE_NUMBER</td>
<td>VARCHAR2</td>
<td>7</td>
<td>YES</td>
<td>This field maintains the phone number associated with</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>address of person.</td>
</tr>
<tr>
<td>SPRADDR_PHONE_EXT</td>
<td>VARCHAR2</td>
<td>4</td>
<td>YES</td>
<td>This field maintains the extension of the phone number</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>associated with address of person.</td>
</tr>
<tr>
<td>SPRADDR_STATUS_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>This field identifies if address information is inactive.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Valid value is I - Inactive.</td>
</tr>
<tr>
<td>SPRADDR_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>This field defines the most current date a record is</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>added or changed.</td>
</tr>
<tr>
<td>SPRADDR_USER</td>
<td>VARCHAR2</td>
<td>30</td>
<td>YES</td>
<td>The Id for the User who create or update the record.</td>
</tr>
<tr>
<td>SPRADDR_ASRC_CODE</td>
<td>VARCHAR2</td>
<td>4</td>
<td>YES</td>
<td>Address source code.</td>
</tr>
<tr>
<td>SPRADDR_DELIVERY_POINT</td>
<td>NUMBER</td>
<td>2</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>SPRADDR_CORRECTION_DIGIT</td>
<td>NUMBER</td>
<td>1</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>SPRADDR_CARRIER_ROUTE</td>
<td>VARCHAR2</td>
<td>4</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>SPRADDR_GST_TAX_ID</td>
<td>VARCHAR2</td>
<td>15</td>
<td>YES</td>
<td>GST TAX IDENTIFICATION NUMBER: Goods and Services Tax</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Identification of vendor at this address</td>
</tr>
<tr>
<td>SPRADDR_REVIEWED_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Reviewed Indicator, Y indicates the address has been</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>reviewed.</td>
</tr>
<tr>
<td>SPRADDR_REVIEWED_USER</td>
<td>VARCHAR2</td>
<td>30</td>
<td>YES</td>
<td>Reviewed User, indicates the user who reviewed the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>address.</td>
</tr>
</tbody>
</table>

**Target Data Structure:**
(repeat for each table or filename related to this interface)

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Database Name: Banner Table Name: SPBTELE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPRTELE_PIDM</td>
<td></td>
</tr>
<tr>
<td>SPRTELE_SEQNO</td>
<td></td>
</tr>
<tr>
<td>SPRTELE_TELE_CODE</td>
<td></td>
</tr>
<tr>
<td>SPRTELE_ACTIVITY_DATE</td>
<td></td>
</tr>
<tr>
<td>SPRTELE_PHONE_AREA</td>
<td></td>
</tr>
<tr>
<td>SPRTELE_PHONE_NUMBER</td>
<td></td>
</tr>
<tr>
<td>SPRTELE_PHONE_EXT</td>
<td></td>
</tr>
<tr>
<td>SPRTELE_STATUS_IND</td>
<td></td>
</tr>
<tr>
<td>SPRTELE_ATYP_CODE</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPRTELE_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number.</td>
</tr>
<tr>
<td>SPRTELE_SEQNO</td>
<td>NUMBER</td>
<td>3</td>
<td>NO</td>
<td>Unique sequence number for telephone numbers</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>associated with PIDM.</td>
</tr>
<tr>
<td>SPRTELE_TELE_CODE</td>
<td>VARCHAR2</td>
<td>4</td>
<td>NO</td>
<td>Telephone Type Code.</td>
</tr>
<tr>
<td>SPRTELE_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>Date of last activity for telephone record.</td>
</tr>
<tr>
<td>SPRTELE_PHONE_AREA</td>
<td>VARCHAR2</td>
<td>3</td>
<td>YES</td>
<td>Telephone number area code.</td>
</tr>
<tr>
<td>SPRTELE_PHONE_NUMBER</td>
<td>VARCHAR2</td>
<td>7</td>
<td>YES</td>
<td>Telephone number.</td>
</tr>
<tr>
<td>SPRTELE_PHONE_EXT</td>
<td>VARCHAR2</td>
<td>4</td>
<td>YES</td>
<td>Telephone number extension.</td>
</tr>
<tr>
<td>SPRTELE_STATUS_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Status of telephone number, active or inactive.</td>
</tr>
<tr>
<td>SPRTELE_ATYP_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>YES</td>
<td>Optional Address Type code associated with telephone</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>number.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Type</td>
<td>Width</td>
<td>Nullable</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------</td>
<td>-------</td>
<td>----------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SPRTELE_ADDR_SEQNO</td>
<td>NUMBER</td>
<td>2</td>
<td>YES</td>
<td>Address type sequence number associated with address type.</td>
</tr>
<tr>
<td>SPRTELE_PRIMARY_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Primary indicator to denote primary telephone numbers based on telephone types.</td>
</tr>
<tr>
<td>SPRTELE_UNLIST_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Unlisted telephone number indicator.</td>
</tr>
<tr>
<td>SPRTELE_COMMENT</td>
<td>VARCHAR2</td>
<td>60</td>
<td>YES</td>
<td>Comment relating to telephone number.</td>
</tr>
<tr>
<td>SPRTELE_INTL_ACCESS</td>
<td>VARCHAR2</td>
<td>16</td>
<td>YES</td>
<td>Free format International access code for telephone number including country and city code.</td>
</tr>
</tbody>
</table>

**Target Data Structure:**
Database Name: Banner
Table Name: GOREMAL

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>GOREMAL_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>The pidm of the entity who owns this e-mail information.</td>
</tr>
<tr>
<td>GOREMAL_EMAL_CODE</td>
<td>VARCHAR2</td>
<td>4</td>
<td>NO</td>
<td>The type of e-mail address.</td>
</tr>
<tr>
<td>GOREMAL_EMAIL_ADDRESS</td>
<td>VARCHAR2</td>
<td>90</td>
<td>NO</td>
<td>The e-mail address.</td>
</tr>
<tr>
<td>GOREMAL_STATUS_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>NO</td>
<td>The status of the e-mail address.</td>
</tr>
<tr>
<td>GOREMAL_PREFERRED_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>NO</td>
<td>This column indicates if the e-mail address is the preferred contact address.</td>
</tr>
<tr>
<td>GOREMAL_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>The date on which the row was added or modified.</td>
</tr>
<tr>
<td>GOREMAL_USER_ID</td>
<td>VARCHAR2</td>
<td>30</td>
<td>NO</td>
<td>The user id when the row was added or modified.</td>
</tr>
<tr>
<td>GOREMAL_COMMENT</td>
<td>VARCHAR2</td>
<td>60</td>
<td>YES</td>
<td>This is a free format comment regarding the e-mail information.</td>
</tr>
<tr>
<td>GOREMAL_DISP_WEB_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>NO</td>
<td>Indicate whether an e-mail address should appear on Web.</td>
</tr>
</tbody>
</table>

**Target Data Structure:**
Database Name: Banner
Table Name: PEBEMPL

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PEBEMPL_INTERNAL_FT_PT_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>NO</td>
<td>Internal Part Time/Full Time Indicator, used for EEO reporting.</td>
</tr>
<tr>
<td>PEBEMPL_DICD_CODE</td>
<td>VARCHAR2</td>
<td>3</td>
<td>YES</td>
<td>DISTRICT/DIVISION CODE: The district or division associated with an employee.</td>
</tr>
<tr>
<td>PEBEMPL_EGRP_CODE</td>
<td>VARCHAR2</td>
<td>4</td>
<td>YES</td>
<td>EMPLOYEE GROUP CODE: An employee classification which is broken down into employee classes.</td>
</tr>
<tr>
<td>PEBEMPL_IPEDS_SOFT_MONEY_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>Field Name</td>
<td>Type</td>
<td>Length</td>
<td>Nullable</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>----------</td>
<td>--------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PEBEMPL_FIRST_WORK_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>The first day that employee worked.</td>
</tr>
<tr>
<td>PEBEMPL_LAST_WORK_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>The last day that employee worked.</td>
</tr>
<tr>
<td>PEBEMPL_CALIF_PENSION_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>The California Pension Ind.</td>
</tr>
<tr>
<td>PEBEMPL_NRSI_CODE</td>
<td>VARCHAR2</td>
<td>4</td>
<td>YES</td>
<td>The National Retirement System Ind.</td>
</tr>
<tr>
<td>PEBEMPL_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>PIDM: Internal System Identification Number.</td>
</tr>
<tr>
<td>PEBEMPL_EMPL_STATUS</td>
<td>VARCHAR2</td>
<td>1</td>
<td>NO</td>
<td>EMPLOYEE STATUS: The status of the employee.</td>
</tr>
<tr>
<td>PEBEMPL_COAS_CODE_HOME</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>CHART OF ACCOUNTS CODE HOME: The home Chart of Accounts Code for the employee.</td>
</tr>
<tr>
<td>PEBEMPL_ORGN_CODE_HOME</td>
<td>VARCHAR2</td>
<td>6</td>
<td>NO</td>
<td>HOME ORGANIZATION CODE: The home organization code of the employee.</td>
</tr>
<tr>
<td>PEBEMPL_COAS_CODE_DIST</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>CHART OF ACCOUNTS CODE DISTRIBUTION: The Chart of Accounts Code to which this employees labor is distributed.</td>
</tr>
<tr>
<td>PEBEMPL_ORGN_CODE_DIST</td>
<td>VARCHAR2</td>
<td>6</td>
<td>NO</td>
<td>ORGANIZATION CODE DISTRIBUTION: The organization code used for labor distribution.</td>
</tr>
<tr>
<td>PEBEMPL_ECLS_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>NO</td>
<td>EMPLOYEE CLASS CODE: The employee grouping to which the employee belongs.</td>
</tr>
<tr>
<td>PEBEMPL_LCAT_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>NO</td>
<td>LEAVE CATEGORY CODE: The leave category to which the employee belongs.</td>
</tr>
<tr>
<td>PEBEMPL_BCAT_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>NO</td>
<td>BENEFIT CATEGORY CODE: The benefit category to which the employee belongs.</td>
</tr>
<tr>
<td>PEBEMPL_FIRST_HIRE_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>FIRST HIRE DATE: The date the employee was first hired.</td>
</tr>
<tr>
<td>PEBEMPL_CURRENT_HIRE_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>CURRENT HIRE DATE: The employees current date of hire.</td>
</tr>
<tr>
<td>PEBEMPL_ADJ_SERVICE_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>ADJUSTED SERVICE DATE: Date used to determine years of service for leave accrual if the First Hire Date and Current Hire Date are not equal.</td>
</tr>
<tr>
<td>PEBEMPL_SENIORITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>SENIORITY DATE: Currently not used by system. Reserved for future use.</td>
</tr>
<tr>
<td>PEBEMPL_LREA_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>YES</td>
<td>LEAVE REASON CODE: A code identifying the reason of the leave of absence for the employee.</td>
</tr>
<tr>
<td>PEBEMPL_LOA_BEG_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>LEAVE OF ABSENCE BEGIN DATE: The begin date of a leave of absence for the employee.</td>
</tr>
<tr>
<td>PEBEMPL_LOA_END_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>LEAVE OF ABSENCE END DATE: The anticipated end date for an employees leave of absence.</td>
</tr>
<tr>
<td>PEBEMPL_TREA_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>YES</td>
<td>TERMINATION REASON CODE: Code used to identify the reason of termination.</td>
</tr>
<tr>
<td>PEBEMPL_TERM_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>TERMINATION DATE: Termination date of the employee.</td>
</tr>
</tbody>
</table>
### Table: PEBEMPL

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PEBEMPL_I9_FORM_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>I9 FORM INDICATOR: Indicates whether or not immigration information has been received.</td>
</tr>
<tr>
<td>PEBEMPL_I9_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>I9 DATE: The date immigration information was received.</td>
</tr>
<tr>
<td>PEBEMPL_I9_EXPIRE_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>I9 EXPIRATION DATE: The date the immigration information expires.</td>
</tr>
<tr>
<td>PEBEMPL.getActivity_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>ACTIVITY_DATE: Date when the record was inserted or last modified.</td>
</tr>
<tr>
<td>PEBEMPL_WKPR_CODE</td>
<td>VARCHAR2</td>
<td>4</td>
<td>YES</td>
<td>WKPR_CODE: Work Period Code</td>
</tr>
<tr>
<td>PEBEMPL_FLSA_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>NO</td>
<td>FLSA INDICATOR: Used for Fair Labor Overtime compensation. (C)ash, (A)ccrual or (N)o FLSA.</td>
</tr>
<tr>
<td>PEBEMPL_DAYS_IN_CANADA</td>
<td>NUMBER</td>
<td>3</td>
<td>YES</td>
<td>DAYS IN CANADA: Number of days employee worked in Canada.</td>
</tr>
<tr>
<td>PEBEMPL_1042_RECIPIENT_CD</td>
<td>VARCHAR2</td>
<td>2</td>
<td>YES</td>
<td>1042 RECIPIENT CODE: Identifies the type of individual who is receiving this 1042 statement.</td>
</tr>
</tbody>
</table>

### Target Application Information:

**Target Application Name:** Banner

**Target Application Description:** Banner person college will be updated in this interface

**Database Name:** Banner

**Table Name:** SORPCOL

### Table: SORPCOL

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SORPCOL_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number of person.</td>
</tr>
<tr>
<td>SORPCOL_SBGI_CODE</td>
<td>VARCHAR2</td>
<td>6</td>
<td>NO</td>
<td>This field identifies the prior college code of the prospect/applicant.</td>
</tr>
<tr>
<td>SORPCOL_TRANS_RECV_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>This field identifies the date an academic transcript was received from the prior college.</td>
</tr>
<tr>
<td>SORPCOL_TRANS_REV_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>This field identifies the date an academic transcript was reviewed by the institution.</td>
</tr>
<tr>
<td>SORPCOL_OFFICIAL_TRANS</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>This field indicates whether the transcript received was an official transcript. Y denotes yes.</td>
</tr>
<tr>
<td>SORPCOL_ADMR_CODE</td>
<td>VARCHAR2</td>
<td>4</td>
<td>YES</td>
<td>This field identifies the admission checklist request item code which may be cross referenced, ie. receipt of transcript here updates admissions checklist received date.</td>
</tr>
<tr>
<td>SORPCOL_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>This field identifies the last date the SORPCOL record was updated.</td>
</tr>
</tbody>
</table>

### Target Application Information:

**Target Application Name:** Banner

**Target Application Description:** Banner person education table will be updated in this interface

**Database Name:** Banner

**Table Name:** SORDEGR

### Table: SORDEGR

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SORDEGRL_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number of person.</td>
</tr>
</tbody>
</table>
| SORDEGR_SBGI_CODE                 | VARCHAR2     | 6     | NO       | This field identifies the prior college code of
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SORDEGR_DEGC_CODE</td>
<td>VARCHAR2</td>
<td>6</td>
<td>YES</td>
<td>This field identifies the degree received at the prior college.</td>
</tr>
<tr>
<td>SORDEGR_DEGR_SEQ_NO</td>
<td>NUMBER</td>
<td>2</td>
<td>NO</td>
<td>This field is a unique sequence number assigned to the prior college degree.</td>
</tr>
<tr>
<td>SORDEGR_ATTEND_FROM</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>This field represents the first date of attendance at the prior college.</td>
</tr>
<tr>
<td>SORDEGR_ATTEND_TO</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>This field represents the last date of attendance at the prior college.</td>
</tr>
<tr>
<td>SORDEGR_HOURS_TRANSFERRED</td>
<td>NUMBER</td>
<td>5</td>
<td>YES</td>
<td>This field indicates the total number hours transferred from the prior college. This field is informational and does not update the transfer GPA in Academic History.</td>
</tr>
<tr>
<td>SORDEGR_GPA_TRANSFERRED</td>
<td>NUMBER</td>
<td>10</td>
<td>YES</td>
<td>This field identifies the transfer GPA. This is informational and does not update the transfer GPA in Academic History.</td>
</tr>
<tr>
<td>SORDEGR_DEGC_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>Prior college degree date</td>
</tr>
<tr>
<td>SORDEGR_DEGC_YEAR</td>
<td>VARCHAR2</td>
<td>4</td>
<td>YES</td>
<td>Prior college degree year</td>
</tr>
<tr>
<td>SORDEGR_COLN_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>YES</td>
<td>Prior college code.</td>
</tr>
<tr>
<td>SORDEGR_HONR_CODE</td>
<td>VARCHAR2</td>
<td>6</td>
<td>YES</td>
<td>Honor code</td>
</tr>
<tr>
<td>SORDEGR_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>This field identifies the last date the SORDEGR record was updated.</td>
</tr>
<tr>
<td>SORDEGR_TERM_DEGREE</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Terminal degree indicator</td>
</tr>
<tr>
<td>SORDEGR_EGOL_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>SORDEGR_PRIMARY_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td></td>
</tr>
</tbody>
</table>

**Target Application Name:** Banner

**Target Application Description:** Banner person education table will be updated in this interface

**Target Data Structure:**
(repeat for each table or filename related to this interface)

**Database Name:** Banner
**Table Name:** SORMAJR

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SORMAJR_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number of person.</td>
</tr>
<tr>
<td>SORMAJR_SBGI_CODE</td>
<td>VARCHAR2</td>
<td>6</td>
<td>NO</td>
<td>Prior college code</td>
</tr>
<tr>
<td>SORMAJR_DEGC_CODE</td>
<td>VARCHAR2</td>
<td>6</td>
<td>YES</td>
<td>Prior college degree code</td>
</tr>
<tr>
<td>SORMAJR_DEGR_SEQ_NO</td>
<td>NUMBER</td>
<td>2</td>
<td>NO</td>
<td>Prior college degree seq no</td>
</tr>
<tr>
<td>SORMAJR_MAJR_CODE_MAJOR</td>
<td>VARCHAR2</td>
<td>4</td>
<td>NO</td>
<td>Major code</td>
</tr>
<tr>
<td>SORMAJR_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>Activity date</td>
</tr>
</tbody>
</table>

**Target Application Name:** Banner

**Target Application Description:** Banner person education table will be updated in this interface

**Target Data Structure:**
(repeat for each table or filename related to this interface)

**Database Name:** Banner
**Table Name:** SORMINR

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SORMINR_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number of person.</td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Length</td>
<td>Required</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------</td>
<td>--------</td>
<td>----------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td>SORMINR_SBGI_CODE</td>
<td>VARCHAR2</td>
<td>6</td>
<td>NO</td>
<td>Prior college code</td>
</tr>
<tr>
<td>SORMINR_DEGC_CODE</td>
<td>VARCHAR2</td>
<td>6</td>
<td>YES</td>
<td>Prior college degree code</td>
</tr>
<tr>
<td>SORMINR_DEGR_SEQ_NO</td>
<td>NUMBER</td>
<td>2</td>
<td>NO</td>
<td>Prior college degree seq no</td>
</tr>
<tr>
<td>SORMINR_MAJR_CODE_MINOR</td>
<td>VARCHAR2</td>
<td>4</td>
<td>NO</td>
<td>Minor code</td>
</tr>
<tr>
<td>SORMINR_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>Activity date</td>
</tr>
</tbody>
</table>
Target Application Name: Banner
Target Application Description: Banner person emergency contact table will be updated in this interface

Target Data Structure: (repeat for each table or filename related to this interface)

<table>
<thead>
<tr>
<th>Database Name: Banner</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table Name: SPREMRG</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPREMRG_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number of person.</td>
</tr>
<tr>
<td>SPREMRG_PRIORITY</td>
<td>VARCHAR2</td>
<td>1</td>
<td>NO</td>
<td>Priority indicator associated with emergency address of person record.</td>
</tr>
<tr>
<td>SPREMRG_LAST_NAME</td>
<td>VARCHAR2</td>
<td>25</td>
<td>NO</td>
<td>Last name of person associated with emergency address information.</td>
</tr>
<tr>
<td>SPREMRG_FIRST_NAME</td>
<td>VARCHAR2</td>
<td>15</td>
<td>NO</td>
<td>First name of person associated with emergency address information.</td>
</tr>
<tr>
<td>SPREMRG_MI</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Middle name of person associated with emergency address information.</td>
</tr>
<tr>
<td>SPREMRG_STREET_LINE1</td>
<td>VARCHAR2</td>
<td>30</td>
<td>YES</td>
<td>Line one of emergency address.</td>
</tr>
<tr>
<td>SPREMRG_STREET_LINE2</td>
<td>VARCHAR2</td>
<td>30</td>
<td>YES</td>
<td>Line two of emergency address.</td>
</tr>
<tr>
<td>SPREMRG_STREET_LINE3</td>
<td>VARCHAR2</td>
<td>30</td>
<td>YES</td>
<td>Line three of emergency address.</td>
</tr>
<tr>
<td>SPREMRG_CITY</td>
<td>VARCHAR2</td>
<td>20</td>
<td>YES</td>
<td>City associated with emergency address.</td>
</tr>
<tr>
<td>SPREMRG_STAT_CODE</td>
<td>VARCHAR2</td>
<td>3</td>
<td>YES</td>
<td>State associated with emergency address.</td>
</tr>
<tr>
<td>SPREMRG_NATN_CODE</td>
<td>VARCHAR2</td>
<td>5</td>
<td>YES</td>
<td>Nation/country associated with emergency address.</td>
</tr>
<tr>
<td>SPREMRG ZIP</td>
<td>VARCHAR2</td>
<td>10</td>
<td>YES</td>
<td>Zip code associated with emergency address.</td>
</tr>
<tr>
<td>SPREMRG_PHONE_AREA</td>
<td>VARCHAR2</td>
<td>3</td>
<td>YES</td>
<td>Area code of phone number associated with emergency address.</td>
</tr>
<tr>
<td>SPREMRG_PHONE_NUMBER</td>
<td>VARCHAR2</td>
<td>7</td>
<td>YES</td>
<td>Phone number associated with emergency address.</td>
</tr>
<tr>
<td>SPREMRG_PHONE_EXT</td>
<td>VARCHAR2</td>
<td>4</td>
<td>YES</td>
<td>Extension of phone number associated with emergency address.</td>
</tr>
<tr>
<td>SPREMRG_REL_T_CODE</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>This field identifies the relationship between emergency contact person and the person record.</td>
</tr>
<tr>
<td>SPREMRG_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>Most current date that the record was created or changed. This field does not display on screen.</td>
</tr>
<tr>
<td>SPREMRG_ATYP_CODE</td>
<td>TYPE</td>
<td>2</td>
<td>YES</td>
<td>The address type associated with the emergency address</td>
</tr>
</tbody>
</table>
**Target Application Name:** Banner  
**Target Application Description:** Banner person skill table will be updated in this interface

**Target Data Structure:** (repeat for each table or filename related to this interface)  
**Database Name:** Banner  
**Table Name:** PPRSKIL

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field Type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPRSKIL_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number of person.</td>
</tr>
<tr>
<td>PPRSKIL_SKIL_CODE</td>
<td>VARCHAR2</td>
<td>6</td>
<td>NO</td>
<td>SKILL CODE: A code identifying the type of skill. The code must be previously defined on the Skill Codes Rule Form (PTRSKIL). If the Level Required Indicator equals Y then the Level must be entered on the form.</td>
</tr>
<tr>
<td>PPRSKIL_SKLV_CODE</td>
<td>VARCHAR2</td>
<td>3</td>
<td>YES</td>
<td>SKILL LEVEL CODE: A code identifying the level of expertise achieved within the skill. The code must be previously defined on the Skill Levels Rule Form (PTRSKLV). The description of the skill level will default onto the form.</td>
</tr>
<tr>
<td>PPRSKIL_COMMENTS</td>
<td>VARCHAR2</td>
<td>60</td>
<td>YES</td>
<td>COMMENTS: Free Comments related to the skill.</td>
</tr>
<tr>
<td>PPRSKIL_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>ACTIVITY DATE: Date of last activity (insert or update) on this record.</td>
</tr>
<tr>
<td>PPRSKIL_START_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>START DATE: Date the skill was learned.</td>
</tr>
<tr>
<td>PPRSKIL_END_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>END DATE: Date the skill was ended.</td>
</tr>
<tr>
<td>PPRSKIL_LAST_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>DATE LAST USED: Date the skill was last used.</td>
</tr>
<tr>
<td>PPRSKIL_READ_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Indicates if person can read this skill.</td>
</tr>
<tr>
<td>PPRSKIL_WRITE_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Indicates if person can write this skill.</td>
</tr>
<tr>
<td>PPRSKIL_SPEAK_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Indicates if person can speak this language skill.</td>
</tr>
<tr>
<td>PPRSKIL_TRANS_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Indicates if person can translate this language skill.</td>
</tr>
<tr>
<td>PPRSKIL_TEACH_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Indicates if person can teach this language skill.</td>
</tr>
<tr>
<td>PPRSKIL_NATIVE_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>Indicates native language for person.</td>
</tr>
</tbody>
</table>
**Target Application Name:** Banner

**Target Application Description:** Banner person driver’s license table will be updated in this interface

**Target Data Structure:**
(repeat for each table or filename related to this interface)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPRDLIC_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number of person.</td>
</tr>
<tr>
<td>PPRDLIC_DRIVER_LICENSE</td>
<td>VARCHAR2</td>
<td>20</td>
<td>NO</td>
<td>Driver license number</td>
</tr>
<tr>
<td>Updated by NESSIE, viewed in NESSIE and ECOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPRDLIC_ISSUE_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>Driver license start date</td>
</tr>
<tr>
<td>Updated by NESSIE, viewed in NESSIE and ECOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPRDLIC_SKIL_CODE</td>
<td>VARCHAR2</td>
<td>6</td>
<td>YES</td>
<td></td>
</tr>
<tr>
<td>Updated by NESSIE, viewed in NESSIE and ECOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPRDLIC_STATUS</td>
<td>VARCHAR2</td>
<td>6</td>
<td>YES</td>
<td>Driver licenses status; system defined menu</td>
</tr>
<tr>
<td>Updated by NESSIE, viewed in NESSIE and ECOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPRDLIC_EXPIRE_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>Driver license end date</td>
</tr>
<tr>
<td>Updated by NESSIE, viewed in NESSIE and ECOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPRDLIC_STATE_CODE_DRIVER</td>
<td>VARCHAR2</td>
<td>2</td>
<td>YES</td>
<td>Driver license issuing state</td>
</tr>
<tr>
<td>Updated by NESSIE, viewed in NESSIE and ECOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPRDLIC_NATION_CODE_DRIVER</td>
<td>VARCHAR2</td>
<td>30</td>
<td>YES</td>
<td>Driver license issuing nation</td>
</tr>
<tr>
<td>Updated by NESSIE, viewed in NESSIE and ECOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PPRDLIC_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>ACTIVITY DATE: Date of last activity (insert or update) on this record.</td>
</tr>
</tbody>
</table>
**Target Application Name:** Banner  
**Target Application Description:** Banner person prior history table will be updated in this interface

**Target Data Structure:**  
(repeat for each table or filename related to this interface)  

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPREXPE_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number of person.</td>
</tr>
<tr>
<td>PPREXPE_POSN_TITLE</td>
<td>VARCHAR2</td>
<td>30</td>
<td>YES</td>
<td>POSITION TITLE: The title of the position occupied while obtaining this experience.</td>
</tr>
<tr>
<td>PPREXPE_EMTY_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>YES</td>
<td>EMPLOYER TYPE CODE: A code identifying the employer type. This type code must be previously defined on the Employer Type Validation Form (PTVEMTY).</td>
</tr>
<tr>
<td>PPREXPE_EMPR_NAME</td>
<td>VARCHAR2</td>
<td>30</td>
<td>YES</td>
<td>EMPLOYER NAME: The name of the previous or current employer at which this experience was obtained.</td>
</tr>
<tr>
<td>PPREXPE_START_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>START DATE: The starting date of the period of time during which this experience was obtained.</td>
</tr>
<tr>
<td>PPREXPE_END_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>END DATE: The end date for the period of time during which this experience was obtained.</td>
</tr>
<tr>
<td>PPREXPE_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>ACTIVITY DATE: Date of last activity (insert or update) on this record.</td>
</tr>
<tr>
<td>PPREXPE_CURR_SALARY</td>
<td>NUMBER</td>
<td>11</td>
<td>YES</td>
<td>CURRENT SALARY: Current salary at this position.</td>
</tr>
<tr>
<td>PPREXPE_CURR_HOURLY</td>
<td>NUMBER</td>
<td>11</td>
<td>YES</td>
<td>CURRENT HOURLY RATE: Current hourly rate at this position.</td>
</tr>
<tr>
<td>PPREXPE_REASON_LEFT</td>
<td>VARCHAR2</td>
<td>60</td>
<td>YES</td>
<td>REASON EMPLOYEE LEFT: The reason the employee left that position.</td>
</tr>
<tr>
<td>PPREXPE_EMPS_SUPERVISE</td>
<td>NUMBER</td>
<td>4</td>
<td>YES</td>
<td>NUMBER OF EMPLOYEES SUPERVISED: Number of employees applicant supervised.</td>
</tr>
<tr>
<td>PPREXPE_SUPER_NAME</td>
<td>VARCHAR2</td>
<td>60</td>
<td>YES</td>
<td>SUPERVISOR'S NAME: Name of applicant's supervisor.</td>
</tr>
<tr>
<td>PPREXPE_SUPER_PHONE_AREA</td>
<td>VARCHAR2</td>
<td>3</td>
<td>YES</td>
<td>SUPERVISOR'S AREA CODE: Supervisors area code.</td>
</tr>
<tr>
<td>PPREXPE_SUPER_PHONE_NUMBER</td>
<td>VARCHAR2</td>
<td>7</td>
<td>YES</td>
<td>SUPERVISOR'S PHONE NUMBER: Supervisors phone number.</td>
</tr>
<tr>
<td>PPREXPE_SUPER_PHONE_EXT</td>
<td>VARCHAR2</td>
<td>4</td>
<td>YES</td>
<td>SUPERVISOR'S EXTENTION: Supervisors extension.</td>
</tr>
<tr>
<td>PPREXPE_CONTACT_IND</td>
<td>VARCHAR2</td>
<td>1</td>
<td>YES</td>
<td>SUPERVISOR'S CONTACT IND: Indicates if supervisor maybe contacted.</td>
</tr>
<tr>
<td>PPREXPE_SUPER_COMMENT</td>
<td>LONG</td>
<td>0</td>
<td>YES</td>
<td>SUPERVISOR'S COMMENT: Any comments the supervisor may have given.</td>
</tr>
<tr>
<td>PPREXPE_POSN_DUTIES</td>
<td>VARCHAR2</td>
<td>255</td>
<td>YES</td>
<td>Position duties</td>
</tr>
<tr>
<td>PPREXPE_STAT_CODE</td>
<td>VARCHAR2</td>
<td>3</td>
<td>YES</td>
<td>State/Province of previous employer.</td>
</tr>
<tr>
<td>PPREXPE_NATN_CODE</td>
<td>VARCHAR2</td>
<td>5</td>
<td>YES</td>
<td>Nation of previous employer.</td>
</tr>
<tr>
<td>Target Field Name</td>
<td>Field type</td>
<td>Width</td>
<td>Nullable</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------</td>
<td>-------</td>
<td>----------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PPRCERT_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number of person.</td>
</tr>
<tr>
<td>PPRCERT_CERT_CODE</td>
<td>VARCHAR2</td>
<td>4</td>
<td>NO</td>
<td>CERTIFICATION CODE: 4 character code identifying the certification or license. The certification code must be previously defined on the Certification Code Rule Form (PPRCERT).</td>
</tr>
<tr>
<td>PPRCERT_CERT_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>CERTIFICATION DATE: The date the certification or license was obtained. The Date Req field on the Certification Code Rule Form (PPRCERT) controls whether this field must be entered.</td>
</tr>
<tr>
<td>PPRCERT_LIC_CERT_STATUS</td>
<td>VARCHAR2</td>
<td>10</td>
<td>YES</td>
<td>CERTIFICATION or LICENSE STATUS: Status of license (e.g., in effect, suspended, expired). Status should previously be defined on a validation form.</td>
</tr>
<tr>
<td>PPRCERT_NEXT_CERT_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>NEXT CERTIFICATION DATE: The date on which the employee is scheduled to be re-certified</td>
</tr>
<tr>
<td>PPRCERT_EXPIRE_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>CERTIFICATION EXPIRATION DATE: The date the certification or license will expire.</td>
</tr>
<tr>
<td>PPRCERT_CERT_NO</td>
<td>VARCHAR2</td>
<td>15</td>
<td>YES</td>
<td>CERTIFICATION NUMBER: The number of the certification or license if applicable.</td>
</tr>
<tr>
<td>PPRCERT_STAT_CODE</td>
<td>VARCHAR2</td>
<td>3</td>
<td>YES</td>
<td>STATE OF ISSUE: State issuing the certification or license. Pull from list of valid state codes.</td>
</tr>
<tr>
<td>PPRCERT_NATN_CODE</td>
<td>VARCHAR2</td>
<td>5</td>
<td>YES</td>
<td>NATION OF ISSUE: Country issuing the certification or license. Pull from list of valid nation codes.</td>
</tr>
<tr>
<td>COMMENT</td>
<td>VARCHAR2</td>
<td>??</td>
<td>YES</td>
<td>Used to store IssuingAgency information</td>
</tr>
<tr>
<td>PPRCERT_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>ACTIVITY DATE: Date of last activity (insert or update) on this record.</td>
</tr>
<tr>
<td>Target Application Name:</td>
<td>Banner</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Target Application Description:</td>
<td>Banner person endorsement table will be updated in this interface</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Target Data Structure:</td>
<td>(repeat for each table or filename related to this interface)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Database Name:</td>
<td>Banner</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Table Name:</td>
<td>PPRENDS</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPRENDS_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number of person.</td>
</tr>
<tr>
<td>PPRENDS_CERT_CODE</td>
<td>VARCHAR2</td>
<td>4</td>
<td>NO</td>
<td>CERTIFICATION CODE: The code identifying the certification previously defined on the Certification Code Rule Form (PTRCERT).</td>
</tr>
<tr>
<td>PPRENDS_ENDS_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>NO</td>
<td>ENDORSEMENT CODE: A code identifying who or what organization endorsed the certification. The description will default onto the form. The Endorsement Code must be prev. defined on the Endorsement Code Validation Form (PTVENDS).</td>
</tr>
<tr>
<td>PPRENDS_ENDS_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>ENDORSEMENT DATE: The date the endorsement was given.</td>
</tr>
<tr>
<td>PPRENS_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>ACTIVITY DATE: Date of last activity (insert or update) on this record.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Target Application Name:</th>
<th>Banner</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Application Description:</td>
<td>Banner person publication table will be updated in this interface</td>
</tr>
<tr>
<td>Target Data Structure:</td>
<td>(repeat for each table or filename related to this interface)</td>
</tr>
<tr>
<td>Database Name:</td>
<td>Banner</td>
</tr>
<tr>
<td>Table Name:</td>
<td>PPRPUBL</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Target Field Name</th>
<th>Field type</th>
<th>Width</th>
<th>Nullable</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPRPUBL_PIDM</td>
<td>NUMBER</td>
<td>8</td>
<td>NO</td>
<td>Internal identification number of person.</td>
</tr>
<tr>
<td>PPRPUBL_PUBT_CODE</td>
<td>VARCHAR2</td>
<td>2</td>
<td>NO</td>
<td>PUBLICATION CODE: A code identifying the type of publication. The code must be previously defined on the Publication Code Validation Form (PTVPUBT).</td>
</tr>
<tr>
<td>PPRPUBL_TITLE</td>
<td>VARCHAR2</td>
<td>60</td>
<td>NO</td>
<td>PUBLICATION TITLE: The title of the publication.</td>
</tr>
<tr>
<td>PPRPUBL_PUBLISHER</td>
<td>VARCHAR2</td>
<td>60</td>
<td>YES</td>
<td>PUBLISHER: The publisher of the publication.</td>
</tr>
<tr>
<td>PPRPUBL_PUBL_DATE</td>
<td>DATE</td>
<td>7</td>
<td>YES</td>
<td>PUBLICATION DATE: The date the publication was published.</td>
</tr>
<tr>
<td>PPRPUBL_ACTIVITY_DATE</td>
<td>DATE</td>
<td>7</td>
<td>NO</td>
<td>Publication record activity date.</td>
</tr>
<tr>
<td>PPRPUBL_VOLUME</td>
<td>VARCHAR2</td>
<td>25</td>
<td>YES</td>
<td>PUBLICATION VOLUME NUMBER: The volume number of the publication.</td>
</tr>
<tr>
<td>PPRPUBL_ISSUE</td>
<td>VARCHAR2</td>
<td>40</td>
<td>YES</td>
<td>PUBLICATION ISSUE NUMBER: The issue number of the publication.</td>
</tr>
<tr>
<td>PPRPUBL_PAGES</td>
<td>VARCHAR2</td>
<td>25</td>
<td>YES</td>
<td>PUBLICATION PAGE RANGE: Range of pages of publication.</td>
</tr>
</tbody>
</table>
1.3. List the interfaces that take data changes from the application named in the template to other existing applications.

[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.]

N/A. The interfaces for Banner online users to create or change basic person and basic employee information will not be available during phase 1 (i.e. December 2001 - June 2002) to ensure data consistency between Paymaster and Banner.

List of interfaces: XXXXXXXX, XXXXXXXX

### 1.3.1. Source application details:

<table>
<thead>
<tr>
<th>Source Application Name:</th>
<th>This is the application named in the template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Application Description:</td>
<td>Description already given for the Application named in the template.</td>
</tr>
<tr>
<td>Source Data Structure: (repeat for each table or filename related to this interface)</td>
<td>Database Name: XXXXXXXX</td>
</tr>
<tr>
<td></td>
<td>Table Name: XXXXXXXX</td>
</tr>
<tr>
<td></td>
<td>File Name: XXXXXXXX</td>
</tr>
<tr>
<td>Source Field Name</td>
<td>Field type</td>
</tr>
<tr>
<td>Example LAST-NAME-10</td>
<td>CHAR</td>
</tr>
</tbody>
</table>

### 1.3.2. Target application details:

<table>
<thead>
<tr>
<th>Target Application Name:</th>
<th>XXXXXXXX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Application Description:</td>
<td>XXXX XXXX (Brief description)</td>
</tr>
<tr>
<td>Target Data Structure: (repeat for each table or filename related to this interface)</td>
<td>Database Name: XXXXXXXX</td>
</tr>
<tr>
<td></td>
<td>Table Name: XXXXXXXX</td>
</tr>
<tr>
<td></td>
<td>File Name: XXXXXXXX</td>
</tr>
<tr>
<td>Target Field Name</td>
<td>Field type</td>
</tr>
<tr>
<td>Example TPERSON.LAST_NAME</td>
<td>VARCHAR</td>
</tr>
</tbody>
</table>
2. **Step 2**

Describe and define Banner 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 AITS integration staff. **This section should be completed prior to integration design meetings with AITS integration staff.**

2.1. Describe the proposed business processes that the application named in the template will supports, 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:** XXXXXXXXX

**Timeline:** XXXXXXXX

**Flow:** XXXXXXXXX

2.2. **What Banner data will this application require?**

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

N/A

2.3. **What data will Banner require from the application named in this template?**

[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.]

N/A
3. Step 3
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 AITS integration staff.

3.1. Application 1: Banner
- Banner must consume the basic person and basic employee synchronization messages produced by Paymaster to keep its data current.
- If errors occur during the consumption of a BasicPerson or BasicEmployee synchronization message, Banner must produce a CoreMessaging-Sync-Error-Sync message indicating that an error occurred.
- Banner must consume “Attachment” requests (create, update, delete) from Nessie and produce the appropriate replies to keep Banner data current.
- Banner must consume “Attachment” query requests and provide the appropriate replies to allow Nessie update information via its user interfaces.
- Banner must consume validation list queries and provide the appropriate list based on the list name and list filter provided in the query.
- When attachment information changes (created, updated or deleted) Banner must produce a sync message indicating the change.
- Banner must consume InstitutionalIdentity sync messages.

3.2. Application 2: Paymaster
- Paymaster must produce synchronization messages to keep Banner up-to-date whenever it has change(s) in the basic person and basic employee information.

3.3. Application 3: Nessie
- Nessie must produce query request message to request “Attachment” information from Banner.
- Nessie will produce attachment requests to create, update and delete attachment data in Banner.

3.4 Application 4: Nessie Gateway
- The Nessie gateway will produce query request messages to request “Attachment” information from Banner.
- For new employees, Nessie must store the attachment information until the new employees are added to Banner in the nightly Paymaster synchronization process. Nessie Gateway must subscribe to Paymaster new person synchronization message and must, at that point, issue the person attachment information create requests for Banner after receiving the University identity data from the BasicPerson sync messages. Nessie Gateway must handle the replies from Banner.

3.5 Application 5: PANDA
- PANDA will produce query request messages to request “Attachment” information from Banner.
4. Step 4
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 AITS integration staff.

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

SctSegments and UiSegments

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 SCT and University of Illinois 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.]

- BasicPerson
- Name
- Address
- Phone
- Email
- BasicEmployee
- EmploymentDates
- EmploymentLeave
- EmploymentTermination
- LightweightPerson
- UnknownPerson
- Education
- EmergencyContact
- HonorAward
- Language
- LicenseCertification
- Publication
- WorkHistory
- DriversLicense
- List

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

<table>
<thead>
<tr>
<th>File</th>
<th>Name</th>
<th>Proposed Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
5. Step 5

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 AITS integration staff.

5.1. Existing Messages

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

3. Person-BasicPerson-Create-Sync
4. Person-BasicPerson-Delete-Sync
5. Person-BasicPerson-Update-Sync
6. Employee-BasicEmployee-Create-Sync
7. Employee-BasicEmployee-Delete-Sync
8. Employee-BasicEmployee-Update-Sync
9. CoreMessaging-Generic-Response-Reply
10. CoreMessaging-Sync-Error-Sync
11. Person-Education-Create-Request
12. Person-Education-Create-Sync
13. Person-Education-Delete-Request
14. Person-Education-Delete-Sync
15. Person-Education-Provide-Reply
16. Person-Education-Query-Request
17. Person-Education-Update-Request
18. Person-Education-Update-Sync
19. Person-EmergencyContact-Create-Request
20. Person-EmergencyContact-Create-Sync
21. Person-EmergencyContact-Delete-Request
22. Person-EmergencyContact-Delete-Sync
23. Person-EmergencyContact-Provide-Reply
24. Person-EmergencyContact-Query-Request
25. Person-EmergencyContact-Update-Request
26. Person-EmergencyContact-Update-Sync
27. Person-HonorAward-Create-Request
28. Person-HonorAward-Create-Sync
29. Person-HonorAward-Delete-Request
30. Person-HonorAward-Delete-Sync
31. Person-HonorAward-Provide-Reply
32. Person-HonorAward-Query-Request
33. Person-HonorAward-Update-Request
34. Person-HonorAward-Update-Sync
35. Person-Language-Create-Request
36. Person-Language-Delete-Request
37. Person-Language-Create-Sync
38. Person-Language-Delete-Sync
39. Person-Language-Provide-Reply
40. Person-Language-Query-Request
41. Person-Language-Update-Requested
42. Person-Language-Update-Sync
43. Person-LicenseCertification-Create-Request
44. Person-LicenseCertification-Delete-Request
45. Person-LicenseCertification-Create-Sync
46. Person-LicenseCertification-Delete-Sync
47. Person-LicenseCertification-Provide-Reply
48. Person-LicenseCertification-Query-Request
49. Person-LicenseCertification-Update-Request
50. Person-LicenseCertification-Update-Sync
51. Person-Publication-Create-Request
52. Person-Publication-Create-Sync
53. Person-Publication-Delete-Request
54. Person-Publication-Delete-Sync
55. Person-Publication-Provide-Reply
56. Person-Publication-Query-Request
57. Person-Publication-Update-Request
58. Person-Publication-Update-Sync
59. Person-WorkHistory-Create-Request
60. Person-WorkHistory-Delete-Request
61. Person-WorkHistory-Create-Sync
62. Person-WorkHistory-Delete-Sync
63. Person-WorkHistory-Provide-Reply
64. Person-WorkHistory-Query-Request
65. Person-WorkHistory-Update-Request
66. Person-WorkHistory-Update-Sync
67. Person-DriversLicense-Create-Request
68. Person-DriversLicense-Delete-Request
69. Person-DriversLicense-Create-Sync
70. Person-DriversLicense-Delete-Sync
71. Person-DriversLicense-Provide-Reply
72. Person-DriversLicense-Query-Request
73. Person-DriversLicense-Update-Request
74. Person-DriversLicense-Update-Sync
75. Validation-List-Query-Request
76. Validation-List-Provide-Reply

5.2. Proposed Messages

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

1. None
6. Step 6
Name the existing messaging components that will be used or define the new messaging components 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 AITS integration staff.

6.1. Existing Messaging Components
[List existing components to be used in this interface or indicate that none are available for use.]

6.2. New Messaging Components

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Status</th>
<th>Responsible Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nessie Gateway</td>
<td>Gateway</td>
<td>planned</td>
<td>UO-HR/AITS</td>
</tr>
<tr>
<td>Nessie Newhire</td>
<td>Application</td>
<td>Planned</td>
<td>UO-HR/AITS</td>
</tr>
<tr>
<td>Nessie</td>
<td>Application</td>
<td>In progress</td>
<td>UO-HR/AITS</td>
</tr>
<tr>
<td>Banner Gateway</td>
<td>Gateway</td>
<td>In progress</td>
<td>SCT</td>
</tr>
<tr>
<td>Paymaster Gateway</td>
<td>Gateway</td>
<td>planned</td>
<td>AITS</td>
</tr>
<tr>
<td>PANDA</td>
<td>Application</td>
<td>Planned</td>
<td>UO-HR/AITS</td>
</tr>
<tr>
<td>ICard Gateway</td>
<td>Gateway</td>
<td>Planned</td>
<td>OBFS/AITS</td>
</tr>
</tbody>
</table>

7. Step 7
For the messaging component(s) of the application named in this template, list the new 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 AITS integration staff.

7.1. Messages

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.]

1. **Generic-Response-Reply**  ([Message DTD](#) | Sample Message)

   The Banner gateway may provide this reply message to Nessie or the Nessie gateway in response to a create, update or delete request. It XML format is:

   ```
   <!ELEMENT CoreMessagingGenericResponse (ControlAreaReply)>
   Banner will use the ControlAreaReply to indicate the success or failure of the message. Any errors encountered during the consumption of a message should be included in the Result/Error element of the ControlAreaReply. All Generic-Response-Reply messages should include the status (success or failure) in the Result element of the ControlAreaReply.
   ```

2. **EmergencyContact-Provide-Reply**

   The Banner gateway must provide this reply message to an application that produces an EmergencyContact-Query-Request.

   The EmergencyContact object(s) in the DataArea of the Provide message will be mapped from Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
</table>

Page 81 of 119
3. **EmergencyContact-Create-Sync**

   This message must be produced whenever an EmergencyContact record is created in Banner. This includes EmergencyContact records that are created via the Banner application itself. Refer to the EmergencyContact-Provide-Reply message production for Table-Message mapping information.

4. **EmergencyContact-Update-Sync**

   This message must be produced whenever an EmergencyContact record is updated in Banner. This includes EmergencyContact records that are updated via the Banner application itself. The Update-Sync must include both the new data (the EmergencyContact information after the update) in the DataArea/NewData/EmergencyContact element and the old EmergencyContact information (an image of the EmergencyContact prior to the update) in the DataArea/Baseline/EmergencyContact element. Refer to the EmergencyContact-Provide-Reply message production for Table-Message mapping information.

5. **EmergencyContact-Delete-Sync**

   This message must be produced whenever an EmergencyContact record is deleted from Banner. This includes EmergencyContact records that are deleted via the Banner application itself. Refer to the EmergencyContact-Provide-Reply message production for Table-Message mapping information.

6. **Education-Provide-Reply**

   The Banner gateway must provide this reply message to an application that produces an Education-Query-Request.
The Education object(s) in the DataArea of the Provide message will be mapped from Banner tables as follows:

**Note:** Since an Education record can have multiple Degrees, the production logic needs to retrieve all degrees for the person associated with the current education record. Also, since a Degree can have multiple Majors and Minors, the production logic will have to retrieve all Majors for the Education record for the current Degree as well as all Minors for the current Degree.

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>OwnerId</td>
<td>SPRIDEN</td>
<td>SPRIDEN_ID FROM SRIDEN TABLE WHERE SPRIDEN_ID = OwnerId AND SPRIDEN_CHANGE_IND IS BLANK.</td>
</tr>
</tbody>
</table>

**Institution**

| InstitutionName   | STVSBGI      | STVSBGI_DESC WHERE STVSBGI_CODE = SORPCOL_SBG DESC WHERE SORPCOL_SBG CODE = |

**Degree**

| DegreeName        | STVDEGC      | STVDEGC_DESC WHERE SORDEGR_DEGC_CODE = STVDEGC_CODE |
| DegreeDate        | SORDEGR      | SORDEGR_DEGC_DATE |
| CreditEarned      | SORDEGR      | SORDEGR_HOURS_TRANSFERRED |
| GPA               | SORDEGR      | SORDEGR_GPA_TRANSFERRED |
| Major             | STVMAJR      | STVMAJR_DESC WHERE STVMAJR_CODE = SORMAJR_MAJR_CODE_MAJOR AND SORDEGR_CODE = SORMAJR_DEGC_CODE AND SORDEGR_PIDM = SPRIDEN_PIDM |
| Minor             | STVMAJR      | STVMAJOR_DESC WHERE STVMAJOR_CODE = SORMINR_MAJR_CODE_MINOR AND SORDEGR_CODE = SORMINR_DEGC_CODE AND SORDEGR_PIDM = SPRIDEN_PIDM |

**Attendance**

| StartDate         | SORDEGR      | SORDEGR_ATTEND_FROM |
| EndDate           | SORDEGR      | SORDEGR_ATTEND_TO   |
| officialTranscrip | SORPCOL      | SORPCOL_OFFICIAL_TRANS (Yes or No) |

7. **Education-Create-Sync**

This message must be produced whenever an Education record is created in Banner. This includes Education records that are created via the Banner application itself. Refer to the Education-Provide-Reply message production for Table-Message mapping information.

8. **Education-Update-Sync**

This message must be produced whenever an Education record is updated in Banner. This includes Education records that are updated via the Banner application itself. The Update-Sync must include both the new data (the Education information after the update) in the
DataArea/NewData/Education element and the old Education information (an image of the Education prior to the update) in the DataArea/Baseline/Education element. Refer to the Education-Provide-Reply message production for Table-Message mapping information.

9. Education-Delete-Sync
This message must be produced whenever an Education record is deleted from Banner. This includes Education records that are deleted via the Banner application itself. Refer to the Education-Provide-Reply message production for Table-Message mapping information.

10. HonorAward-Provide-Reply
The Banner gateway must provide this reply message to an application that produces an HonorAward-Query-Request.

The HonorAward object(s) in the DataArea of the Provide message will be mapped from Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>ownerId</td>
<td>SPRIDEN</td>
<td>SPRIDEN_ID FROM SRIDEN TABLE WHERE SPRIDEN_ID = OwnerId AND SPRIDEN_CHANGE_IND IS BLANK.</td>
</tr>
<tr>
<td>Institution</td>
<td>PPRHNAW</td>
<td>PPRHNAW_AWARD_ORG</td>
</tr>
<tr>
<td>HonorAwardTitle</td>
<td>PPRHNAW</td>
<td>PPRHNAW_TITLE</td>
</tr>
<tr>
<td>HonorAwardDate</td>
<td>PPRHNAW</td>
<td>PPRHNAW_AWARD_DATE</td>
</tr>
<tr>
<td>ExpirationDate</td>
<td>PPRHNAW</td>
<td>PPRHNAW_AWARD_DATE_EXPIRE</td>
</tr>
</tbody>
</table>

11. HonorAward-Create-Sync
This message must be produced whenever an HonorAward record is created in Banner. This includes HonorAward records that are created via the Banner application itself. Refer to the HonorAward-Provide-Reply message production for Table-Message mapping information.

12. HonorAward-Update-Sync
This message must be produced whenever an HonorAward record is updated in Banner. This includes HonorAward records that are updated via the Banner application itself. The Update-Sync must include both the new data (the HonorAward information after the update) in the DataArea/NewData/HonorAward element and the old HonorAward information (an image of the HonorAward prior to the update) in the DataArea/Baseline/HonorAward element. Refer to the HonorAward-Provide-Reply message production for Table-Message mapping information.

13. HonorAward-Delete-Sync
This message must be produced whenever an HonorAward record is deleted from Banner. This includes HonorAward records that are deleted via the Banner application itself. Refer to the HonorAward-Provide-Reply message production for Table-Message mapping information.

14. Language-Provide-Reply
The Banner gateway must provide this reply message to an application that produces a Language-Query-Request.
The Language object(s) in the DataArea of the Provide message will be mapped from Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>OwnerId</td>
<td>SPRIDEN</td>
<td>SPRIDEN_ID FROM SPRIDEN TABLE SPRIDEN_ID = OwnerId AND WHERE SPRIDEN_CHANGE_IND IS BLANK.</td>
</tr>
<tr>
<td>LanguageName</td>
<td>PTRSKIL</td>
<td>PTRSKIL_DESC WHERE PPRSKIL_CODE = PPRSKIL_CODE</td>
</tr>
<tr>
<td>Read</td>
<td>PPRSKIL</td>
<td>PPRSKIL_READ_IND</td>
</tr>
<tr>
<td>Write</td>
<td>PPRSKIL</td>
<td>PPRSKIL_WRITE_IND</td>
</tr>
<tr>
<td>Speak</td>
<td>PPRSKIL</td>
<td>PPRSKIL_SPEAK_IND</td>
</tr>
<tr>
<td>Translate</td>
<td>PPRSKIL</td>
<td>PPRSKIL_TRANS_IND</td>
</tr>
<tr>
<td>Teach</td>
<td>PPRSKIL</td>
<td>PPRSKIL_TEACH_IND</td>
</tr>
<tr>
<td>Native</td>
<td>PPRSKIL</td>
<td>PPRSKIL_NATIVE_IND</td>
</tr>
</tbody>
</table>

15. **Language-Create-Sync**
   This message must be produced whenever a Language record is created in Banner. This includes Language records that are created via the Banner application itself. Refer to the Language-Provide-Reply message production for Table-Message mapping information.

16. **Language-Update-Sync**
   This message must be produced whenever a Language record is updated in Banner. This includes Language records that are updated via the Banner application itself. The Update-Sync must include both the new data (the Language information after the update) in the DataArea/NewData/Language element and the old Language information (an image of the Language prior to the update) in the DataArea/Baseline/Language element. Refer to the Language-Provide-Reply message production for Table-Message mapping information.

17. **Language-Delete-Sync**
   This message must be produced whenever a Language record is deleted from Banner. This includes Language records that are deleted via the Banner application itself. Refer to the Language-Provide-Reply message production for Table-Message mapping information.

18. **LicenseCertification-Provide-Reply**
   The Banner gateway must provide this reply message to an application that produces a LicenseCertification-Query-Request.

   The LicenseCertification object(s) in the DataArea of the Provide message will be mapped from Banner tables as follows:

   **Note:** Since a LicenseCertification record can have multiple Endorsements, the production logic will have to get all Endorsements for each LicenseCertification for the person.

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>OwnerId</td>
<td>SPRIDEN</td>
<td>SPRIDEN_ID FROM SPRIDEN TABLE WHERE SPRIDEN_ID = OwnerId AND</td>
</tr>
</tbody>
</table>
19. LicenseCertification-Create-Sync
This message must be produced whenever a LicenseCertification record is created in Banner. This includes LicenseCertification records that are created via the Banner application itself. Refer to the LicenseCertification-Provide-Reply message production for Table-Message mapping information.

20. LicenseCertification-Update-Sync
This message must be produced whenever a LicenseCertification record is updated in Banner. This includes LicenseCertification records that are updated via the Banner application itself. The Update-Sync must include both the new data (the LicenseCertification information after the update) in the DataArea/NewData/LicenseCertification element and the old LicenseCertification information (an image of the LicenseCertification prior to the update) in the DataArea/Baseline/LicenseCertification element. Refer to the LicenseCertification-Provide-Reply message production for Table-Message mapping information.

21. LicenseCertification-Delete-Sync
This message must be produced whenever a LicenseCertification record is deleted from Banner. This includes LicenseCertification records that are deleted via the Banner application itself. Refer to the LicenseCertification-Provide-Reply message production for Table-Message mapping information.

22. Publication-Provide-Reply
The Banner gateway must provide this reply message to an application that produces a Publication-Query-Request.
The Publication object(s) in the DataArea of the Provide message will be mapped from Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>OwnerId</td>
<td>SPRIDEN</td>
<td>SPRIDEN_ID FROM SRIDEN TABLE WHERE SPRIDEN_ID = OwnerId AND SPRIDEN_CHANGE_IND IS BLANK.</td>
</tr>
<tr>
<td>Publisher</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PublisherName</td>
<td>PPRPUBL</td>
<td>PPRPUBL_PUBLISHER</td>
</tr>
<tr>
<td>PublicationTitle</td>
<td>PPRPUBL</td>
<td>PPRPUBL_TITLE</td>
</tr>
<tr>
<td>PublicationDate</td>
<td>PPRPUBL</td>
<td>PPRPUBL_PUB_DATE</td>
</tr>
<tr>
<td>VolumeNumber</td>
<td>PPRPUBL</td>
<td>PPRPUBL_VOLUME</td>
</tr>
<tr>
<td>IssueNumber</td>
<td>PPRPUBL</td>
<td>PPRPUBL_ISSUE</td>
</tr>
<tr>
<td>PageNumber</td>
<td>PPRPUBL</td>
<td>PPRPUBL_PAGES</td>
</tr>
<tr>
<td>PublicationType</td>
<td>PTVPUBT</td>
<td>PTVPUBT_DESC WHERE PTVPUBT_CODE = PPRPUBL_PUBT_CODE</td>
</tr>
</tbody>
</table>

23. Publication-Create-Sync
This message must be produced whenever a Publication record is created in Banner. This includes Publication records that are created via the Banner application itself. Refer to the Publication-Provide-Reply message production for Table-Message mapping information.

24. Publication-Update-Sync
This message must be produced whenever a Publication record is updated in Banner. This includes Publication records that are updated via the Banner application itself. The Update-Sync must include both the new data (the Publication information after the update) in the DataArea/NewData/Publication element and the old Publication information (an image of the Publication prior to the update) in the DataArea/Baseline/Publication element. Refer to the Publication-Provide-Reply message production for Table-Message mapping information.

25. Publication-Delete-Sync
This message must be produced whenever a Publication record is deleted from Banner. This includes Publication records that are deleted via the Banner application itself. Refer to the Publication-Provide-Reply message production for Table-Message mapping information.

26. WorkHistory-Provide-Reply
The Banner gateway must provide this reply message to an application that produces a WorkHistory-Query-Request.

The WorkHistory object(s) in the DataArea of the Provide message will be mapped from Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>OwnerId</td>
<td>SPRIDEN</td>
<td>SPRIDEN_ID FROM SRIDEN TABLE WHERE SPRIDEN_ID = OwnerId AND</td>
</tr>
</tbody>
</table>
Employer

| EmployerName | PPREXPE | PPREXPE_EMPR_NAME |
| State        | STVSTAT | STVSTAT_DESC WHERE STVSTAT_CODE = PPREXPE_STAT_CODE |
| Nation       | STVNATN | STVNATN_NATION WHERE STVNATN_CODE = PPREXPE_NATN_CODE |

Supervisor

| SupervisorName | PPREXPE | PPREXPE_SUPER_NAME |
| PhoneArea     | PPREXPE | PPREXPE_SUPER_PHONE_AREA |
| PhoneNumber   | PPREXPE | PPREXPE_SUPER_PHONE_NUMBER |
| PhoneExtension| PPREXPE | PPREXPE_SUPER_PHONE_EXT |

Position

| PositionTitle | PPREXPE | PPREXPE_POSN_TITLE |
| StartDate     | PPREXPE | PPREXPE_START_DATE |
| EndDate       | PPREXPE | PPREXPE_END_DATE |
| Salary        | PPREXPE | PPREXPE_CURR_SALARY |
| HourlyRate    | PPREXPE | PPREXPE_CURR_HOURLY |
| PositionDuty  | PPREXPE | PPREXPE_POSN_DUTIES |
| ReasonLeft    | PPREXPE | PPREXPE_REASON_LEFT |

27. WorkHistory-Create-Sync

This message must be produced whenever a WorkHistory record is created in Banner. This includes WorkHistory records that are created via the Banner application itself. Refer to the WorkHistory-Provide-Reply message production for Table-Message mapping information.

28. WorkHistory-Update-Sync

This message must be produced whenever a WorkHistory record is updated in Banner. This includes WorkHistory records that are updated via the Banner application itself. The Update-Sync must include both the new data (the WorkHistory information after the update) in the DataArea/NewData/WorkHistory element and the old WorkHistory information (an image of the WorkHistory prior to the update) in the DataArea/Baseline/WorkHistory element. Refer to the WorkHistory-Provide-Reply message production for Table-Message mapping information.

29. WorkHistory-Delete-Sync

This message must be produced whenever a WorkHistory record is deleted from Banner. This includes WorkHistory records that are deleted via the Banner application itself. Refer to the WorkHistory-Provide-Reply message production for Table-Message mapping information.

30. DriversLicense-Provide-Reply

The Banner gateway must provide this reply message to an application that produces a DriversLicense-Query-Request.

The DriversLicense object(s) in the DataArea of the Provide message will be mapped from Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>ownerId</td>
<td>SPRIDEN</td>
<td>SPRIDEN_ID FROM SRIDEN TABLE WHERE SPRIDEN_ID = OwnerId AND</td>
</tr>
</tbody>
</table>
31. DriversLicense-Create-Sync
This message must be produced whenever a DriversLicense record is created in Banner. This includes DriversLicense records that are created via the Banner application itself. Refer to the DriversLicense-Provide-Reply message production for Table-Message mapping information.

32. DriversLicense-Update-Sync
This message must be produced whenever a DriversLicense record is updated in Banner. This includes DriversLicense records that are updated via the Banner application itself. The Update-Sync must include both the new data (the DriversLicense information after the update) in the DataArea/NewData/DriversLicense element and the old DriversLicense information (an image of the DriversLicense prior to the update) in the DataArea/Baseline/DriversLicense element. Refer to the DriversLicense-Provide-Reply message production for Table-Message mapping information.

33. DriversLicense-Delete-Sync
This message must be produced whenever a DriversLicense record is deleted from Banner. This includes DriversLicense records that are deleted via the Banner application itself. Refer to the DriversLicense-Provide-Reply message production for Table-Message mapping information.

34. Visa-Provide-Reply
The Banner gateway must provide this reply message to an application that produces a Visa-Query-Request.

The Visa object(s) in the DataArea of the Provide message will be mapped from Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>ownerId</td>
<td>SPRIDEN</td>
<td>SPRIDEN_ID FROM SRIDEN TABLE WHERE SPRIDEN_ID = OwnerId AND SPRIDEN_CHANGE_IND IS BLANK.</td>
</tr>
</tbody>
</table>
35. List-Provide-Reply

The Banner gateway must provide this reply message to an application that produces a List-Query-Request. The List-Query-Request will contain information regarding the list type and any filters that need to be applied to the result from the query. The List-Provide-Reply will include a list of descriptions from a “valid values” table that corresponds to the list type in the query request. There will be several lists that Banner needs to provide. All lists should be provided in alphabetic order. They are:

<table>
<thead>
<tr>
<th>List Name</th>
<th>Filters</th>
<th>Element in the Provide Message DataArea</th>
<th>Banner Table</th>
<th>Banner Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address Types</td>
<td>List/ListItem/ListItemValue</td>
<td>STVATYP</td>
<td>STVATYP</td>
<td>STVATYP_DESC</td>
</tr>
<tr>
<td>Gender</td>
<td>List/ListItem/ListItemValue</td>
<td>Static</td>
<td>STVMRTL</td>
<td>STVMRTL_DESC</td>
</tr>
<tr>
<td>Marital Status</td>
<td>List/ListItem/ListItemValue</td>
<td>STVMRTL</td>
<td>STVMRTL</td>
<td>STVMRTL_DESC</td>
</tr>
<tr>
<td>Ethnicity</td>
<td>List/ListItem/ListItemValue</td>
<td>STVETHN</td>
<td>STVETHN</td>
<td>STVETHN_DESC</td>
</tr>
<tr>
<td>State/Province</td>
<td>States only</td>
<td>List/ListItem/ListItemValue</td>
<td>STVSTAT</td>
<td>STVSTAT_DESC</td>
</tr>
<tr>
<td></td>
<td>Provinces Only</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Illinois Counties</td>
<td>List/ListItem/ListItemValue</td>
<td>STVCNTY</td>
<td>STVCNTY</td>
<td>STVCNTY_DESC</td>
</tr>
<tr>
<td>Nation</td>
<td>List/ListItem/ListItemValue</td>
<td>STVMAJR</td>
<td>STVMAJR</td>
<td>STVMAJR_DESC</td>
</tr>
<tr>
<td>Educational</td>
<td>Institutions for a State</td>
<td>List/ListItem/ListItemValue</td>
<td>STVMAJR</td>
<td>STVMAJR_DESC</td>
</tr>
<tr>
<td>Institutions</td>
<td>Institutions outside the US</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Education Degrees</td>
<td>Associate degrees</td>
<td>List/ListItem/ListItemValue</td>
<td>STVDEGC</td>
<td>STVDEGC_DESC</td>
</tr>
<tr>
<td></td>
<td>Bachelor degrees</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Masters degrees</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Doctoral degrees</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Education Major</td>
<td>STVMAJR_VALID_MAJOR_IND=&quot;Y&quot;</td>
<td>List/ListItem/ListItemValue</td>
<td>STVMAJR</td>
<td>STVMAJR_DESC</td>
</tr>
<tr>
<td>Education Minors</td>
<td>STVMAJR_VALID_MINOR_IND=&quot;Y&quot;</td>
<td>List/ListItem/ListItemValue</td>
<td>STVMAJR</td>
<td>STVMAJR_DESC</td>
</tr>
<tr>
<td>Relationship to</td>
<td>List/ListItem/ListItemValue</td>
<td>STVRELT</td>
<td>STVRELT</td>
<td>STVRELT_DESC</td>
</tr>
<tr>
<td>Employee</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Languages</td>
<td>PTRSKIL_CODE LIKE &quot;LA&quot;</td>
<td>List/ListItem/ListItemValue</td>
<td>PTRSKIL</td>
<td>PTRSKIL_DESC</td>
</tr>
<tr>
<td>Licenses</td>
<td>PTRCERT_CODE LIKE &quot;L&quot;</td>
<td>List/ListItem/ListItemValue</td>
<td>PTRCERT</td>
<td>PTRCERT_DESC</td>
</tr>
<tr>
<td>Certifications</td>
<td>PTRCERT_CODE LIKE &quot;C&quot;</td>
<td>List/ListItem/ListItemValue</td>
<td>PTRCERT</td>
<td>PTRCERT_DESC</td>
</tr>
<tr>
<td>License/Certificate Status</td>
<td>PTRCERT_CODE LIKE &quot;C&quot;</td>
<td>List/ListItem/ListItemValue</td>
<td>PTRCERT</td>
<td>PTRCERT_DESC</td>
</tr>
<tr>
<td>Endorsing Agencies</td>
<td>List/ListItem/ListItemValue</td>
<td>PTVENDS</td>
<td>PTVENDS</td>
<td>PTVENDS_DESC</td>
</tr>
</tbody>
</table>

35. List-Provide-Reply

The Banner gateway must provide this reply message to an application that produces a List-Query-Request. The List-Query-Request will contain information regarding the list type and any filters that need to be applied to the result from the query. The List-Provide-Reply will include a list of descriptions from a “valid values” table that corresponds to the list type in the query request. There will be several lists that Banner needs to provide. All lists should be provided in alphabetic order. They are:

<table>
<thead>
<tr>
<th>List Name</th>
<th>Filters</th>
<th>Element in the Provide Message DataArea</th>
<th>Banner Table</th>
<th>Banner Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address Types</td>
<td>List/ListItem/ListItemValue</td>
<td>STVATYP</td>
<td>STVATYP_DESC</td>
<td></td>
</tr>
<tr>
<td>Gender</td>
<td>List/ListItem/ListItemValue</td>
<td>Static</td>
<td>STVMRTL_DESC</td>
<td></td>
</tr>
<tr>
<td>Marital Status</td>
<td>List/ListItem/ListItemValue</td>
<td>STVMRTL</td>
<td>STVMRTL_DESC</td>
<td></td>
</tr>
<tr>
<td>Ethnicity</td>
<td>List/ListItem/ListItemValue</td>
<td>STVETHN</td>
<td>STVETHN_DESC</td>
<td></td>
</tr>
<tr>
<td>State/Province</td>
<td>States only, Provinces Only</td>
<td>List/ListItem/ListItemValue</td>
<td>STVSTAT, STVSTAT_DESC</td>
<td></td>
</tr>
<tr>
<td>Illinois Counties</td>
<td>List/ListItem/ListItemValue</td>
<td>STVCNTY</td>
<td>STVCNTY_DESC</td>
<td></td>
</tr>
<tr>
<td>Nation</td>
<td>List/ListItem/ListItemValue</td>
<td>STVMAJR</td>
<td>STVMAJR_DESC</td>
<td></td>
</tr>
<tr>
<td>Educational Institutions</td>
<td>Institutions for a State, Institutions outside the US</td>
<td>List/ListItem/ListItemValue</td>
<td>STVMAJR, STVMAJR_DESC</td>
<td></td>
</tr>
<tr>
<td>Education Degrees</td>
<td>Associate degrees, Bachelor degrees, Masters degrees, Doctoral degrees</td>
<td>List/ListItem/ListItemValue</td>
<td>STVDEGC, STVDEGC_DESC</td>
<td></td>
</tr>
<tr>
<td>Education Major</td>
<td>STVMAJR_VALID_MAJOR_IND=&quot;Y&quot;</td>
<td>List/ListItem/ListItemValue</td>
<td>STVMAJR, STVMAJR_DESC</td>
<td></td>
</tr>
<tr>
<td>Education Minors</td>
<td>STVMAJR_VALID_MINOR_IND=&quot;Y&quot;</td>
<td>List/ListItem/ListItemValue</td>
<td>STVMAJR, STVMAJR_DESC</td>
<td></td>
</tr>
<tr>
<td>Relationship to Employee</td>
<td>List/ListItem/ListItemValue</td>
<td>STVRELT</td>
<td>STVRELT_DESC</td>
<td></td>
</tr>
<tr>
<td>Languages</td>
<td>PTRSKIL_CODE LIKE &quot;LA&quot;</td>
<td>List/ListItem/ListItemValue</td>
<td>PTRSKIL, PTRSKIL_DESC</td>
<td></td>
</tr>
<tr>
<td>Licenses</td>
<td>PTRCERT_CODE LIKE &quot;L&quot;</td>
<td>List/ListItem/ListItemValue</td>
<td>PTRCERT, PTRCERT_DESC</td>
<td></td>
</tr>
<tr>
<td>Certifications</td>
<td>PTRCERT_CODE LIKE &quot;C&quot;</td>
<td>List/ListItem/ListItemValue</td>
<td>PTRCERT, PTRCERT_DESC</td>
<td></td>
</tr>
<tr>
<td>License/Certificate Status</td>
<td>PTRCERT_CODE LIKE &quot;C&quot;</td>
<td>List/ListItem/ListItemValue</td>
<td>PTRCERT, PTRCERT_DESC</td>
<td></td>
</tr>
<tr>
<td>Endorsing Agencies</td>
<td>List/ListItem/ListItemValue</td>
<td>PTVENDS</td>
<td>PTVENDS_DESC</td>
<td></td>
</tr>
</tbody>
</table>
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.]

1. **Person-BasicPerson-Create-Sync** *(Message DTD | Sample Message)*
The Banner gateway will consume this synchronization message in the XML format specified by the corresponding message DTD.

**Data Area Element values** *(see SctSegments for all elements)*:
The DataArea will contain a BasicPerson to be Created in Banner.

**Consumption logic:**
If an error occurs while consuming this message, Banner should produce a Sync-Error-Sync message.

The BasicPerson object in the DataArea of the Create-Sync message should be inserted into Banner as considered appropriate by SCT. This includes using all existing person creation logic presently prescribed by SCT. This means that SCT should derive all subsequently derived Banner fields implied in this mapping.

Analysis at the University of Illinois indicates that this object might map to Banner as follows in the table below. SCT should consult with the University of Illinois if the mapping appears incorrect or incomplete to SCT.

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
</table>
| InstitutionalId   | SPRIDEN      | SPRIDEN_ID  
|                   |              | Note: The SPRIDEN_ID may contain the temporary ID (Banner ID) for the person. |
| Name              | SPRIDEN      | Value “Yes” where SPRIDEN_CHANGE_IND IS BLANK. |
|                   |              | SPRIDEN_LAST_NAME |
|                   | SPRIDEN      | SPRIDEN_FIRST_NAME |
|                   | SPRIDEN      | SPRIDEN_MI |
| Prefix            | SPBPERS      | SPBPERS_NAME_PREFIX |
| Suffix            | SPBPERS      | SPBPERS_NAME_SUFFIX |
| SocialSecurityNumber | SPBPERS   | SPBPERS_SSN |
| BirthDate         | SPBPERS      | SPBPERS_BIRTH_DATE |
| Gender            | SPBPERS      | SPBPERS_SEX |
| DeceasedDate      | SPBPERS      | SPBPERS_DEAD_DATE |
| Ethnicity         | SPBPERS      | SPBPERS_ETHN_CODE |
| Address (multiple)|              |       |
Review the phase I release of EnterpriseObjects.xml file for details on all valid values for each of the XML elements and attributes that will be delivered in message to Banner that Banner must support. Banner must accept all enterprise values for all fields, since the University is configuring Banner with enterprise values.

2. **Person-BasicPerson-Delete-Sync** *(Message DTD | Sample Message)*
   The Banner gateway will consume this synchronization message in the XML format specified by the corresponding message DTD.

   **Data Area Element values** (see SctSegments for all elements):
   The DataArea will contain a BasicPerson to be Deleted in Banner. The DeleteData/DeleteAction@type will take a value of either “Delete” or “Purge”.

   **Consumption logic:**
   If an error occurs while consuming this message, Banner should produce a Sync-Error-Sync message.

   The BasicPerson object in the DataArea of the Delete-Sync message should be marked for delete from Banner in the following Banner tables:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>occurrences)</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>SPRADDR</td>
<td>APRADDR_ATYP_CODE</td>
</tr>
<tr>
<td>Street1</td>
<td>SPRADDR</td>
<td>SPRADDR_STREET_LINE1</td>
</tr>
<tr>
<td>Street2</td>
<td>SPRADDR</td>
<td>SPRADDR_STREET_LINE2</td>
</tr>
<tr>
<td>Street3</td>
<td>SPRADDR</td>
<td>SPRADDR_STREET_LINE3</td>
</tr>
<tr>
<td>CityOrLocality</td>
<td>SPRADDR</td>
<td>SPRADDR_CITY</td>
</tr>
<tr>
<td>County</td>
<td>SPRADDR</td>
<td>SPRADDR_CNTY_CODE</td>
</tr>
<tr>
<td>State</td>
<td>SPRADDR</td>
<td>SPRADDR_STAT_CODE</td>
</tr>
<tr>
<td>Nation</td>
<td>SPRADDR</td>
<td>SPRADDR_NATN_CODE</td>
</tr>
<tr>
<td>Zip</td>
<td>SPRADDR</td>
<td>SPRADDR_ZIP</td>
</tr>
<tr>
<td>EffectiveDate</td>
<td>SPRADDR</td>
<td>SPRADDR_FROM_DATE</td>
</tr>
<tr>
<td>TerminationDate</td>
<td>SPRADDR</td>
<td>SPRADDR_TO_DATE</td>
</tr>
<tr>
<td>Phone</td>
<td></td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>SPRTELE</td>
<td>SPRTELE_TELE_CODE</td>
</tr>
<tr>
<td>PhoneAddressType</td>
<td>SPRTELE</td>
<td>SPRTELE_ATYP_CODE</td>
</tr>
<tr>
<td>CountryCode</td>
<td>SPRTELE</td>
<td>SPRTELE_INTL_ACCESS</td>
</tr>
<tr>
<td>PhoneArea</td>
<td>SPRTELE</td>
<td>SPRTELE_PHONE_AREA</td>
</tr>
<tr>
<td>PhoneNumber</td>
<td>SPRTELE</td>
<td>SPRTELE_PHONE_NUMBER</td>
</tr>
<tr>
<td>PhoneExtension</td>
<td>SPRTELE</td>
<td>SPRTELE_PHONE_EXTENSION</td>
</tr>
<tr>
<td>Email</td>
<td></td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>GOREMAL</td>
<td>GOREMAL_EMAL_CODE</td>
</tr>
<tr>
<td>status</td>
<td>GOREMAL</td>
<td>GOREMAL_STATUS_IND</td>
</tr>
<tr>
<td>preferred</td>
<td>GOREMAL</td>
<td>GOREMAL_PREFERRED_IND</td>
</tr>
<tr>
<td>EmailAdress</td>
<td>GOREMAL</td>
<td>GOREMAL_EMAIL_ADDRESS</td>
</tr>
</tbody>
</table>
3. **Person-BasicPerson-Update-Sync** *(Message DTD | Sample Message)*

The Banner gateway will consume this synchronization message in the XML format specified by the corresponding message DTD.

**Data Area Element values** (see **SctSegments** for all elements):

- **DataArea/NewData** will contain a BasicPerson object that has been changed.
- **DataArea/Baseline** will contain the a BasicPerson object that should match the current “Banner” image of the BasicPerson.

**Consumption logic:**

If an error occurs while consuming this message, Banner should produce a Sync-Error-Sync message.

Banner should always accept the DataArea/NewData/BasicPerson as the “authoritative” version of the object. Therefore, Banner must update the Banner tables with the BasicPerson from the Update-Sync message even if the DataArea/Baseline/BasicPerson does not match the current Banner image of the BasicPerson. However, it should produce a Sync-Error-Sync message indicating that it over-wrote Banner data with information from Paymaster since Paymaster is the authoritative source of BasicPerson and BasicEmployee information.

See **BasicPerson-Create-Sync** consumption logic for table Message-Banner mappings.

4. **Employee-BasicEmployee-Create-Sync** *(Message DTD | Sample Message)*

The Banner gateway will consume this synchronization message in the XML format specified by the corresponding message DTD.

**Data Area Element values** (see **SctSegments** for all elements):

**Consumption logic:**

If an error occurs while consuming this message, Banner should produce a Sync-Error-Sync message.

The BasicEmployee object in the DataArea of the Create-Sync message should inserted into Banner and mapped to Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group</td>
<td>PEBEMPL</td>
<td>PEBEMPL_EGRP_CODE</td>
</tr>
<tr>
<td>Status</td>
<td></td>
<td>PEBEMPL_EMPL_STATUS</td>
</tr>
<tr>
<td>Flsa</td>
<td></td>
<td>PEBEMPL_FLSA_IND</td>
</tr>
<tr>
<td>TimeStatus</td>
<td></td>
<td>PEBEMPL_INTERNAL_FT_PT_IND</td>
</tr>
<tr>
<td>PayCampus</td>
<td></td>
<td>PEBEMPL_DICD_CODE</td>
</tr>
</tbody>
</table>
| InstitutionalId         |              | USED TO GET PIDM  
                                  | PEBEMPL_PIDM |
|                         |              | PEBEMPL_PIDM TAKEN FROM SRIDEN TABLE WHERE 
                                  | SPRIDEN_ID = InstitutionalId AND 
                                  | SPRIDEN_CHANGE_IND IS 
                                  | BLANK. |
| HomeChartOfAccounts     |              | PEBEMPL_COAS_CODE_HOME                    |
5. **Employee-BasicEmployee-Delete-Sync** *(Message DTD | Sample Message)*

The Banner gateway will consume this synchronization message in the XML format specified by the corresponding message DTD.

**Data Area Element values** (see SctSegments for all elements):

**Consumption logic:**
If an error occurs while consuming this message, Banner should produce a Sync-Error-Sync message.

The BasicEmployee object in the DataArea of the Delete-Sync message should be marked for delete from Banner in the following Banner tables:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6. **Employee-BasicEmployee-Update-Sync** *(Message DTD | Sample Message)*

The Banner gateway will consume this synchronization message in the XML format specified by the corresponding message DTD.
Data Area Element values (see SctSegments for all elements):

Consumption logic:
If an error occurs while consuming this message, Banner should produce a Sync-Error-Sync message.

Banner should always accept the DataArea/NewData/BasicEmployee as the “authoritative” version of the object. Therefore, Banner must update the Banner tables with the BasicEmployee from the Update-Sync message even if the DataArea/Baseline/BasicEmployee does not match the current Banner image of the BasicEmployee. However, it should produce a Sync-Error-Sync message indicating that it over-wrote Banner data with information from Paymaster since Paymaster is the authoritative source of BasicPerson and BasicEmployee information.

See BasicPerson-Create-Sync consumption logic for table Message-Banner mappings.

7. InstitutionalIdentity-Update-Sync

Data Area Element values (see SctSegments for all elements)

Consumption logic:
The Banner gateway must consume this message when an InstitutionalIdentity is changed within the Icard system. It must use the information in this message to keep the SPRIDEN_ID in the SPRIDEN table up-to-date as InstitutionalIdentities change in the organization.

The InstitutionalIdentity object in the DataArea of the Sync message will be mapped to Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>InstitutionalIdentity</td>
<td>SPRIDEN</td>
<td>SPRIDEN_ID</td>
</tr>
<tr>
<td>InstitutionalId</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8. EmergencyContact-Create-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to create an EmergencyContact record in Banner for a given person (ownerId). Banner should retrieve the PIDM from the SPRIDEN table based on the ownerId associated to the EmergencyContact passed in. Then, using the PIDM, Banner should create the EmergencyContact record for the person in the appropriate Banner tables with information from the message.

After the create action has been completed, Banner should produce a Generic-Response-Reply message and return it to the requesting application. The status of the Create should be included in the Result element of the ControlAreaReply.
The EmergencyContact object in the DataArea of the Request message will be mapped to Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>OwnerId</td>
<td>SPREMRG</td>
<td>SPREMRG_PIDM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPRIDEN_PIDM TAKEN FROM SRIDEN TABLE WHERE SPRIDEN_ID = OwnerId AND SPRIDEN_CHANGE_IND IS BLANK.</td>
</tr>
<tr>
<td>Name</td>
<td>SPREMRG</td>
<td>SPREMRG_LAST_NAME</td>
</tr>
<tr>
<td>FirstName</td>
<td>SPREMRG</td>
<td>SPREMRG_FIRST_NAME</td>
</tr>
<tr>
<td>Middle Initial</td>
<td>SPREMRG</td>
<td>SPREMRG_MI</td>
</tr>
<tr>
<td>Phone</td>
<td>SPREMRG</td>
<td>SPREMRG_PHONE_AREA</td>
</tr>
<tr>
<td></td>
<td>SPREMRG</td>
<td>SPREMRG_PHONE_NUMBER</td>
</tr>
<tr>
<td></td>
<td>SPREMRG</td>
<td>SPREMRG_PHONE_EXT</td>
</tr>
<tr>
<td>Address</td>
<td>SPREMRG</td>
<td>SPREMRG_ATYP_CODE = STVATYP_CODE WHERE STVATYP_CODE = Type</td>
</tr>
<tr>
<td>Street1</td>
<td>SPREMRG</td>
<td>SPREMRG_STREET_LINE1</td>
</tr>
<tr>
<td>Street2</td>
<td>SPREMRG</td>
<td>SPREMRG_STREET_LINE2</td>
</tr>
<tr>
<td>Street3</td>
<td>SPREMRG</td>
<td>SPREMRG_STREET_LINE3</td>
</tr>
<tr>
<td>City</td>
<td>SPREMRG</td>
<td>SPREMRG_CITY</td>
</tr>
<tr>
<td>State</td>
<td>SPREMRG</td>
<td>SPREMRG_STAT_CODE</td>
</tr>
<tr>
<td>Zip</td>
<td>SPREMRG</td>
<td>SPREMRG_ZIP</td>
</tr>
<tr>
<td>Relationship</td>
<td>SPREMRG</td>
<td>SPREMRG_RELT_CODE</td>
</tr>
<tr>
<td>Priority</td>
<td>SPREMRG</td>
<td>SPREMRG_PRIORITY</td>
</tr>
</tbody>
</table>

9. EmergencyContact-Update-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to update an EmergencyContact record in Banner for a given person (ownerId). Banner should retrieve the PIDM from the SPRIDEN table based on the ownerId associated to the EmergencyContact passed in. Then, using the PIDM, Banner should update the appropriate Banner tables with the DataArea/NewData/EmergencyContact information contained in the message.

If the DataArea/Baseline/EmergencyContact does not match the current Banner image of the EmergencyContact, the update cannot proceed because this means the Banner data
has been updated from some source other than the requesting application and to allow
the update from the requesting application would mean over-writing those changes.

After the update action has been completed, Banner should produce a Generic-
Response-Reply message and return it to the requesting application. The status
(success or failure and any errors if failure) of the update should be included in the Result
element of the ControlAreaReply.

See the EmergencyContact-Create-Request consumption logic for Message-Table
mappings.

10. EmergencyContact-Delete-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to delete an
EmergencyContact record from Banner.

EmergencyContact information should be deleted from the following table(s):

11. EmergencyContact-Query-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to retrieve a
list of EmergencyContact records for a given person (LightweightPerson/InstitutionalId)
from Banner. Banner should retrieve the PIDM from the SPRIDEN table based on the
InstitutionalId associated to the LightweightPerson passed in. Then, using the PIDM,
Banner should retrieve all EmergencyContact rows for the person and return them in the
EmergencyContact-Provide-Reply message. The status (success or failure and any
errors if failure) of the query should be included in the Result element of the
ControlAreaReply in the Provide message.

See the EmergencyContact-Provide-Reply Production Logic for Message-Table
mappings for the provide message.

12. Education-Create-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to create an
Education record in Banner for a given person (ownerId). Banner should retrieve the
PIDM from the SPRIDEN table based on the ownerId associated to the
Education passed in. Then, using the PIDM, Banner should create the Education record
in the appropriate Banner tables with information from the message.

After the create action has been completed, Banner should produce a Generic-
Response-Reply message and return it to the requesting application. The status of the
Create should be included in the Result element of the ControlAreaReply.
The Education object in the DataArea of the Request message will be mapped to Banner tables as follows:

Note: Since there can be multiple Degrees for an Education record the consumption logic should insert multiple rows in the database for each Degree found in the Create request. Also, since there can be multiple Majors and Minors for a degree, those elements may have to be inserted multiple times for each Major/Minor found for each Degree.

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>OwnerId</td>
<td>SORDEGR</td>
<td>SORDEGR_PIDM</td>
</tr>
<tr>
<td></td>
<td>SORPCOL</td>
<td>SORPCOL_PIDM</td>
</tr>
<tr>
<td></td>
<td>SORMAJR</td>
<td>SORMAJR_PIDM</td>
</tr>
<tr>
<td></td>
<td>SORMINR</td>
<td>SORMINR_PIDM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPRI1DEN_PIDM TAKEN FROM SRI1DEN TABLE WHERE SPRI1DEN_ID = OwnerId AND SPRI1DEN_CHANGE_IND IS BLANK.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>InstitutionName STVSBGI STVSBGI_DESC WHERE STVSBGI_CODE = SORPCOL SBGI_CODE</td>
</tr>
<tr>
<td>Degree</td>
<td>SORDEGR</td>
<td>SORDEGR_DEGC_CODE = STVDEGC_CODE WHERE STVDEGC_DESC = DegreeName</td>
</tr>
<tr>
<td></td>
<td>SORPCOL</td>
<td>SORPCOL_ATTEND_FROM</td>
</tr>
<tr>
<td></td>
<td>SORMAJR</td>
<td>SORMAJR_MAJR_CODE_MAJOR = STVMAJR_CODE WHERE STVMAJR_DESC = Major</td>
</tr>
<tr>
<td></td>
<td>SORMINR</td>
<td>SORMINR_MAJR_CODE_MINOR = STVMAJOR_CODE WHERE STVMAJOR_DESC = Minor</td>
</tr>
<tr>
<td>Attendance</td>
<td>SORDEGR</td>
<td>SORDEGR_ATTEND_FROM</td>
</tr>
<tr>
<td></td>
<td>SORDEGR</td>
<td>SORDEGR_ATTEND_TO</td>
</tr>
<tr>
<td>officialTranscriptRecieved</td>
<td>SORPCOL</td>
<td>SORPCOL_OFFICIAL_TRANS (Y or N)</td>
</tr>
</tbody>
</table>

13. Education-Update-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to update an Education record in Banner for a given person (ownerId). Banner should retrieve the PIDM from the SPRIDEN table based on the ownerId associated to the Education passed in. Then, using the PIDM, Banner should update the appropriate Banner tables with the DataArea/NewData/Education information from the message.

If the DataArea/Baseline/Education does not match the current Banner image of the Education record, the update cannot proceed because this means the Banner data has been updated from some source other than the requesting application and to allow the update from the requesting application would mean over-writing those changes.

After the update action has been completed, Banner should produce a Generic-Response-Reply message and return it to the requesting application. The status (success or failure and any errors if failure) of the update should be included in the Result element of the ControlAreaReply.

See the Education-Create-Request consumption logic for Message-Table mappings.

14. Education-Delete-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:

15. Education-Query-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to retrieve a list of Education records for a given person (LightweightPerson/InstitutionalId) from Banner. Banner should retrieve the PIDM from the SPRIDEN table based on the InstitutionalId associated to the LightweightPerson passed in. Then, using the PIDM, Banner should retrieve all Education rows for the person and return them in the Education-Provide-Reply message. The status (success or failure and any errors if failure) of the query should be included in the Result element of the ControlAreaReply in the Provide message.

See the Education-Provide-Reply Production Logic for Message-Table mappings for the provide message.

16. HonorAward-Create-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to create an HonorAward record in Banner for a given person (ownerId). Banner should retrieve the PIDM from the SPRIDEN table based on the ownerId associated to the HonorAward passed in. Then, using the PIDM, Banner should create the HonorAward record in the appropriate Banner tables with information from the message.
After the create action has been completed, Banner should produce a Generic-
Response-Reply message and return it to the requesting application. The status of the
Create should be included in the Result element of the ControlAreaReply.

The HonorAward object in the DataArea of the Request message will be mapped to
Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>OwnerId</td>
<td>PPRHNAW</td>
<td>PPRHNAW_PIDM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPRIDEN_PIDM TAKEN FROM SRIDEN TABLE WHERE SPRIDEN_ID = OwnerId AND SPRIDEN_CHANGE_IND IS BLANK.</td>
</tr>
<tr>
<td>Institution</td>
<td>PPRHNAW</td>
<td>PPRHNAW_AWARD_ORG</td>
</tr>
<tr>
<td>InstitutionName</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HonorAwardTitle</td>
<td>PPRHNAW</td>
<td>PPRHNAW_TITLE</td>
</tr>
<tr>
<td>HonorAwardDate</td>
<td>PPRHNAW</td>
<td>PPRHNAW_AWARD_DATE</td>
</tr>
<tr>
<td>ExpirationDate</td>
<td>PPRHNAW</td>
<td>PPRHNAW_AWARD_DATE_EXPIRE</td>
</tr>
</tbody>
</table>

17. **HonorAward-Update-Request**

**Data Area Element values** (see SctSegments for all elements):

**Consumption logic:**
The Banner gateway must consume this request when an application wants to update an
HonorAward record in Banner for a given person (ownerId). Banner should retrieve the
PIDM from the SPRIDEN table based on the ownerId associated to the
HonorAward passed in. Then, using the PIDM, Banner should update the appropriate
Banner tables with the DataArea/NewData/HonorAward information from the message.

If the DataArea/Baseline/HonorAward does not match the current Banner image of the
HonorAward record, the update cannot proceed because this means the Banner data has
been updated from some source other than the requesting application and to allow the
update from the requesting application would mean over-writing those changes.

After the update action has been completed, Banner should produce a Generic-
Response-Reply message and return it to the requesting application. The status
(success or failure and any errors if failure) of the update should be included in the Result
element of the ControlAreaReply.

See the HonorAward-Create-Request consumption logic for Message-Table mappings.

18. **HonorAward-Delete-Request**

**Data Area Element values** (see SctSegments for all elements):

**Consumption logic:**

19. **HonorAward-Query-Request**
Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to retrieve a list of HonorAward records for a given person (LightweightPerson/InstitutionalId) from Banner. Banner should retrieve the PIDM from the SPRIDEN table based on the InstitutionalId associated to the LightweightPerson passed in. Then, using the PIDM, Banner should retrieve all HonorAward rows for the person and return them in the HonorAward-Provide-Reply message. The status (success or failure and any errors if failure) of the query should be included in the Result element of the ControlAreaReply in the Provide message.

See the HonorAward-Provide-Reply Production Logic for Message-Table mappings for the provide message.

20. Language-Create-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to create a Language record in Banner for a given person (ownerId). Banner should retrieve the PIDM from the SPRIDEN table based on the ownerId associated to the Language passed in. Then, using the PIDM, Banner should create the Language record in the appropriate Banner tables with information from the message.

After the create action has been completed, Banner should produce a Generic-Response-Reply message and return it to the requesting application. The status of the Create should be included in the Result element of the ControlAreaReply.

The Language object in the DataArea of the Request message will be mapped to Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>OwnerId</td>
<td>PPRSKIL</td>
<td>PPRSKIL_PIDM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPRIDEN_PIDM TAKEN FROM SPRIDEN TABLE WHERE SPRIDEN_ID = OwnerId AND SPRIDEN_CHANGE_IND IS BLANK.</td>
</tr>
<tr>
<td>LanguageName</td>
<td>PPRSKIL</td>
<td>PPRSKIL_CODE = PTRSKIL_CODE WHERE PTRSKIL_DESC = LanguageName</td>
</tr>
<tr>
<td>Read</td>
<td>PPRSKIL</td>
<td>PPRSKIL_READ_IND</td>
</tr>
<tr>
<td>Write</td>
<td>PPRSKIL</td>
<td>PPRSKIL_WRITE_IND</td>
</tr>
<tr>
<td>Speak</td>
<td>PPRSKIL</td>
<td>PPRSKIL_SPEAK_IND</td>
</tr>
<tr>
<td>Translate</td>
<td>PPRSKIL</td>
<td>PPRSKIL_TRANS_IND</td>
</tr>
<tr>
<td>Teach</td>
<td>PPRSKIL</td>
<td>PPRSKIL_TEACH_IND</td>
</tr>
<tr>
<td>Native</td>
<td>PPRSKIL</td>
<td>PPRSKIL_NATIVE_IND</td>
</tr>
</tbody>
</table>
21. Language-Update-Request

**Data Area Element values** (see SctSegments for all elements):

**Consumption logic:**
The Banner gateway must consume this request when an application wants to update a Language record in Banner for a given person (ownerId). Banner should retrieve the PIDM from the SPRIDEN table based on the ownerId associated to the Language passed in. Then, using the PIDM, Banner should update the appropriate Banner tables with the DataArea/NewData/Language information from the message.

If the DataArea/Baseline/Language does not match the current Banner image of the Language record, the update cannot proceed because this means the Banner data has been updated from some source other than the requesting application and to allow the update from the requesting application would mean over-writing those changes.

After the update action has been completed, Banner should produce a Generic-Response-Reply message and return it to the requesting application. The status (success or failure and any errors if failure) of the update should be included in the Result element of the ControlAreaReply.

See the Language-Create-Request consumption logic for Message-Table mappings.

22. Language-Delete-Request

**Data Area Element values** (see SctSegments for all elements):

**Consumption logic:**

23. Language-Query-Request

**Data Area Element values** (see SctSegments for all elements):

**Consumption logic:**
The Banner gateway must consume this request when an application wants to retrieve a list of Language records for a given person (LightweightPerson/InstitutionalId) from Banner. Banner should retrieve the PIDM from the SPRIDEN table based on the InstitutionalId associated to the LightweightPerson passed in. Then, using the PIDM, Banner should retrieve all Language rows for the person and return them in the Language-Provide-Reply message. The status (success or failure and any errors if failure) of the query should be included in the Result element of the ControlAreaReply in the Provide message.

See the Language-Provide-Reply Production Logic for Message-Table mappings for the provide message.

24. LicenseCertification-Create-Request

**Data Area Element values** (see SctSegments for all elements):

**Consumption logic:**
The Banner gateway must consume this request when an application wants to create a LicenseCertification record in Banner for a given person (ownerId). Banner should retrieve the PIDM from the SPRIDEN table based on the ownerId associated to the LicenseCertification passed in. Then, using the PIDM, Banner should create the LicenseCertification record in the appropriate Banner tables with information from the message.

After the create action has been completed, Banner should produce a Generic-Response-Reply message and return it to the requesting application. The status of the Create should be included in the Result element of the ControlAreaReply.

The LicenseCertification object in the DataArea of the Request message will be mapped to Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>OwnerId</td>
<td>PPRCERT</td>
<td>PPRCERT_PIDM</td>
</tr>
<tr>
<td></td>
<td>PPRENDS</td>
<td>PPRENDS_PIDM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPRIDEN_PIDM TAKEN FROM SPRIDEN TABLE WHERE SPRIDEN_ID = OwnerId AND SPRIDEN_CHANGE_IND IS BLANK.</td>
</tr>
<tr>
<td>IssuingAgency</td>
<td>PPRCERT</td>
<td>COMMENT COLUMN???</td>
</tr>
<tr>
<td>CertificationName</td>
<td>PPRCERT</td>
<td>PPRCERT_CERT_CODE = PTRCERT_CODE WHERE PTRCERT_DESC = CertificationName</td>
</tr>
<tr>
<td>ExpirationDate</td>
<td>PPRCERT</td>
<td>PPRCERT_EXPIRE_DATE</td>
</tr>
<tr>
<td>StateOfIssue</td>
<td>PPRCERT</td>
<td>PPRCERT_STAT_CODE = STVSTAT_CODE WHERE STVSTAT_DESC = StateOfIssue</td>
</tr>
<tr>
<td>CertificationDate</td>
<td>PPRCERT</td>
<td>PPRCERT_CERT_DATE</td>
</tr>
<tr>
<td>NextCertificationDate</td>
<td>PPRCERT</td>
<td>PPRCERT_NEXT_CERT_DATE</td>
</tr>
<tr>
<td>CertificationNumber</td>
<td>PPRCERT</td>
<td>PPRCERT_CERT_NO</td>
</tr>
<tr>
<td>CertificationStatus</td>
<td>PPRCERT</td>
<td>PPRCERT_CERT_CODE = PTVLCSV_CODE WHERE PTVLCSV_DESC = CertificationStatus</td>
</tr>
<tr>
<td>NationOfIssue</td>
<td>PPRCERT</td>
<td>PPRCERT_NATN_CODE = STVNATN_CODE WHERE STVNATN_NATION = NationOfIssue</td>
</tr>
<tr>
<td>Endorsement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Institution</td>
<td></td>
<td></td>
</tr>
<tr>
<td>InstitutionName</td>
<td>PPRENDS</td>
<td>PPRENDS_CODE = PTVENDS_CODE WHERE PTVENDSDESC = InstitutionName</td>
</tr>
<tr>
<td>EndorsementDate</td>
<td>PPRENDS</td>
<td>PPRENDS_ENDS_DATE</td>
</tr>
</tbody>
</table>
25. LicenseCertification-Update-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to update a LicenseCertification record in Banner for a given person (ownerId). Banner should retrieve the PIDM from the SPRIDEN table based on the ownerId associated to the LicenseCertification passed in. Then, using the PIDM, Banner should update the appropriate Banner tables with the DataArea/NewData/LicenseCertification information from the message.

If the DataArea/Baseline/LicenseCertification does not match the current Banner image of the LicenseCertification record, the update cannot proceed because this means the Banner data has been updated from some source other than the requesting application and to allow the update from the requesting application would mean over-writing those changes.

After the update action has been completed, Banner should produce a Generic-Response-Reply message and return it to the requesting application. The status (success or failure and any errors if failure) of the update should be included in the Result element of the ControlAreaReply.

See the LicenseCertification-Create-Request consumption logic for Message-Table mappings.

26. LicenseCertification-Delete-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:

27. LicenseCertification-Query-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to retrieve a list of LicenseCertification records for a given person (LightweightPerson/InstitutionalId) from Banner. Banner should retrieve the PIDM from the SPRIDEN table based on the InstitutionalId associated to the LightweightPerson passed in. Then, using the PIDM, Banner should retrieve all LicenseCertification rows for the person and return them in the LicenseCertification-Provide-Reply message. The status (success or failure and any errors if failure) of the query should be included in the Result element of the ControlAreaReply in the Provide message.

See the LicenseCertification-Provide-Reply Production Logic for Message-Table mappings for the provide message.

28. Publication-Create-Request
Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to create a Publication record in Banner for a given person (ownerId). Banner should retrieve the PIDM from the SPRIDEN table based on the ownerId associated to the Publication passed in. Then, using the PIDM, Banner should create the Publication record in the appropriate Banner tables with information from the message.

After the create action has been completed, Banner should produce a Generic-Response-Reply message and return it to the requesting application. The status of the Create should be included in the Result element of the ControlAreaReply.

The Publication object in the DataArea of the Request message will be mapped to Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>ownerId</td>
<td>PPRPUBL</td>
<td>PPRPUBL_PIDM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPRIDEN_PIDM TAKEN FROM SRIDEN TABLE WHERE SPRIDEN_ID = OwnerId AND SPRIDEN_CHANGE_IND IS BLANK.</td>
</tr>
<tr>
<td>Publisher</td>
<td>PPRPUBL</td>
<td>PPRPUBL_PUBLISHER</td>
</tr>
<tr>
<td>PublisherName</td>
<td>PPRPUBL</td>
<td>PPRPUBL_PUBLISHER</td>
</tr>
<tr>
<td>PublicationTitle</td>
<td>PPRPUBL</td>
<td>PPRPUBL_PUBLISHER</td>
</tr>
<tr>
<td>PublicationDate</td>
<td>PPRPUBL</td>
<td>PPRPUBL_PUBLISHER</td>
</tr>
<tr>
<td>VolumeNumber</td>
<td>PPRPUBL</td>
<td>PPRPUBL_VOLUME</td>
</tr>
<tr>
<td>IssueNumber</td>
<td>PPRPUBL</td>
<td>PPRPUBL_ISSUE</td>
</tr>
<tr>
<td>PageRange</td>
<td>PPRPUBL</td>
<td>PPRPUBL_PAGES</td>
</tr>
<tr>
<td>PublicationType</td>
<td>PPRPUBL</td>
<td>PPRPUBL_PUBT_CODE = PTVPUBT_CODE WHERE PTVPUBT_DESC = PublicationType</td>
</tr>
</tbody>
</table>

29. Publication-Update-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to update a Publication record in Banner for a given person (ownerId). Banner should retrieve the PIDM from the SPRIDEN table based on the ownerId associated to the Publication passed in. Then, using the PIDM, Banner should update the appropriate Banner tables with the DataArea/NewData/Publication information from the message.

If the DataArea/Baseline/Publication does not match the current Banner image of the Publication record, the update cannot proceed because this means the Banner data has
been updated from some source other than the requesting application and to allow the
update from the requesting application would mean over-writing those changes.

After the update action has been completed, Banner should produce a Generic-
Response-Reply message and return it to the requesting application. The status
(success or failure and any errors if failure) of the update should be included in the Result
element of the ControlAreaReply.

See the Publication-Create-Request consumption logic for Message-Table mappings.

30. Publication-Delete-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:

31. Publication-Query-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to retrieve a
list of Publication records for a given person (LightweightPerson/InstitutionalId) from
Banner. Banner should retrieve the PIDM from the SPRIDEN table based on the
InstitutionalId associated to the LightweightPerson passed in. Then, using the PIDM,
Banner should retrieve all Publication rows for the person and return them in the
Publication-Provide-Reply message. The status (success or failure and any errors if
failure) of the query should be included in the Result element of the ControlAreaReply in
the Provide message.

See the Publication-Provide-Reply Production Logic for Message-Table mappings for the
provide message.

32. WorkHistory-Create-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to create a
WorkHistory record in Banner for a given person (ownerId). Banner should retrieve the
PIDM from the SPRIDEN table based on the ownerId associated to the
WorkHistory passed in. Then, using the PIDM, Banner should create the WorkHistory
record in the appropriate Banner tables with information from the message.

After the create action has been completed, Banner should produce a Generic-
Response-Reply message and return it to the requesting application. The status of the
Create should be included in the Result element of the ControlAreaReply.

The WorkHistory object in the DataArea of the Request message will be mapped to
Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
</table>

Page 106 of 119
### WorkHistory-Update-Request

**Data Area Element values** (see SctSegments for all elements):

**Consumption logic:**

The Banner gateway must consume this request when an application wants to update a WorkHistory record in Banner for a given person (ownerId). Banner should retrieve the PIDM from the SPRIDEN table based on the ownerId associated to the WorkHistory passed in. Then, using the PIDM, Banner should update the appropriate Banner tables with the DataArea/NewData/WorkHistory information from the message.

If the DataArea/Baseline/WorkHistory does not match the current Banner image of the WorkHistory record, the update cannot proceed because this means the Banner data has been updated from some source other than the requesting application and to allow the update from the requesting application would mean over-writing those changes.

After the update action has been completed, Banner should produce a Generic-Response-Reply message and return it to the requesting application. The status (success or failure and any errors if failure) of the update should be included in the Result element of the ControlAreaReply.

See the WorkHistory-Create-Request consumption logic for Message-Table mappings.
34. **WorkHistory-Delete-Request**

**Data Area Element values** (see SctSegments for all elements):

**Consumption logic:**

35. **WorkHistory-Query-Request**

**Data Area Element values** (see SctSegments for all elements):

**Consumption logic:**

The Banner gateway must consume this request when an application wants to retrieve a list of WorkHistory records for a given person (LightweightPerson/InstitutionalId) from Banner. Banner should retrieve the PIDM from the SPRIDEN table based on the InstitutionalId associated to the LightweightPerson passed in. Then, using the PIDM, Banner should retrieve all WorkHistory rows for the person and return them in the WorkHistory-Provide-Reply message. The status (success or failure and any errors if failure) of the query should be included in the Result element of the ControlAreaReply in the Provide message.

See the WorkHistory-Provide-Reply Production Logic for Message-Table mappings for the provide message.

36. **DriversLicense-Create-Request**

**Data Area Element values** (see SctSegments for all elements):

**Consumption logic:**

The Banner gateway must consume this request when an application wants to create a DriversLicense record in Banner for a given person (ownerId). Banner should retrieve the PIDM from the SPRIDEN table based on the ownerId associated to the DriversLicense passed in. Then, using the PIDM, Banner should create the DriversLicense record in the appropriate Banner tables with information from the message.

After the create action has been completed, Banner should produce a Generic-Response-Reply message and return it to the requesting application. The status of the Create should be included in the Result element of the ControlAreaReply.

The DriversLicense object in the DataArea of the Request message will be mapped to Banner tables as follows:

<table>
<thead>
<tr>
<th>Element/Attribute</th>
<th>Banner Table</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>ownerId</td>
<td>PPRDLIC</td>
<td>PPRDLIC_PIDM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SPRIDEN_PIDM TAKEN FROM SRIDEN TABLE WHERE SPRIDEN_ID = OwnerId AND SPRIDEN_CHANGE_IND IS BLANK.</td>
</tr>
</tbody>
</table>
### 37. DriversLicense-Update-Request

**Data Area Element values** (see SctSegments for all elements):

#### Consumption logic:
The Banner gateway must consume this request when an application wants to update a DriversLicense record in Banner for a given person (ownerId). Banner should retrieve the PIDM from the SPRIDEN table based on the ownerId associated to the DriversLicense passed in. Then, using the PIDM, Banner should update the appropriate Banner tables with the DataArea/NewData/DriversLicense information from the message.

If the DataArea/Baseline/DriversLicense does not match the current Banner image of the DriversLicense record, the update cannot proceed because this means the Banner data has been updated from some source other than the requesting application and to allow the update from the requesting application would mean over-writing those changes.

After the update action has been completed, Banner should produce a Generic-Response-Reply message and return it to the requesting application. The status (success or failure and any errors if failure) of the update should be included in the Result element of the ControlAreaReply.

See the DriversLicense-Create-Request consumption logic for Message-Table mappings.

### 38. DriversLicense-Delete-Request

**Data Area Element values** (see SctSegments for all elements):

#### Consumption logic:

### 39. DriversLicense-Query-Request

**Data Area Element values** (see SctSegments for all elements):

#### Consumption logic:
The Banner gateway must consume this request when an application wants to retrieve a list of DriversLicense records for a given person (LightweightPerson/InstitutionalId) from
Banner. Banner should retrieve the PIDM from the SPRIDEN table based on the InstitutionalId associated to the LightweightPerson passed in. Then, using the PIDM, Banner should retrieve all DriversLicense rows for the person and return them in the DriversLicense-Provide-Reply message. The status (success or failure and any errors if failure) of the query should be included in the Result element of the ControlAreaReply in the Provide message.

See the DriversLicense-Provide-Reply Production Logic for Message-Table mappings for the provide message.

40. Visa-Query-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to retrieve a list of Visa records for a given person (LightweightPerson/InstitutionalId) from Banner. Banner should retrieve the PIDM from the SPRIDEN table based on the InstitutionalId associated to the LightweightPerson passed in. Then, using the PIDM, Banner should retrieve all Visa rows for the person and return them in the Visa-Provide-Reply message. The status (success or failure and any errors if failure) of the query should be included in the Result element of the ControlAreaReply in the Provide message.

See the Visa-Provide-Reply Production Logic for Message-Table mappings for the provide message.

41. List-Query-Request

Data Area Element values (see SctSegments for all elements):

Consumption logic:
The Banner gateway must consume this request when an application wants to retrieve a list of "valid values" for a given List (ValidationListSpec/ListType) from Banner. Based on the ListType passed in Banner should retrieve all "description" column rows for the List and return them in the List-Provide-Reply message. The status (success or failure and any errors if failure) of the query should be included in the Result element of the ControlAreaReply in the Provide message.

Based on the ValidationListSpec/ListFilter, the list being returned from Banner may be a subset of values from the Banner table.

See the List-Provide-Reply Production Logic for Message-Table mappings for the provide message.

8. Step 8
For each remaining messaging component listed in step 6, list the new messages that component must produce and consume and provide brief s stories describing the prescribed production and consumption logic. This section should be completed during the integration design phase with AITS integration staff.

8.1. Messaging Component 1: Paymaster Gateway
8.1.1. Messages

1. Person-BasicPerson-Create-Sync
2. Person-BasicPerson-Delete-Sync
3. Person-BasicPerson-Update-Sync
4. Employee-BasicEmployee-Create-Sync
5. Employee-BasicEmployee-Delete-Sync
6. Employee-BasicEmployee-Update-Sync
7. Generic-Response-Reply

8.1.2. Brief Story

The Paymaster gateway must produce the BasicPerson and BasicEmployee synchronization messages.

8.2. Messaging Component 2: Nessie, Nessie Gateway, PANDA

8.2.1. Messages

1. Person-Education-Create-Request
2. Person-Education-Delete-Request
3. Person-Education-Provide-Reply
4. Person-Education-Query-Request
5. Person-Education-Update-Request
6. Person-EmergencyContact-Create-Request
7. Person-EmergencyContact-Delete-Request
8. Person-EmergencyContact-Provide-Reply
9. Person-EmergencyContact-Query-Request
10. Person-EmergencyContact-Update-Request
11. Person-HonorAward-Create-Request
12. Person-HonorAward-Delete-Request
13. Person-HonorAward-Provide-Reply
14. Person-HonorAward-Query-Request
15. Person-HonorAward-Update-Request
16. Person-Language-Create-Request
17. Person-Language-Delete-Request
18. Person-Language-Provide-Reply
19. Person-Language-Query-Request
20. Person-Language-Update-Request
21. Person-LicenseCertification-Create-Request
22. Person-LicenseCertification-Delete-Request
23. Person-LicenseCertification-Provide-Reply
24. Person-LicenseCertification-Query-Request
25. Person-LicenseCertification-Update-Request
26. Person-Publication-Create-Request
27. Person-Publication-Delete-Request
28. Person-Publication-Provide-Reply
29. Person-Publication-Query-Request
30. Person-Publication-Update-Request
31. Person-WorkHistory-Create-Request
32. Person-WorkHistory-Delete-Request
33. Person-WorkHistory-Provide-Reply
34. Person-WorkHistory-Query-Request
35. Person-WorkHistory-Update-Request
36. Person-DriverLicense-Create-Request
37. Person-DriverLicense-Delete-Request
38. Person-DriverLicense-Provide-Reply
39. Person-DriverLicense-Query-Request
40. Person-DriverLicense-Update-Request
41. Validation-List-Query-Request
42. Validation-List-Provide-Reply
8.2.2 Brief Story

Nessie and PANDA will produce attachment query requests when they want to retrieve attachment data for a particular employee. Banner must handle the request and provide reply. Nessie and the Nessie Gateway will also produce attachment create, update and delete requests when it wishes to create, update or delete an employees attachment data from Banner. Banner must consume these requests, perform the consumption logic listed above in step 7.3 and return the appropriate result.
Appendix 3: The GNU Free Documentation License

GNU Free Documentation License
Version 1.1, March 2000

Copyright (C) 2000 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

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
(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 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 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
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."

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,
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.