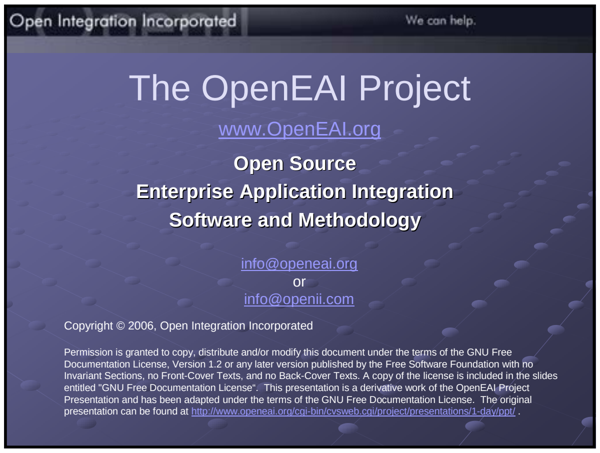

Open Integration Incorporated

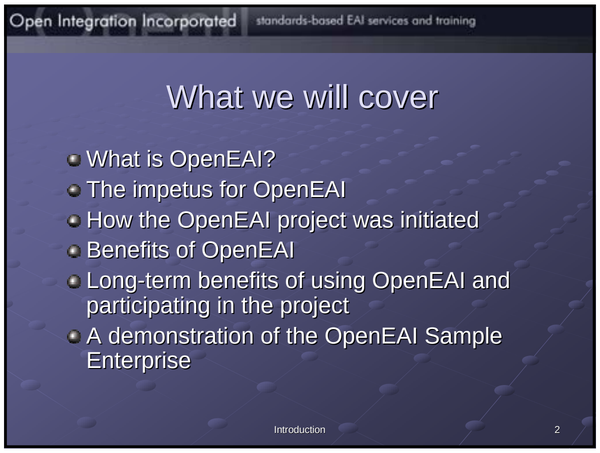
We can help.

The OpenEAI Project

www.OpenEAI.org

Open Source
Enterprise Application Integration
Software and Methodology

info@openeai.org
or
info@openii.com

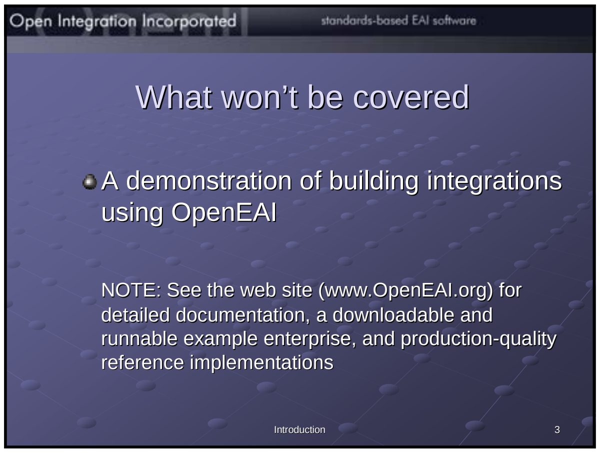

Open Integration Incorporated

standards-based EAI services and training

What we will cover

- What is OpenEAI?
- The impetus for OpenEAI
- How the OpenEAI project was initiated
- Benefits of OpenEAI
- Long-term benefits of using OpenEAI and participating in the project
- A demonstration of the OpenEAI Sample Enterprise

Introduction 2


Open Integration Incorporated

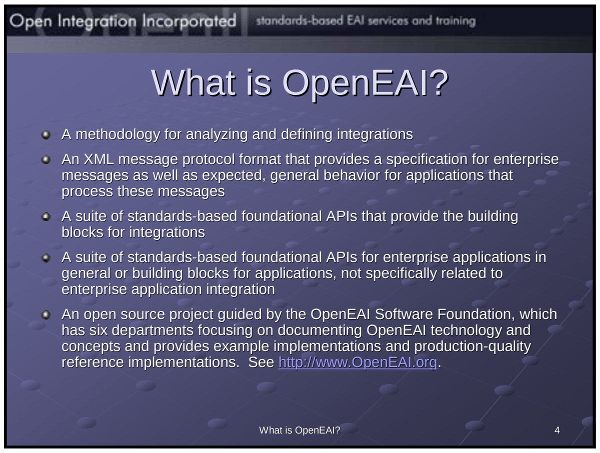
standards-based EAI software

What won't be covered

- A demonstration of building integrations using OpenEAI

NOTE: See the web site (www.OpenEAI.org) for detailed documentation, a downloadable and runnable example enterprise, and production-quality reference implementations

Introduction 3

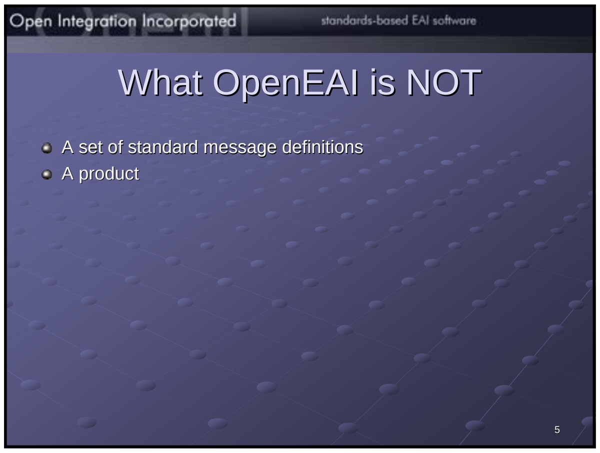


Open Integration Incorporated standards-based EAI services and training

What is OpenEAI?

- A methodology for analyzing and defining integrations
- An XML message protocol format that provides a specification for enterprise messages as well as expected, general behavior for applications that process these messages
- A suite of standards-based foundational APIs that provide the building blocks for integrations
- A suite of standards-based foundational APIs for enterprise applications in general or building blocks for applications, not specifically related to enterprise application integration
- An open source project guided by the OpenEAI Software Foundation, which has six departments focusing on documenting OpenEAI technology and concepts and provides example implementations and production-quality reference implementations. See http://www.OpenEAI.org.

What is OpenEAI? 4

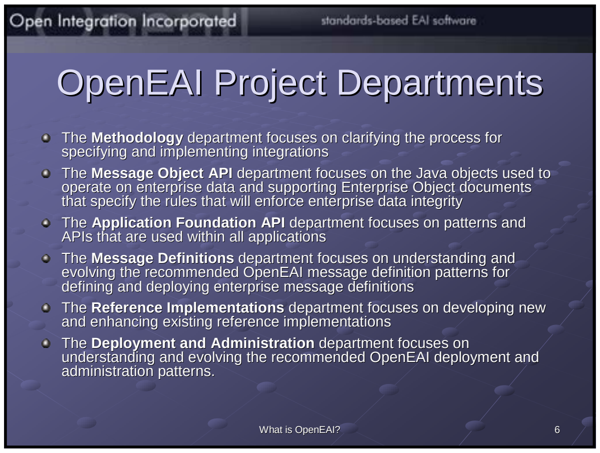


Open Integration Incorporated standards-based EAI software

What OpenEAI is NOT

- A set of standard message definitions
- A product

5

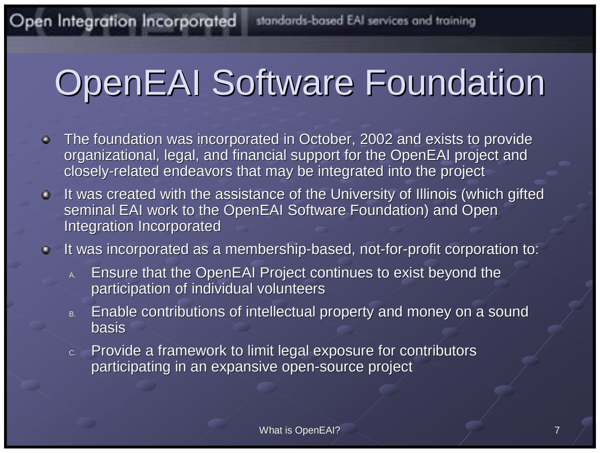


Open Integration Incorporated standards-based EAI software

OpenEAI Project Departments

- The **Methodology** department focuses on clarifying the process for specifying and implementing integrations
- The **Message Object API** department focuses on the Java objects used to operate on enterprise data and supporting Enterprise Object documents that specify the rules that will enforce enterprise data integrity
- The **Application Foundation API** department focuses on patterns and APIs that are used within all applications
- The **Message Definitions** department focuses on understanding and evolving the recommended OpenEAI message definition patterns for defining and deploying enterprise message definitions
- The **Reference Implementations** department focuses on developing new and enhancing existing reference implementations
- The **Deployment and Administration** department focuses on understanding and evolving the recommended OpenEAI deployment and administration patterns.

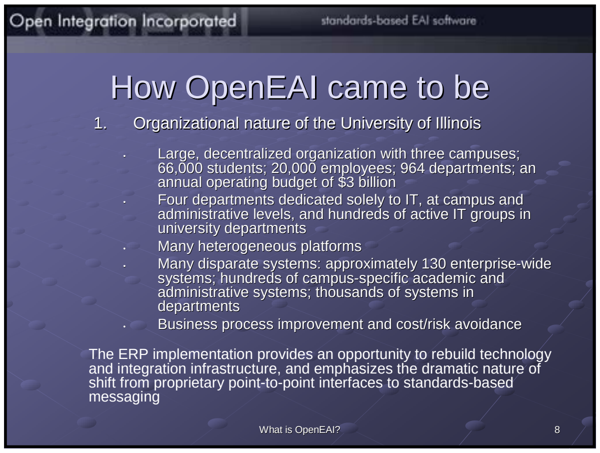
What is OpenEAI? 6


Open Integration Incorporated | standards-based EAI services and training

OpenEAI Software Foundation

- The foundation was incorporated in October, 2002 and exists to provide organizational, legal, and financial support for the OpenEAI project and closely-related endeavors that may be integrated into the project
- It was created with the assistance of the University of Illinois (which gifted seminal EAI work to the OpenEAI Software Foundation) and Open Integration Incorporated
- It was incorporated as a membership-based, not-for-profit corporation to:
  - A. Ensure that the OpenEAI Project continues to exist beyond the participation of individual volunteers
  - B. Enable contributions of intellectual property and money on a sound basis
  - C. Provide a framework to limit legal exposure for contributors participating in an expansive open-source project

What is OpenEAI? 7

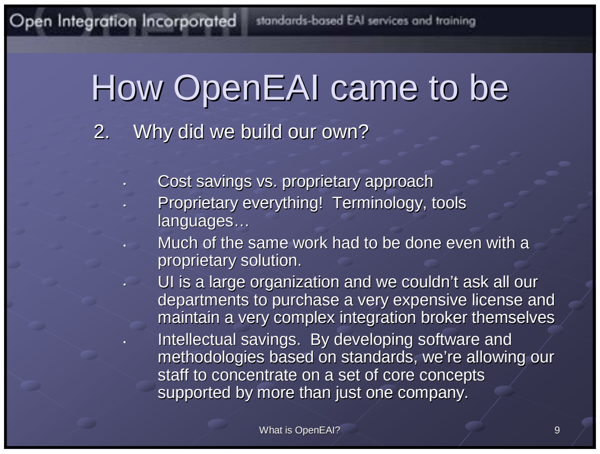

Open Integration Incorporated | standards-based EAI software

How OpenEAI came to be

1. Organizational nature of the University of Illinois
   - Large, decentralized organization with three campuses; 66,000 students; 20,000 employees; 964 departments; an annual operating budget of $3 billion
   - Four departments dedicated solely to IT, at campus and administrative levels, and hundreds of active IT groups in university departments
   - Many heterogeneous platforms
   - Many disparate systems: approximately 130 enterprise-wide systems; hundreds of campus-specific academic and administrative systems; thousands of systems in departments
   - Business process improvement and cost/risk avoidance

The ERP implementation provides an opportunity to rebuild technology and integration infrastructure, and emphasizes the dramatic nature of shift from proprietary point-to-point interfaces to standards-based messaging

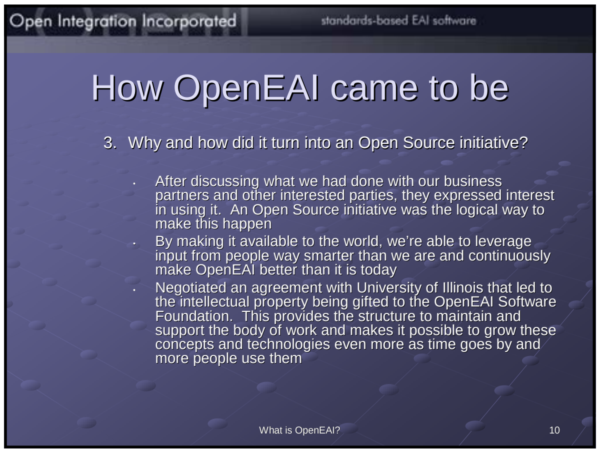
What is OpenEAI? 8


Open Integration Incorporated | standards-based EAI services and training

How OpenEAI came to be

2. Why did we build our own?
   - Cost savings vs. proprietary approach
   - Proprietary everything! Terminology, tools languages…
   - Much of the same work had to be done even with a proprietary solution.
   - UI is a large organization and we couldn't ask all our departments to purchase a very expensive license and maintain a very complex integration broker themselves
   - Intellectual savings. By developing software and methodologies based on standards, we're allowing our staff to concentrate on a set of core concepts supported by more than just one company.

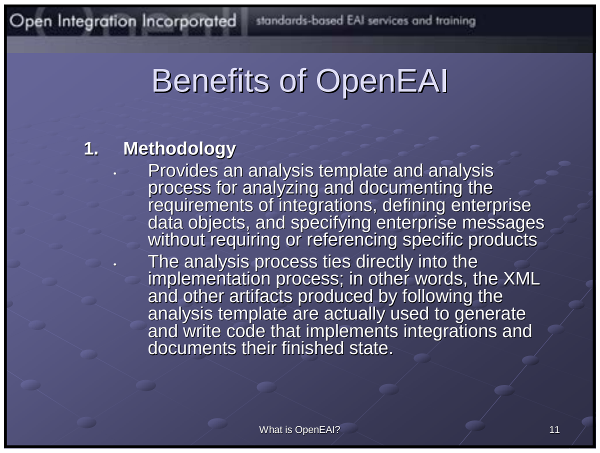
What is OpenEAI? 9


Open Integration Incorporated standards-based EAI software

How OpenEAI came to be

3. Why and how did it turn into an Open Source initiative?
   - After discussing what we had done with our business partners and other interested parties, they expressed interest in using it. An Open Source initiative was the logical way to make this happen
   - By making it available to the world, we're able to leverage input from people way smarter than we are and continuously make OpenEAI better than it is today
   - Negotiated an agreement with University of Illinois that led to the intellectual property being gifted to the OpenEAI Software Foundation. This provides the structure to maintain and support the body of work and makes it possible to grow these concepts and technologies even more as time goes by and more people use them

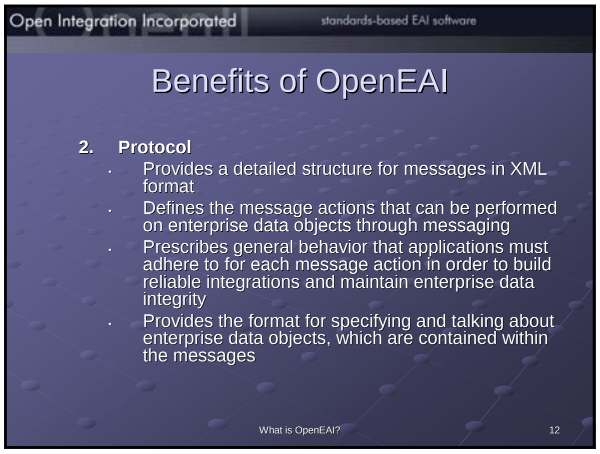
What is OpenEAI? 10


Open Integration Incorporated standards-based EAI services and training

Benefits of OpenEAI

1. **Methodology**
   - Provides an analysis template and analysis process for analyzing and documenting the requirements of integrations, defining enterprise data objects, and specifying enterprise messages without requiring or referencing specific products
   - The analysis process ties directly into the implementation process; in other words, the XML and other artifacts produced by following the analysis template are actually used to generate and write code that implements integrations and documents their finished state.

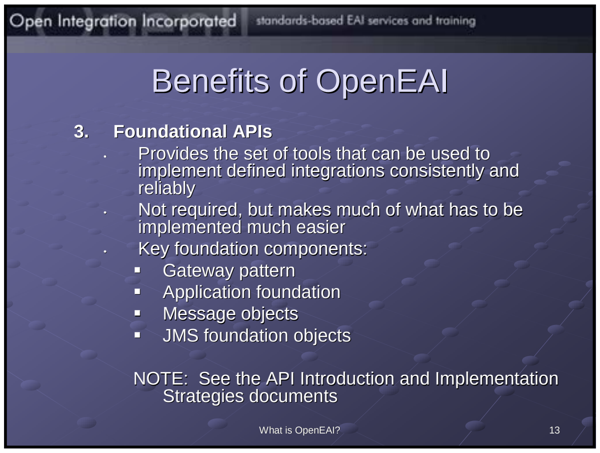
What is OpenEAI? 11


Open Integration Incorporated standards-based EAI software

Benefits of OpenEAI

2. **Protocol**
   - Provides a detailed structure for messages in XML format
   - Defines the message actions that can be performed on enterprise data objects through messaging
   - Prescribes general behavior that applications must adhere to for each message action in order to build reliable integrations and maintain enterprise data integrity
   - Provides the format for specifying and talking about enterprise data objects, which are contained within the messages

What is OpenEAI? 12

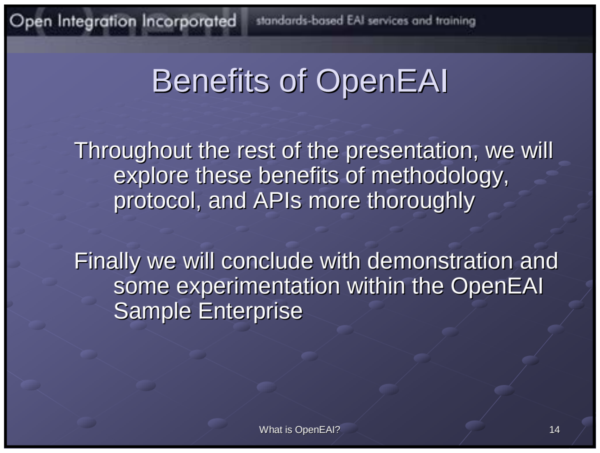

Open Integration Incorporated standards-based EAI services and training

Benefits of OpenEAI

3. **Foundational APIs**
   - Provides the set of tools that can be used to implement defined integrations consistently and reliably
   - Not required, but makes much of what has to be implemented much easier
   - Key foundation components:
     - Gateway pattern
     - Application foundation
     - Message objects
     - JMS foundation objects

NOTE: See the API Introduction and Implementation Strategies documents

What is OpenEAI? 13

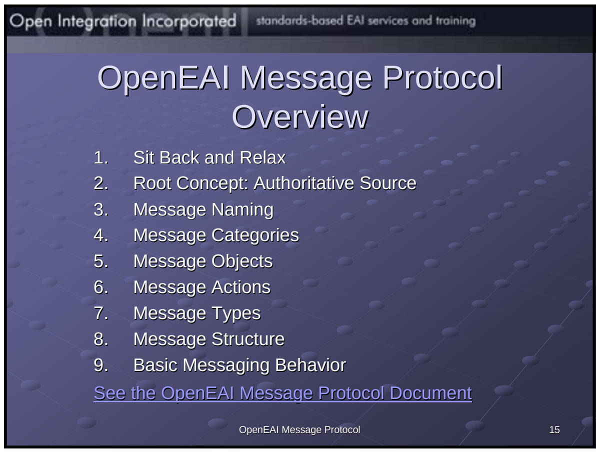

Open Integration Incorporated standards-based EAI services and training

Benefits of OpenEAI

Throughout the rest of the presentation, we will explore these benefits of methodology, protocol, and APIs more thoroughly

Finally we will conclude with demonstration and some experimentation within the OpenEAI Sample Enterprise

What is OpenEAI? 14

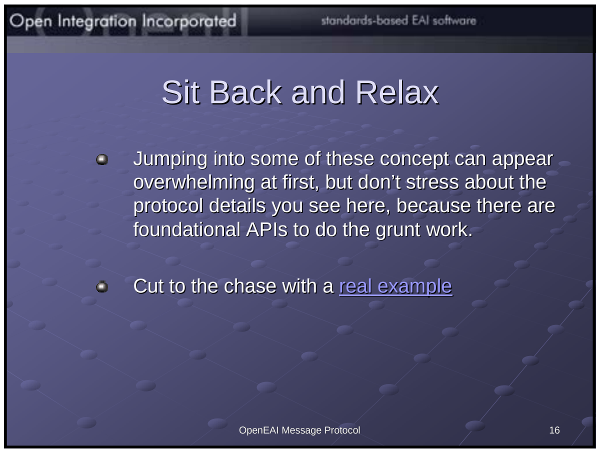

Open Integration Incorporated standards-based EAI services and training

OpenEAI Message Protocol Overview

1. Sit Back and Relax
2. Root Concept: Authoritative Source
3. Message Naming
4. Message Categories
5. Message Objects
6. Message Actions
7. Message Types
8. Message Structure
9. Basic Messaging Behavior

See the OpenEAI Message Protocol Document

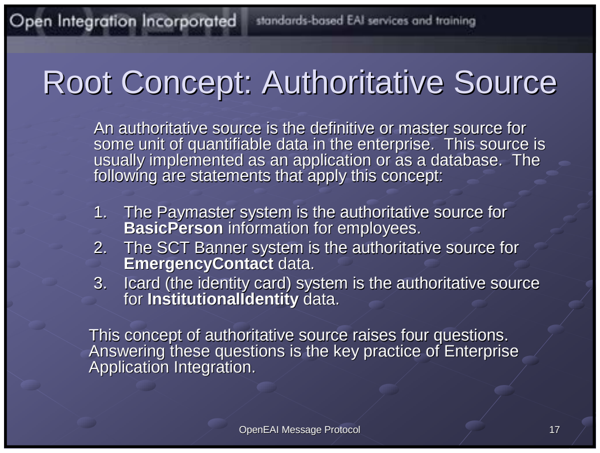
OpenEAI Message Protocol 15


Open Integration Incorporated standards-based EAI software

Sit Back and Relax

- Jumping into some of these concept can appear overwhelming at first, but don't stress about the protocol details you see here, because there are foundational APIs to do the grunt work.
- Cut to the chase with a real example

OpenEAI Message Protocol 16


Open Integration Incorporated standards-based EAI services and training

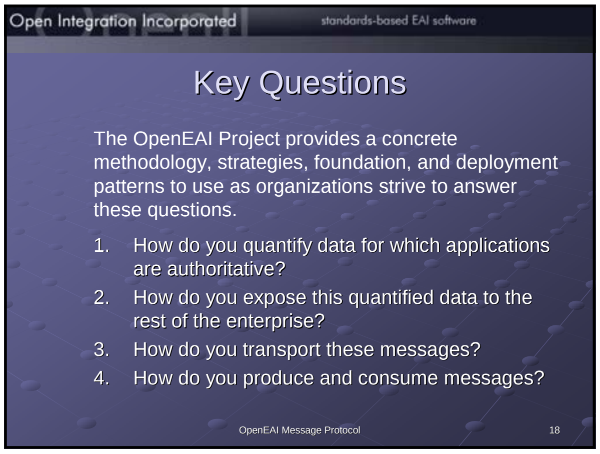
Root Concept: Authoritative Source

An authoritative source is the definitive or master source for some unit of quantifiable data in the enterprise. This source is usually implemented as an application or as a database. The following are statements that apply this concept:

1. The Paymaster system is the authoritative source for **BasicPerson** information for employees.
2. The SCT Banner system is the authoritative source for **EmergencyContact** data.
3. Icard (the identity card) system is the authoritative source for **InstitutionalIdentity** data.

This concept of authoritative source raises four questions. Answering these questions is the key practice of Enterprise Application Integration.

OpenEAI Message Protocol 17

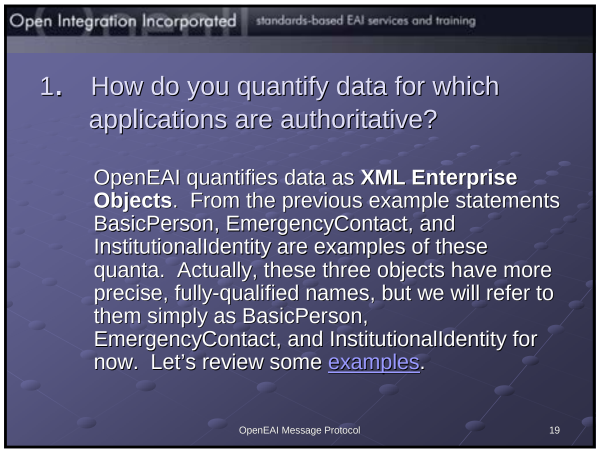

Open Integration Incorporated standards-based EAI software

Key Questions

The OpenEAI Project provides a concrete methodology, strategies, foundation, and deployment patterns to use as organizations strive to answer these questions.

1. How do you quantify data for which applications are authoritative?
2. How do you expose this quantified data to the rest of the enterprise?
3. How do you transport these messages?
4. How do you produce and consume messages?

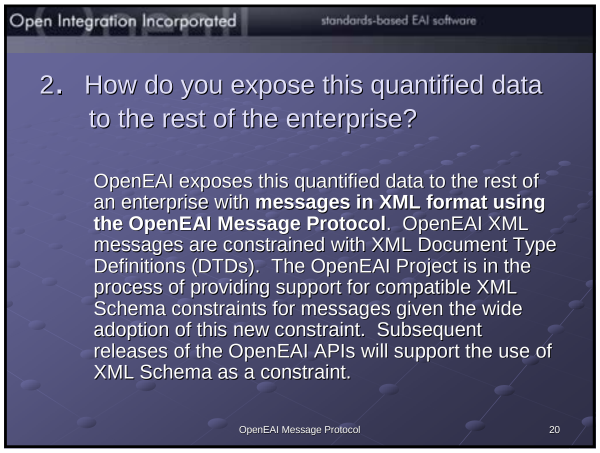
OpenEAI Message Protocol 18


Open Integration Incorporated standards-based EAI services and training

1. How do you quantify data for which applications are authoritative?

OpenEAI quantifies data as **XML Enterprise Objects**. From the previous example statements BasicPerson, EmergencyContact, and InstitutionalIdentity are examples of these quanta. Actually, these three objects have more precise, fully-qualified names, but we will refer to them simply as BasicPerson, EmergencyContact, and InstitutionalIdentity for now. Let's review some examples.

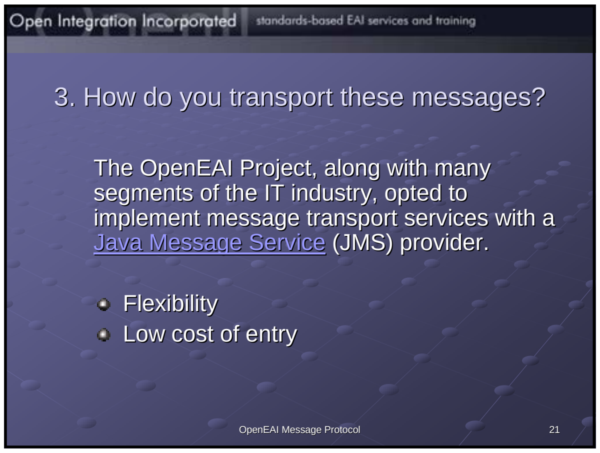
OpenEAI Message Protocol 19


Open Integration Incorporated standards-based EAI software

2. How do you expose this quantified data to the rest of the enterprise?

OpenEAI exposes this quantified data to the rest of an enterprise with **messages in XML format using the OpenEAI Message Protocol**. OpenEAI XML messages are constrained with XML Document Type Definitions (DTDs). The OpenEAI Project is in the process of providing support for compatible XML Schema constraints for messages given the wide adoption of this new constraint. Subsequent releases of the OpenEAI APIs will support the use of XML Schema as a constraint.

OpenEAI Message Protocol 20

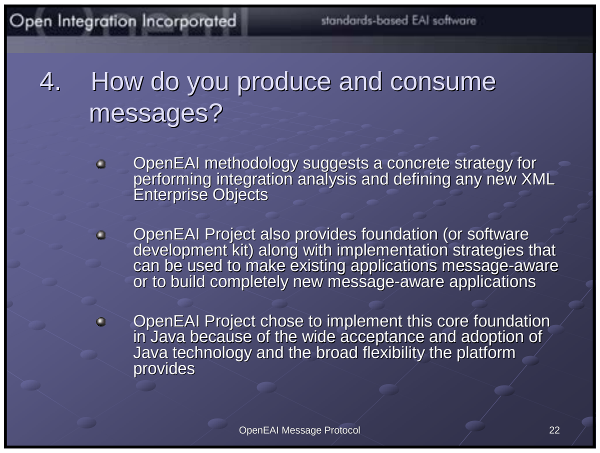

Open Integration Incorporated standards-based EAI services and training

3. How do you transport these messages?

The OpenEAI Project, along with many segments of the IT industry, opted to implement message transport services with a Java Message Service (JMS) provider.

- Flexibility
- Low cost of entry

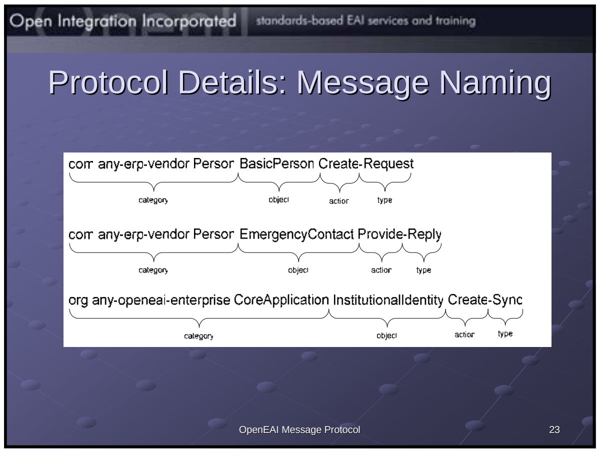
OpenEAI Message Protocol 21


Open Integration Incorporated standards-based EAI software

4. How do you produce and consume messages?

- OpenEAI methodology suggests a concrete strategy for performing integration analysis and defining any new XML Enterprise Objects
- OpenEAI Project also provides foundation (or software development kit) along with implementation strategies that can be used to make existing applications message-aware or to build completely new message-aware applications
- OpenEAI Project chose to implement this core foundation in Java because of the wide acceptance and adoption of Java technology and the broad flexibility the platform provides

OpenEAI Message Protocol 22


Open Integration Incorporated standards-based EAI services and training

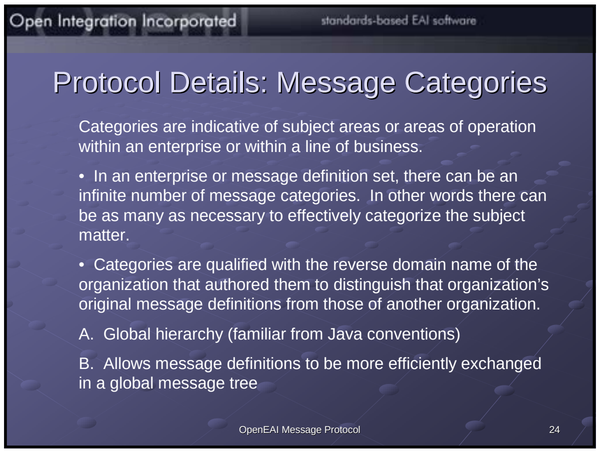
Protocol Details: Message Naming

$\underbrace{\text{com any-erp-vendor Person}}_{\text{category}}\ \underbrace{\text{BasicPerson}}_{\text{object}}\ \underbrace{\text{Create}}_{\text{action}}\text{-}\underbrace{\text{Request}}_{\text{type}}$

$\underbrace{\text{com any-erp-vendor Person}}_{\text{category}}\ \underbrace{\text{EmergencyContact}}_{\text{object}}\ \underbrace{\text{Provide}}_{\text{action}}\text{-}\underbrace{\text{Reply}}_{\text{type}}$

$\underbrace{\text{org any-openeai-enterprise CoreApplication}}_{\text{category}}\ \underbrace{\text{InstitutionalIdentity}}_{\text{object}}\ \underbrace{\text{Create}}_{\text{action}}\text{-}\underbrace{\text{Sync}}_{\text{type}}$

OpenEAI Message Protocol 23


Open Integration Incorporated standards-based EAI software

Protocol Details: Message Categories

Categories are indicative of subject areas or areas of operation within an enterprise or within a line of business.

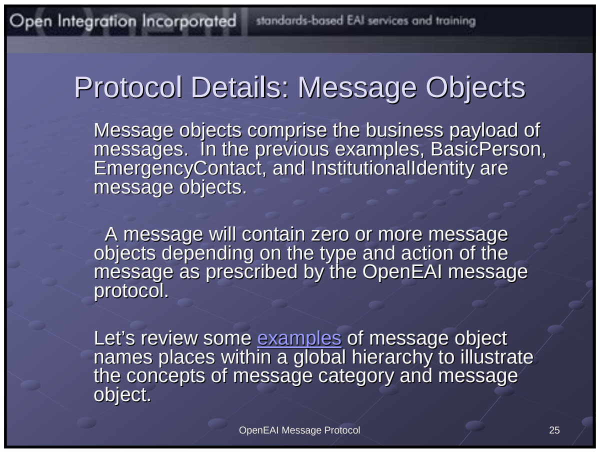
• In an enterprise or message definition set, there can be an infinite number of message categories. In other words there can be as many as necessary to effectively categorize the subject matter.

• Categories are qualified with the reverse domain name of the organization that authored them to distinguish that organization's original message definitions from those of another organization.

A. Global hierarchy (familiar from Java conventions)

B. Allows message definitions to be more efficiently exchanged in a global message tree

OpenEAI Message Protocol 24


Open Integration Incorporated standards-based EAI services and training

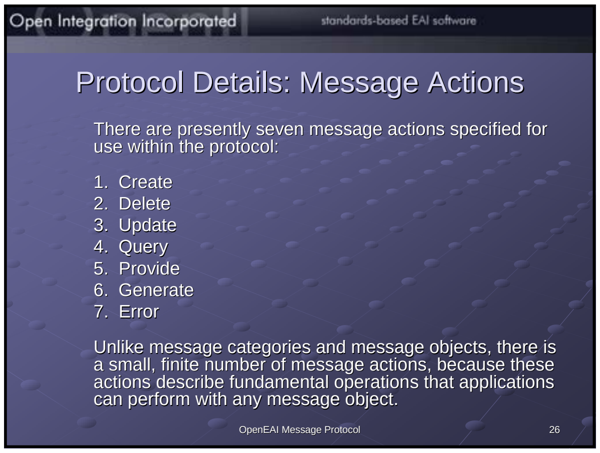
Protocol Details: Message Objects

Message objects comprise the business payload of messages. In the previous examples, BasicPerson, EmergencyContact, and InstitutionalIdentity are message objects.

A message will contain zero or more message objects depending on the type and action of the message as prescribed by the OpenEAI message protocol.

Let's review some examples of message object names places within a global hierarchy to illustrate the concepts of message category and message object.

OpenEAI Message Protocol 25


Open Integration Incorporated standards-based EAI software

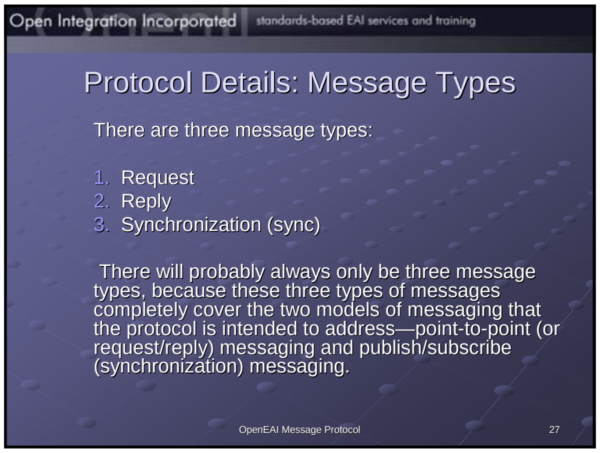
Protocol Details: Message Actions

There are presently seven message actions specified for use within the protocol:

1. Create
2. Delete
3. Update
4. Query
5. Provide
6. Generate
7. Error

Unlike message categories and message objects, there is a small, finite number of message actions, because these actions describe fundamental operations that applications can perform with any message object.

OpenEAI Message Protocol 26


Open Integration Incorporated standards-based EAI services and training

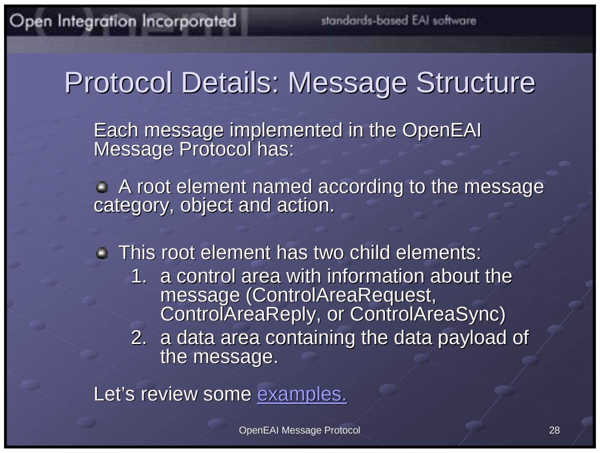
Protocol Details: Message Types

There are three message types:

1. Request
2. Reply
3. Synchronization (sync)

There will probably always only be three message types, because these three types of messages completely cover the two models of messaging that the protocol is intended to address—point-to-point (or request/reply) messaging and publish/subscribe (synchronization) messaging.

OpenEAI Message Protocol 27


Open Integration Incorporated standards-based EAI software

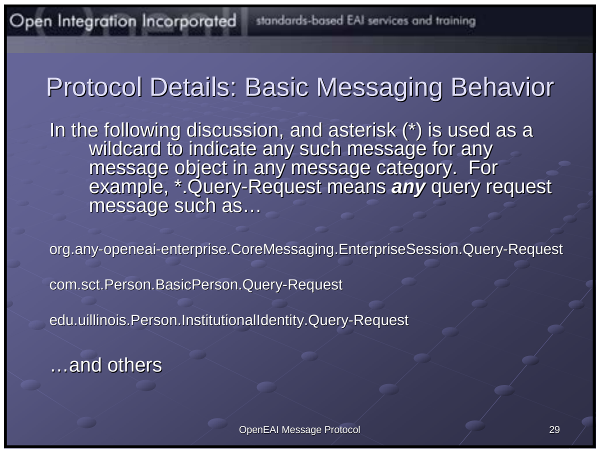
Protocol Details: Message Structure

Each message implemented in the OpenEAI Message Protocol has:

- A root element named according to the message category, object and action.
- This root element has two child elements:
  1. a control area with information about the message (ControlAreaRequest, ControlAreaReply, or ControlAreaSync)
  2. a data area containing the data payload of the message.

Let's review some examples.

OpenEAI Message Protocol 28


Open Integration Incorporated standards-based EAI services and training

Protocol Details: Basic Messaging Behavior

In the following discussion, and asterisk (*) is used as a wildcard to indicate any such message for any message object in any message category. For example, *.Query-Request means ***any*** query request message such as…

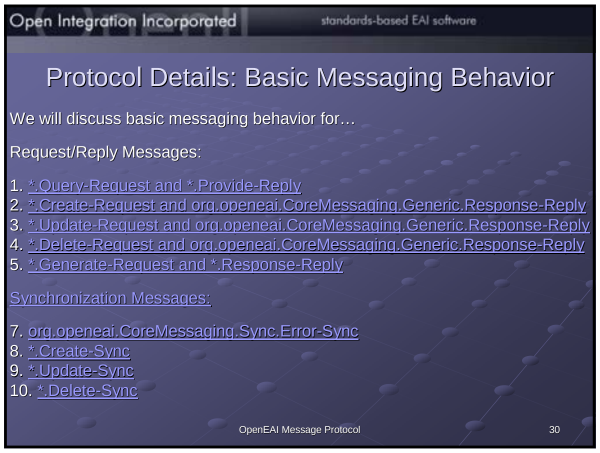
org.any-openeai-enterprise.CoreMessaging.EnterpriseSession.Query-Request

com.sct.Person.BasicPerson.Query-Request

edu.uillinois.Person.InstitutionalIdentity.Query-Request

…and others

OpenEAI Message Protocol 29


Open Integration Incorporated standards-based EAI software

Protocol Details: Basic Messaging Behavior

We will discuss basic messaging behavior for…

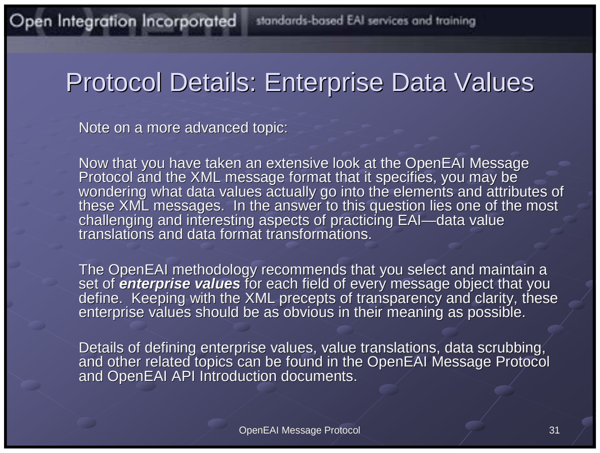
Request/Reply Messages:

1. *.Query-Request and *.Provide-Reply
2. *.Create-Request and org.openeai.CoreMessaging.Generic.Response-Reply
3. *.Update-Request and org.openeai.CoreMessaging.Generic.Response-Reply
4. *.Delete-Request and org.openeai.CoreMessaging.Generic.Response-Reply
5. *.Generate-Request and *.Response-Reply

Synchronization Messages:

7. org.openeai.CoreMessaging.Sync.Error-Sync
8. *.Create-Sync
9. *.Update-Sync
10. *.Delete-Sync

OpenEAI Message Protocol 30


Open Integration Incorporated standards-based EAI services and training

Protocol Details: Enterprise Data Values

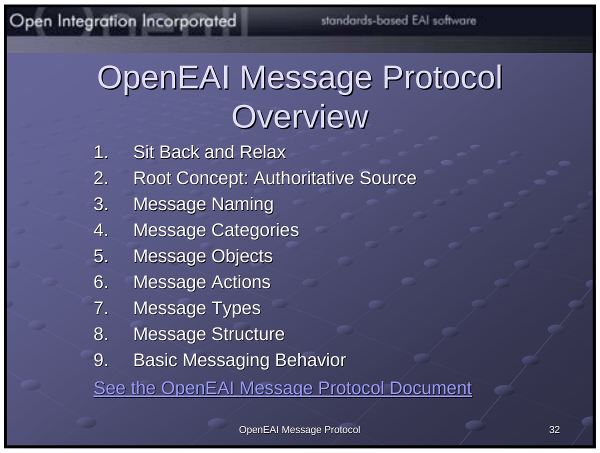
Note on a more advanced topic:

Now that you have taken an extensive look at the OpenEAI Message Protocol and the XML message format that it specifies, you may be wondering what data values actually go into the elements and attributes of these XML messages. In the answer to this question lies one of the most challenging and interesting aspects of practicing EAI—data value translations and data format transformations.

The OpenEAI methodology recommends that you select and maintain a set of ***enterprise values*** for each field of every message object that you define. Keeping with the XML precepts of transparency and clarity, these enterprise values should be as obvious in their meaning as possible.

Details of defining enterprise values, value translations, data scrubbing, and other related topics can be found in the OpenEAI Message Protocol and OpenEAI API Introduction documents.

OpenEAI Message Protocol 31

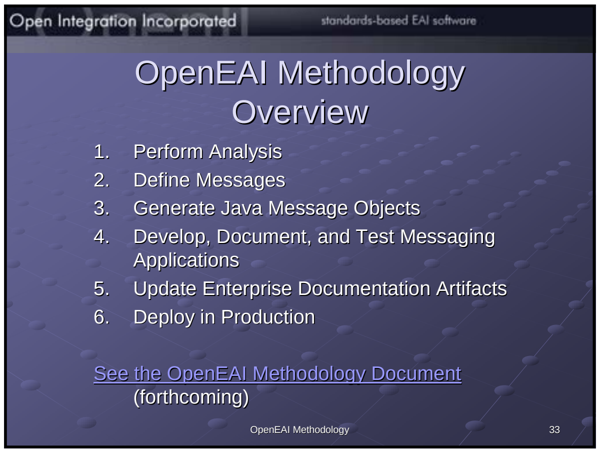

Open Integration Incorporated standards-based EAI software

OpenEAI Message Protocol Overview

1. Sit Back and Relax
2. Root Concept: Authoritative Source
3. Message Naming
4. Message Categories
5. Message Objects
6. Message Actions
7. Message Types
8. Message Structure
9. Basic Messaging Behavior

See the OpenEAI Message Protocol Document

OpenEAI Message Protocol 32

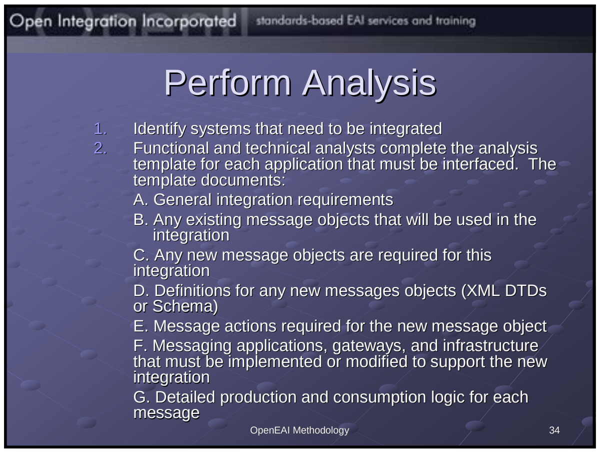

Open Integration Incorporated standards-based EAI software

OpenEAI Methodology Overview

1. Perform Analysis
2. Define Messages
3. Generate Java Message Objects
4. Develop, Document, and Test Messaging Applications
5. Update Enterprise Documentation Artifacts
6. Deploy in Production

See the OpenEAI Methodology Document (forthcoming)

OpenEAI Methodology 33


Open Integration Incorporated standards-based EAI services and training

Perform Analysis

1. Identify systems that need to be integrated
2. Functional and technical analysts complete the analysis template for each application that must be interfaced. The template documents:

A. General integration requirements

B. Any existing message objects that will be used in the integration

C. Any new message objects are required for this integration

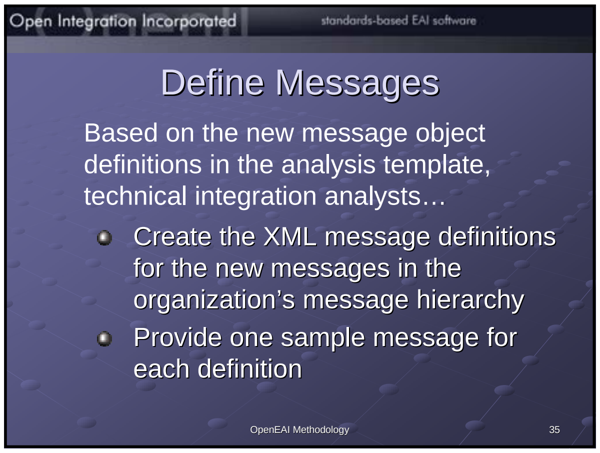
D. Definitions for any new messages objects (XML DTDs or Schema)

E. Message actions required for the new message object

F. Messaging applications, gateways, and infrastructure that must be implemented or modified to support the new integration

G. Detailed production and consumption logic for each message

OpenEAI Methodology 34

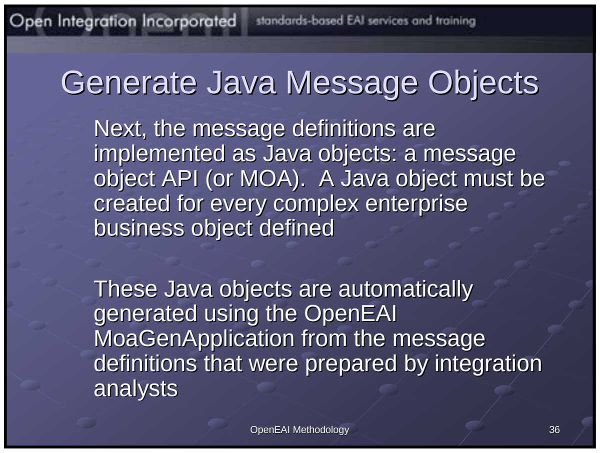

Open Integration Incorporated standards-based EAI software

Define Messages

Based on the new message object definitions in the analysis template, technical integration analysts…

- Create the XML message definitions for the new messages in the organization’s message hierarchy
- Provide one sample message for each definition

OpenEAI Methodology 35

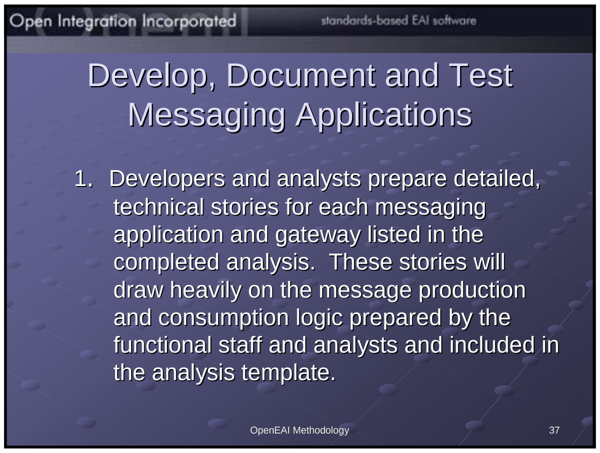

Open Integration Incorporated standards-based EAI services and training

Generate Java Message Objects

Next, the message definitions are implemented as Java objects: a message object API (or MOA). A Java object must be created for every complex enterprise business object defined

These Java objects are automatically generated using the OpenEAI MoaGenApplication from the message definitions that were prepared by integration analysts

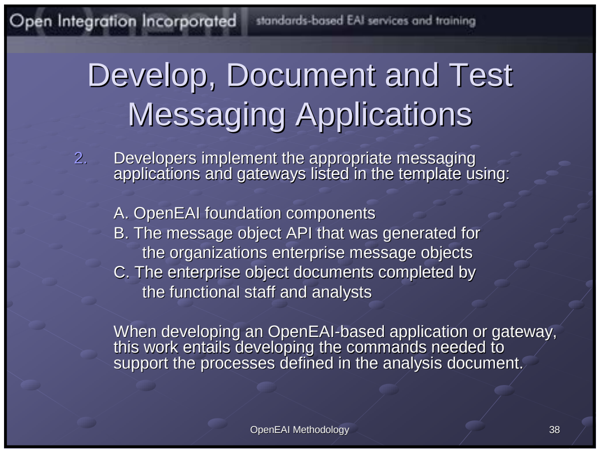
OpenEAI Methodology 36


Open Integration Incorporated standards-based EAI software

Develop, Document and Test Messaging Applications

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 functional staff and analysts and included in the analysis template.

OpenEAI Methodology 37


Open Integration Incorporated standards-based EAI services and training

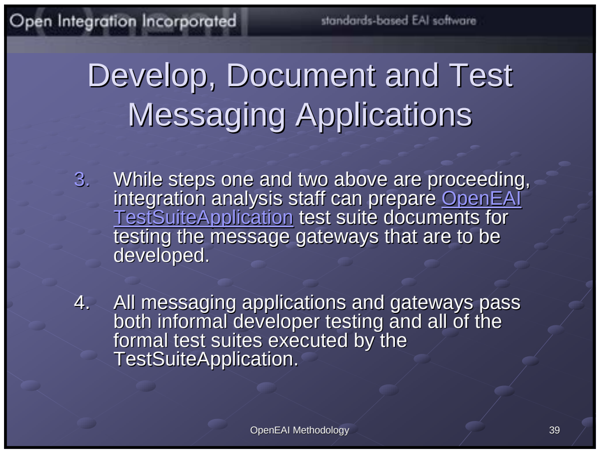
Develop, Document and Test Messaging Applications

2. Developers implement the appropriate messaging applications and gateways listed in the template using:

   A. OpenEAI foundation components
   B. The message object API that was generated for the organizations enterprise message objects
   C. The enterprise object documents completed by the functional staff and analysts

   When developing an OpenEAI-based application or gateway, this work entails developing the commands needed to support the processes defined in the analysis document.

OpenEAI Methodology 38

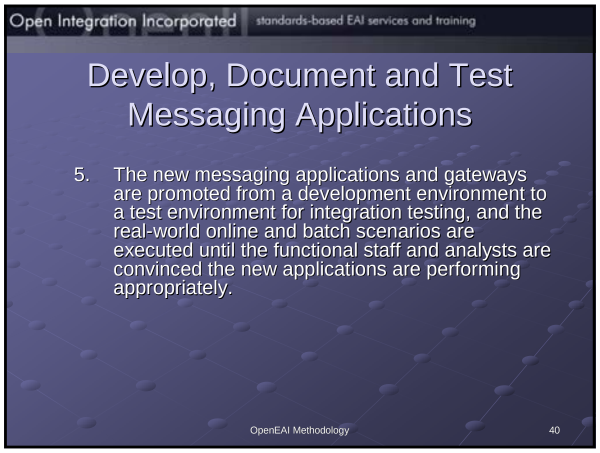

Open Integration Incorporated standards-based EAI software

Develop, Document and Test Messaging Applications

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.

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

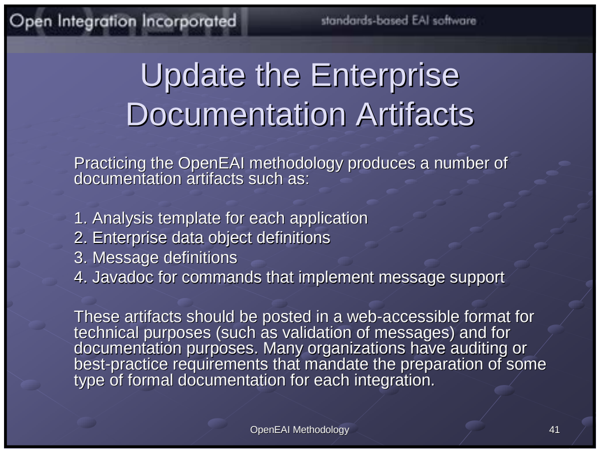
OpenEAI Methodology 39


Open Integration Incorporated standards-based EAI services and training

Develop, Document and Test Messaging Applications

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 functional staff and analysts are convinced the new applications are performing appropriately.

OpenEAI Methodology 40


Open Integration Incorporated standards-based EAI software

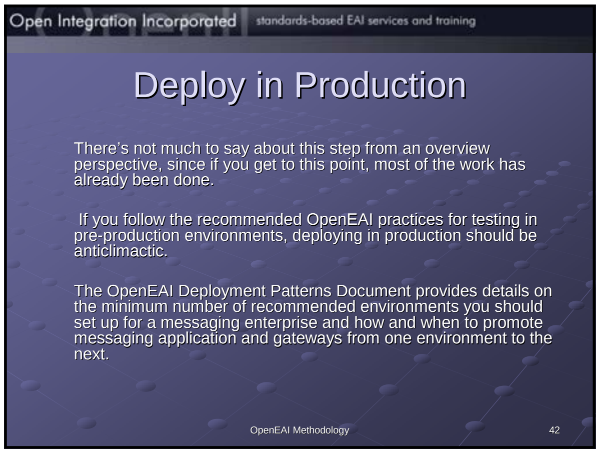
Update the Enterprise Documentation Artifacts

Practicing the OpenEAI methodology produces a number of documentation artifacts such as:

1. Analysis template for each application
2. Enterprise data object definitions
3. Message definitions
4. Javadoc for commands that implement message support

These artifacts should be posted in a web-accessible format for technical purposes (such as validation of messages) and for documentation purposes. Many organizations have auditing or best-practice requirements that mandate the preparation of some type of formal documentation for each integration.

OpenEAI Methodology 41


Open Integration Incorporated standards-based EAI services and training

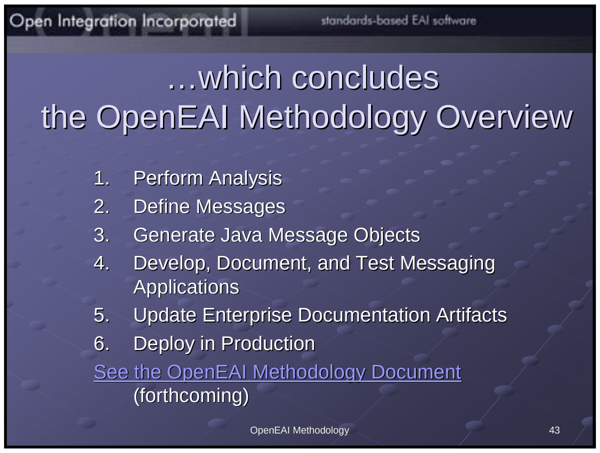
Deploy in Production

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 application and gateways from one environment to the next.

OpenEAI Methodology 42

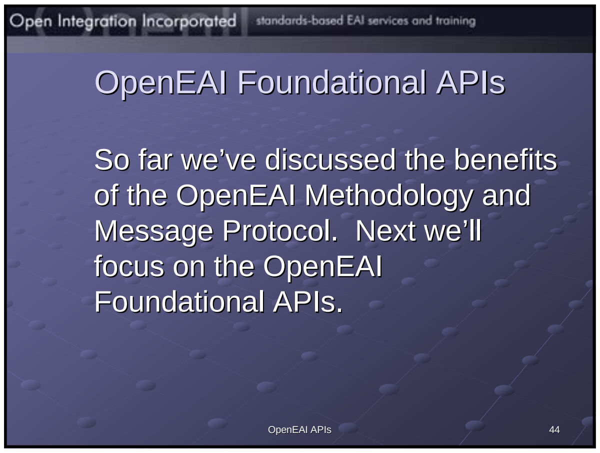

Open Integration Incorporated standards-based EAI software

…which concludes the OpenEAI Methodology Overview

1. Perform Analysis
2. Define Messages
3. Generate Java Message Objects
4. Develop, Document, and Test Messaging Applications
5. Update Enterprise Documentation Artifacts
6. Deploy in Production

See the OpenEAI Methodology Document (forthcoming)

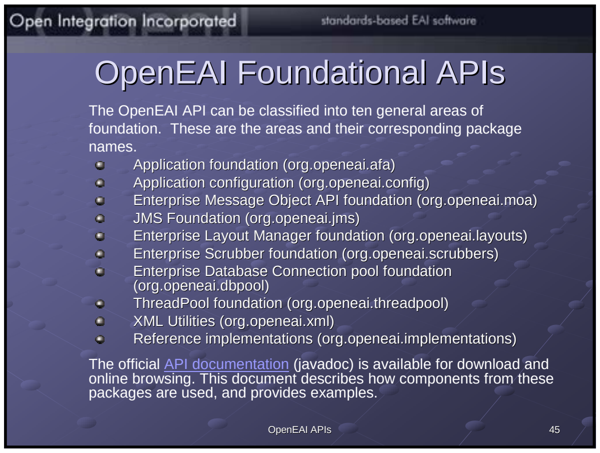
OpenEAI Methodology 43


Open Integration Incorporated standards-based EAI services and training

OpenEAI Foundational APIs

So far we've discussed the benefits of the OpenEAI Methodology and Message Protocol. Next we'll focus on the OpenEAI Foundational APIs.

OpenEAI APIs 44


Open Integration Incorporated standards-based EAI software

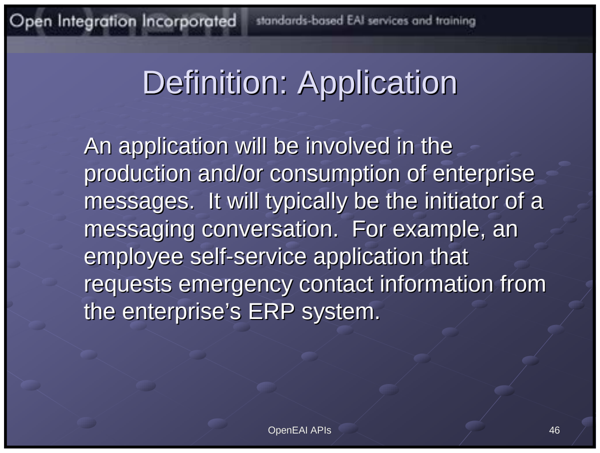
OpenEAI Foundational APIs

The OpenEAI API can be classified into ten general areas of foundation. These are the areas and their corresponding package names.

- Application foundation (org.openeai.afa)
- Application configuration (org.openeai.config)
- Enterprise Message Object API foundation (org.openeai.moa)
- JMS Foundation (org.openeai.jms)
- Enterprise Layout Manager foundation (org.openeai.layouts)
- Enterprise Scrubber foundation (org.openeai.scrubbers)
- Enterprise Database Connection pool foundation (org.openeai.dbpool)
- ThreadPool foundation (org.openeai.threadpool)
- XML Utilities (org.openeai.xml)
- Reference implementations (org.openeai.implementations)

The official API documentation (javadoc) is available for download and online browsing. This document describes how components from these packages are used, and provides examples.

OpenEAI APIs 45

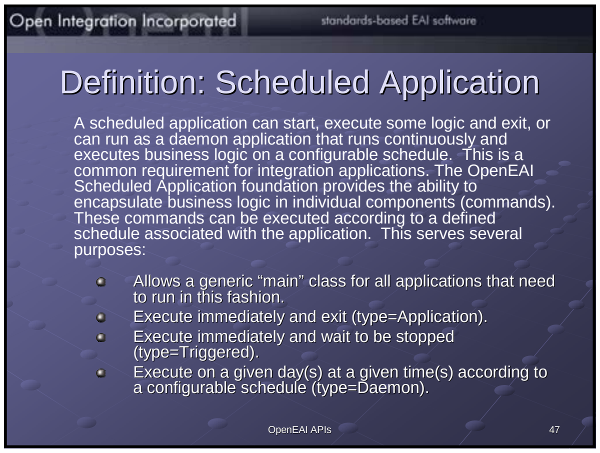


Open Integration Incorporated | standards-based EAI services and training

Definition: Application

An application will be involved in the production and/or consumption of enterprise messages. It will typically be the initiator of a messaging conversation. For example, an employee self-service application that requests emergency contact information from the enterprise's ERP system.

OpenEAI APIs 46



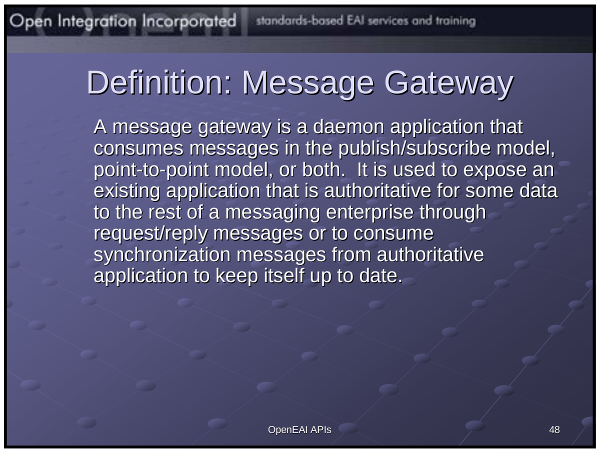
Open Integration Incorporated | standards-based EAI software

Definition: Scheduled Application

A scheduled application can start, execute some logic and exit, or can run as a daemon application that runs continuously and executes business logic on a configurable schedule. This is a common requirement for integration applications. The OpenEAI Scheduled Application foundation provides the ability to encapsulate business logic in individual components (commands). These commands can be executed according to a defined schedule associated with the application. This serves several purposes:

- Allows a generic "main" class for all applications that need to run in this fashion.
- Execute immediately and exit (type=Application).
- Execute immediately and wait to be stopped (type=Triggered).
- Execute on a given day(s) at a given time(s) according to a configurable schedule (type=Daemon).

OpenEAI APIs 47

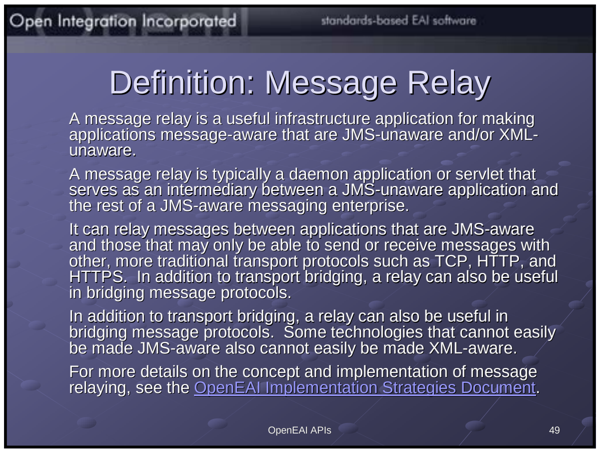


Open Integration Incorporated | standards-based EAI services and training

Definition: Message Gateway

A message gateway is a daemon application that consumes messages in the publish/subscribe model, point-to-point model, or both. It is used to expose an existing application that is authoritative for some data to the rest of a messaging enterprise through request/reply messages or to consume synchronization messages from authoritative application to keep itself up to date.

OpenEAI APIs 48



Open Integration Incorporated standards-based EAI software

Definition: Message Relay

A message relay is a useful infrastructure application for making applications message-aware that are JMS-unaware and/or XML-unaware.

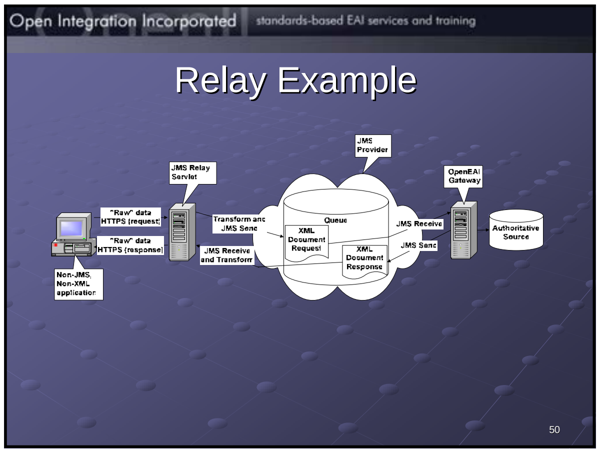
A message relay is typically a daemon application or servlet that serves as an intermediary between a JMS-unaware application and the rest of a JMS-aware messaging enterprise.

It can relay messages between applications that are JMS-aware and those that may only be able to send or receive messages with other, more traditional transport protocols such as TCP, HTTP, and HTTPS. In addition to transport bridging, a relay can also be useful in bridging message protocols.

In addition to transport bridging, a relay can also be useful in bridging message protocols. Some technologies that cannot easily be made JMS-aware also cannot easily be made XML-aware.

For more details on the concept and implementation of message relaying, see the OpenEAI Implementation Strategies Document.

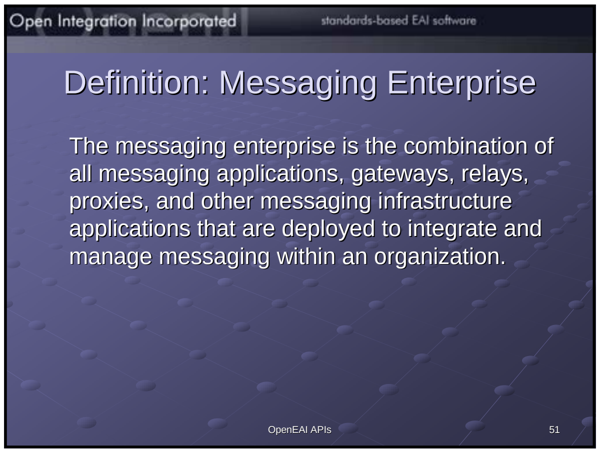
OpenEAI APIs 49



Open Integration Incorporated standards-based EAI services and training

Relay Example

50

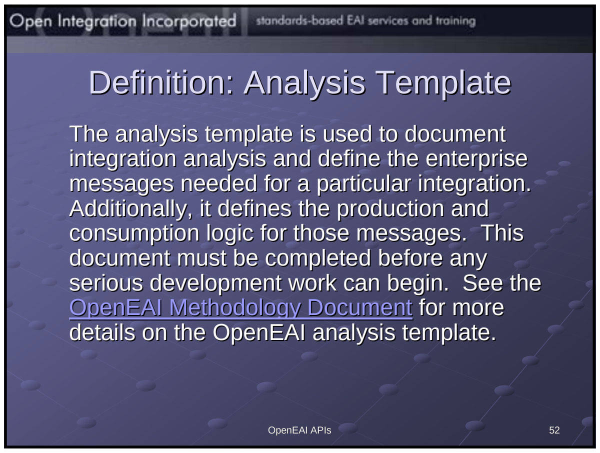


Open Integration Incorporated standards-based EAI software

Definition: Messaging Enterprise

The messaging enterprise is the combination of all messaging applications, gateways, relays, proxies, and other messaging infrastructure applications that are deployed to integrate and manage messaging within an organization.

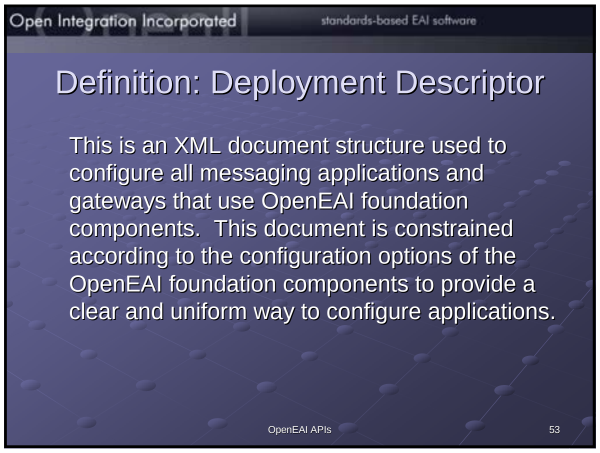
OpenEAI APIs 51


Open Integration Incorporated standards-based EAI services and training

Definition: Analysis Template

The analysis template is used to document integration analysis and define the enterprise messages needed for a particular integration. Additionally, it defines the production and consumption logic for those messages. This document must be completed before any serious development work can begin. See the OpenEAI Methodology Document for more details on the OpenEAI analysis template.

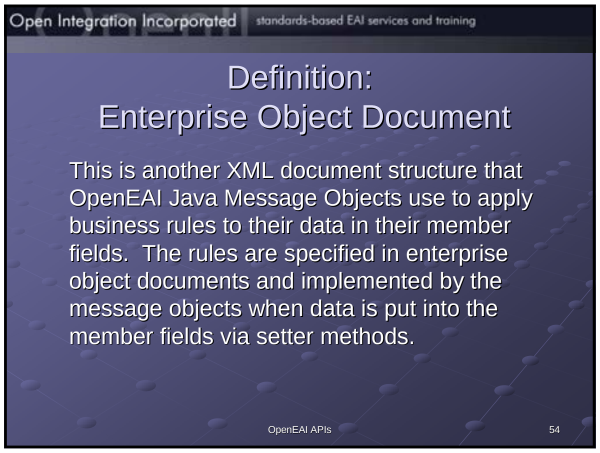
OpenEAI APIs 52


Open Integration Incorporated standards-based EAI software

Definition: Deployment Descriptor

This is an XML document structure used to configure all messaging applications and gateways that use OpenEAI foundation components. This document is constrained according to the configuration options of the OpenEAI foundation components to provide a clear and uniform way to configure applications.

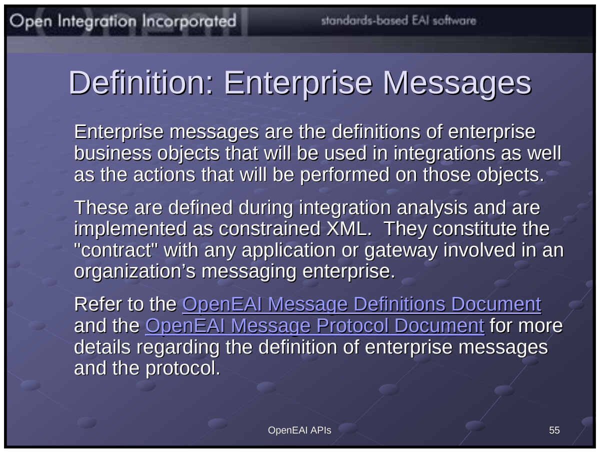
OpenEAI APIs 53


Open Integration Incorporated standards-based EAI services and training

Definition: Enterprise Object Document

This is another XML document structure that OpenEAI Java Message Objects use to apply business rules to their data in their member fields. The rules are specified in enterprise object documents and implemented by the message objects when data is put into the member fields via setter methods.

OpenEAI APIs 54


Open Integration Incorporated standards-based EAI software

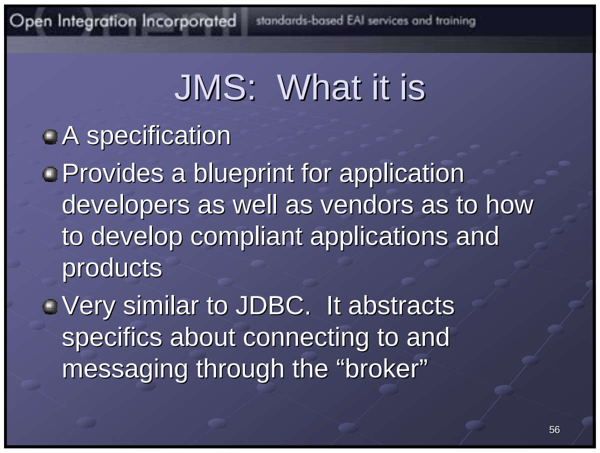
Definition: Enterprise Messages

Enterprise messages are the definitions of enterprise business objects that will be used in integrations as well as the actions that will be performed on those objects.

These are defined during integration analysis and are implemented as constrained XML. They constitute the "contract" with any application or gateway involved in an organization's messaging enterprise.

Refer to the OpenEAI Message Definitions Document and the OpenEAI Message Protocol Document for more details regarding the definition of enterprise messages and the protocol.

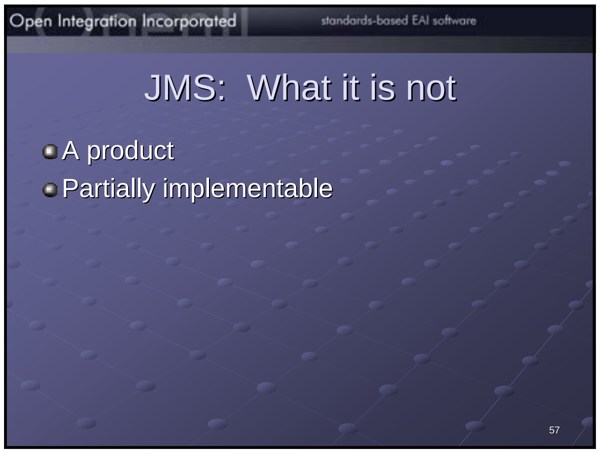
OpenEAI APIs 55


Open Integration Incorporated standards-based EAI services and training

JMS: What it is

- A specification
- Provides a blueprint for application developers as well as vendors as to how to develop compliant applications and products
- Very similar to JDBC. It abstracts specifics about connecting to and messaging through the "broker"

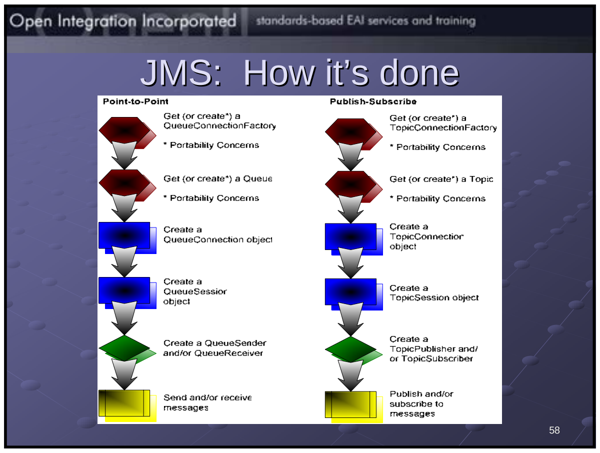
56


Open Integration Incorporated standards-based EAI software

JMS: What it is not

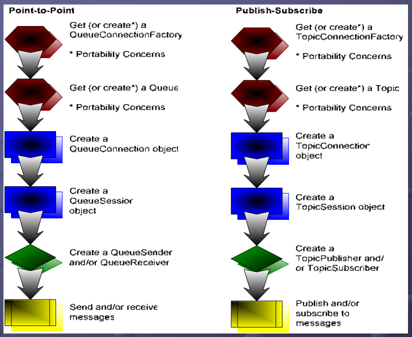
- A product
- Partially implementable

57



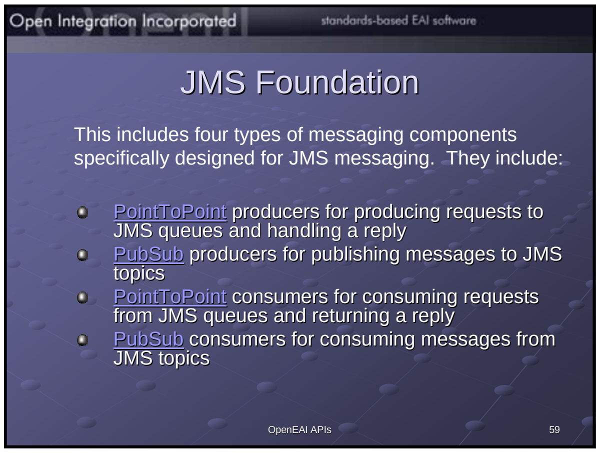
Open Integration Incorporated | standards-based EAI services and training

JMS: How it's done



Point-to-Point
Get (or create*) a QueueConnectionFactory
* Portability Concerns
Get (or create*) a Queue
* Portability Concerns
Create a QueueConnection object
Create a QueueSession object
Create a QueueSender and/or QueueReceiver
Send and/or receive messages
Publish-Subscribe
Get (or create*) a TopicConnectionFactory
* Portability Concerns
Get (or create*) a Topic
* Portability Concerns
Create a TopicConnection object
Create a TopicSession object
Create a TopicPublisher and/ or TopicSubscriber
Publish and/or subscribe to messages

58



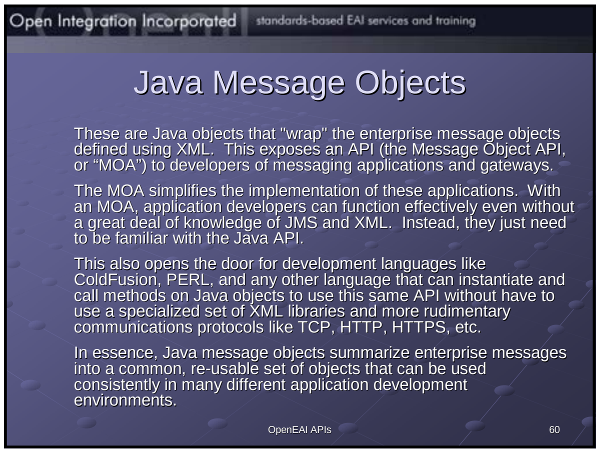
Open Integration Incorporated | standards-based EAI software

JMS Foundation

This includes four types of messaging components specifically designed for JMS messaging. They include:

- PointToPoint producers for producing requests to JMS queues and handling a reply
- PubSub producers for publishing messages to JMS topics
- PointToPoint consumers for consuming requests from JMS queues and returning a reply
- PubSub consumers for consuming messages from JMS topics

OpenEAI APIs 59



Open Integration Incorporated | standards-based EAI services and training

Java Message Objects

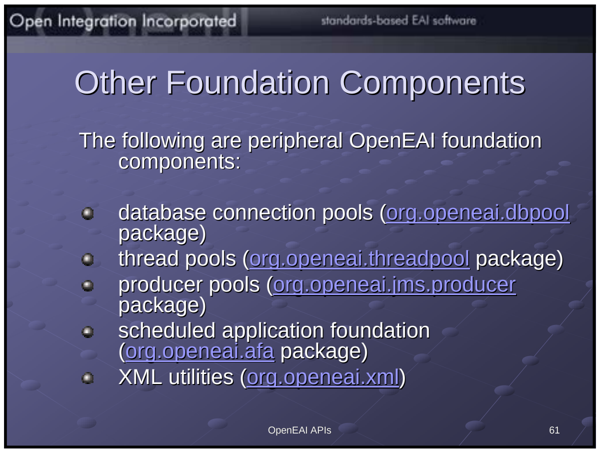
These are Java objects that "wrap" the enterprise message objects defined using XML. This exposes an API (the Message Object API, or "MOA") to developers of messaging applications and gateways.

The MOA simplifies the implementation of these applications. With an MOA, application developers can function effectively even without a great deal of knowledge of JMS and XML. Instead, they just need to be familiar with the Java API.

This also opens the door for development languages like ColdFusion, PERL, and any other language that can instantiate and call methods on Java objects to use this same API without have to use a specialized set of XML libraries and more rudimentary communications protocols like TCP, HTTP, HTTPS, etc.

In essence, Java message objects summarize enterprise messages into a common, re-usable set of objects that can be used consistently in many different application development environments.

OpenEAI APIs 60

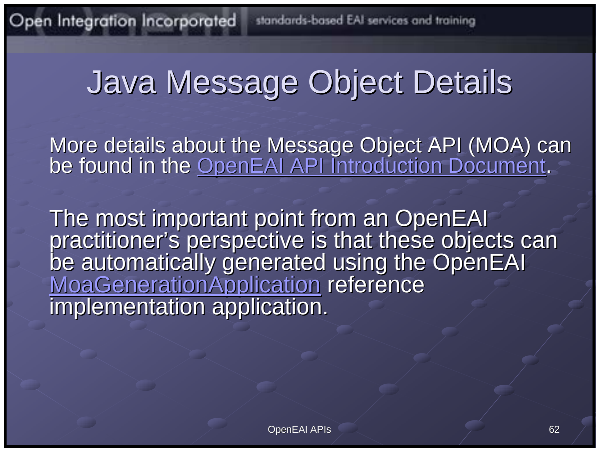

Open Integration Incorporated standards-based EAI software

Other Foundation Components

The following are peripheral OpenEAI foundation components:

- database connection pools (org.openeai.dbpool package)
- thread pools (org.openeai.threadpool package)
- producer pools (org.openeai.jms.producer package)
- scheduled application foundation (org.openeai.afa package)
- XML utilities (org.openeai.xml)

OpenEAI APIs 61

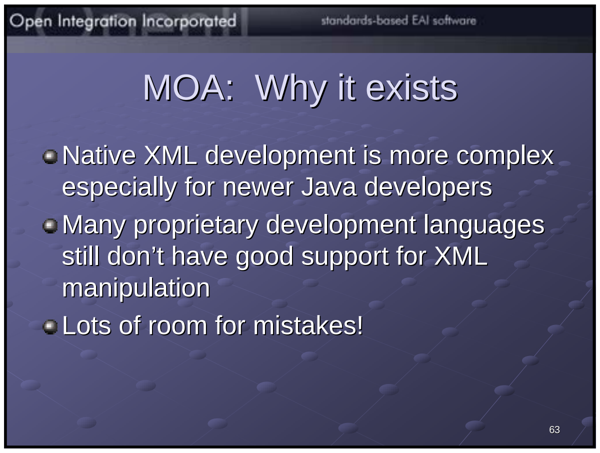

Open Integration Incorporated standards-based EAI services and training

Java Message Object Details

More details about the Message Object API (MOA) can be found in the OpenEAI API Introduction Document.

The most important point from an OpenEAI practitioner's perspective is that these objects can be automatically generated using the OpenEAI MoaGenerationApplication reference implementation application.

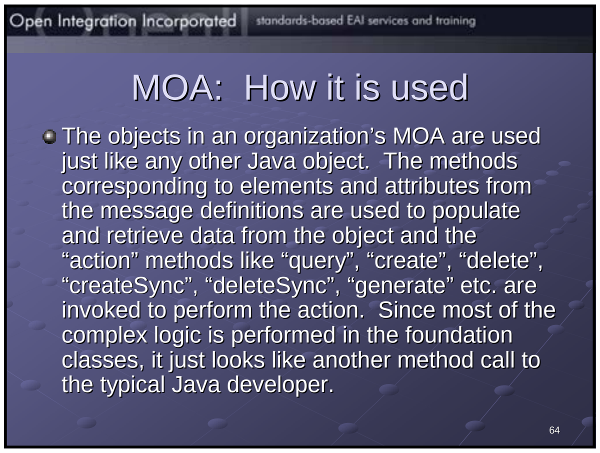
OpenEAI APIs 62


Open Integration Incorporated standards-based EAI software

MOA: Why it exists

- Native XML development is more complex especially for newer Java developers
- Many proprietary development languages still don't have good support for XML manipulation
- Lots of room for mistakes!

63

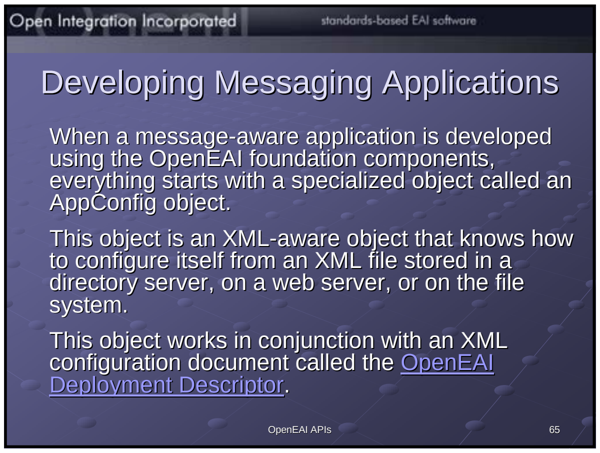


Open Integration Incorporated standards-based EAI services and training

MOA: How it is used

- The objects in an organization's MOA are used just like any other Java object. The methods corresponding to elements and attributes from the message definitions are used to populate and retrieve data from the object and the "action" methods like "query", "create", "delete", "createSync", "deleteSync", "generate" etc. are invoked to perform the action. Since most of the complex logic is performed in the foundation classes, it just looks like another method call to the typical Java developer.

64



Open Integration Incorporated standards-based EAI software

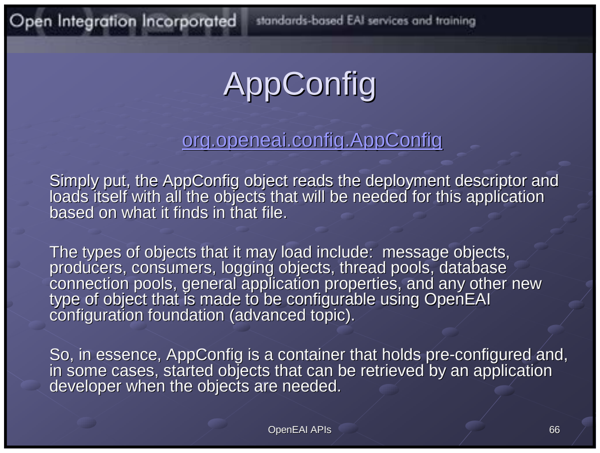
Developing Messaging Applications

When a message-aware application is developed using the OpenEAI foundation components, everything starts with a specialized object called an AppConfig object.

This object is an XML-aware object that knows how to configure itself from an XML file stored in a directory server, on a web server, or on the file system.

This object works in conjunction with an XML configuration document called the OpenEAI Deployment Descriptor.

OpenEAI APIs 65



Open Integration Incorporated standards-based EAI services and training

AppConfig

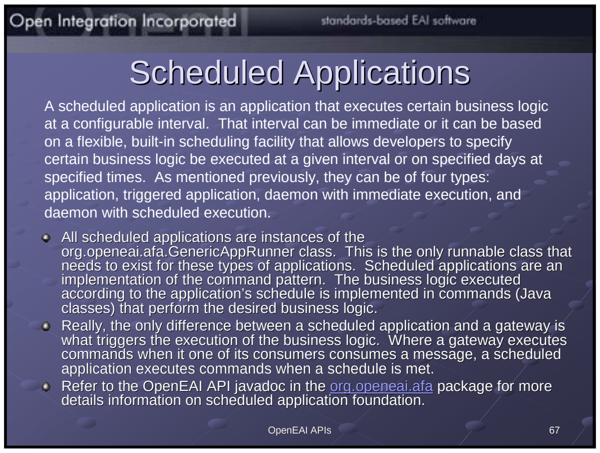
org.openeai.config.AppConfig

Simply put, the AppConfig object reads the deployment descriptor and loads itself with all the objects that will be needed for this application based on what it finds in that file.

The types of objects that it may load include: message objects, producers, consumers, logging objects, thread pools, database connection pools, general application properties, and any other new type of object that is made to be configurable using OpenEAI configuration foundation (advanced topic).

So, in essence, AppConfig is a container that holds pre-configured and, in some cases, started objects that can be retrieved by an application developer when the objects are needed.

OpenEAI APIs 66



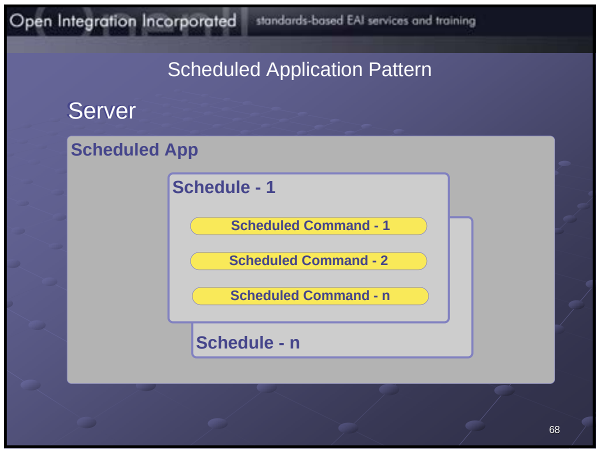
Open Integration Incorporated standards-based EAI software

Scheduled Applications

A scheduled application is an application that executes certain business logic at a configurable interval. That interval can be immediate or it can be based on a flexible, built-in scheduling facility that allows developers to specify certain business logic be executed at a given interval or on specified days at specified times. As mentioned previously, they can be of four types: application, triggered application, daemon with immediate execution, and daemon with scheduled execution.

- All scheduled applications are instances of the org.openeai.afa.GenericAppRunner class. This is the only runnable class that needs to exist for these types of applications. Scheduled applications are an implementation of the command pattern. The business logic executed according to the application's schedule is implemented in commands (Java classes) that perform the desired business logic.
- Really, the only difference between a scheduled application and a gateway is what triggers the execution of the business logic. Where a gateway executes commands when it one of its consumers consumes a message, a scheduled application executes commands when a schedule is met.
- Refer to the OpenEAI API javadoc in the org.openeai.afa package for more details information on scheduled application foundation.

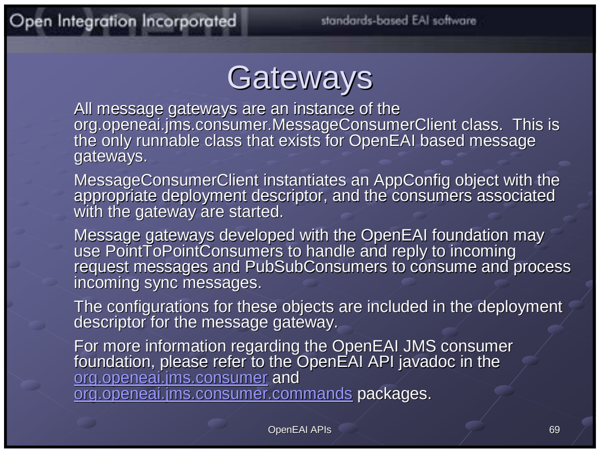
OpenEAI APIs 67



Open Integration Incorporated standards-based EAI services and training

Scheduled Application Pattern

68



Open Integration Incorporated standards-based EAI software

Gateways

All message gateways are an instance of the org.openeai.jms.consumer.MessageConsumerClient class. This is the only runnable class that exists for OpenEAI based message gateways.

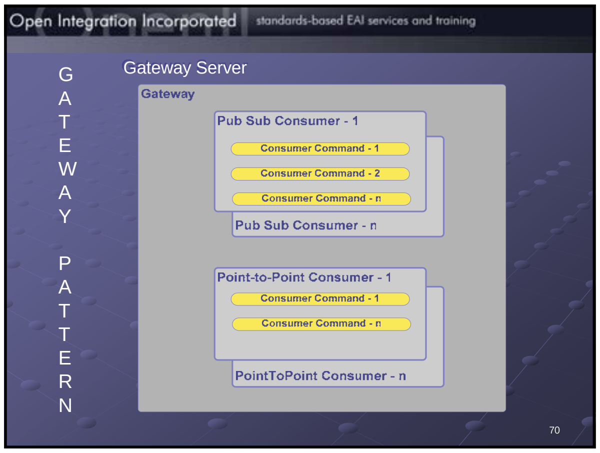
MessageConsumerClient instantiates an AppConfig object with the appropriate deployment descriptor, and the consumers associated with the gateway are started.

Message gateways developed with the OpenEAI foundation may use PointToPointConsumers to handle and reply to incoming request messages and PubSubConsumers to consume and process incoming sync messages.

The configurations for these objects are included in the deployment descriptor for the message gateway.

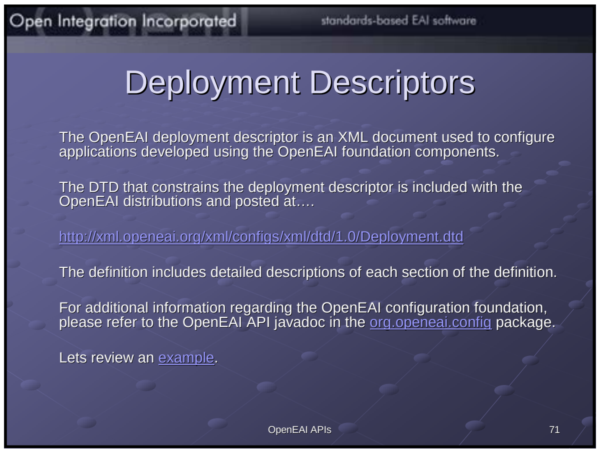
For more information regarding the OpenEAI JMS consumer foundation, please refer to the OpenEAI API javadoc in the org.openeai.jms.consumer and org.openeai.jms.consumer.commands packages.

OpenEAI APIs 69



GATEWAY PATTERN

Gateway Server



Deployment Descriptors

The OpenEAI deployment descriptor is an XML document used to configure applications developed using the OpenEAI foundation components.

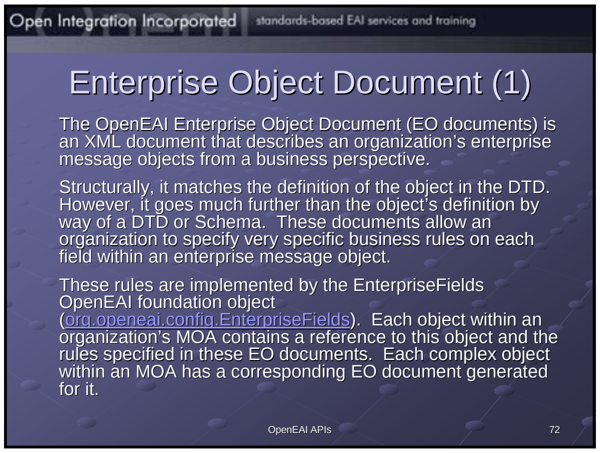
The DTD that constrains the deployment descriptor is included with the OpenEAI distributions and posted at….

http://xml.openeai.org/xml/configs/xml/dtd/1.0/Deployment.dtd

The definition includes detailed descriptions of each section of the definition.

For additional information regarding the OpenEAI configuration foundation, please refer to the OpenEAI API javadoc in the org.openeai.config package.

Lets review an example.

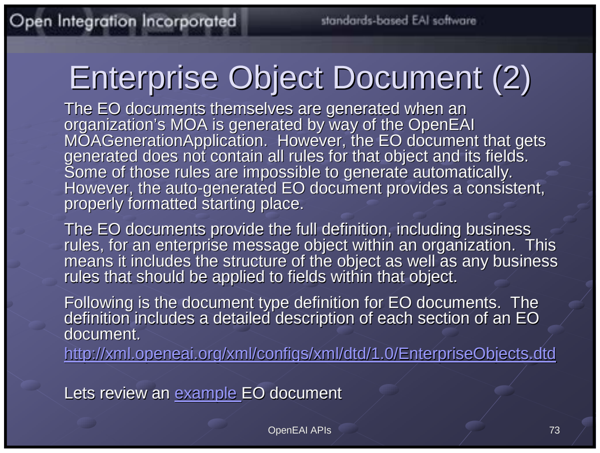


Enterprise Object Document (1)

The OpenEAI Enterprise Object Document (EO documents) is an XML document that describes an organization's enterprise message objects from a business perspective.

Structurally, it matches the definition of the object in the DTD. However, it goes much further than the object's definition by way of a DTD or Schema. These documents allow an organization to specify very specific business rules on each field within an enterprise message object.

These rules are implemented by the EnterpriseFields OpenEAI foundation object (org.openeai.config.EnterpriseFields). Each object within an organization's MOA contains a reference to this object and the rules specified in these EO documents. Each complex object within an MOA has a corresponding EO document generated for it.


Open Integration Incorporated standards-based EAI software

Enterprise Object Document (2)

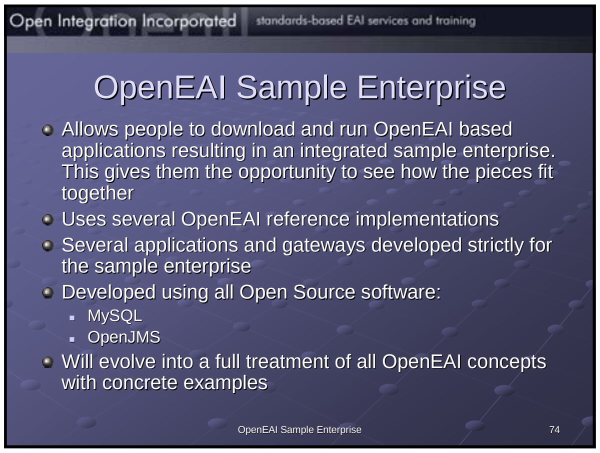
The EO documents themselves are generated when an organization's MOA is generated by way of the OpenEAI MOAGenerationApplication. However, the EO document that gets generated does not contain all rules for that object and its fields. Some of those rules are impossible to generate automatically. However, the auto-generated EO document provides a consistent, properly formatted starting place.

The EO documents provide the full definition, including business rules, for an enterprise message object within an organization. This means it includes the structure of the object as well as any business rules that should be applied to fields within that object.

Following is the document type definition for EO documents. The definition includes a detailed description of each section of an EO document.
http://xml.openeai.org/xml/configs/xml/dtd/1.0/EnterpriseObjects.dtd

Lets review an example EO document

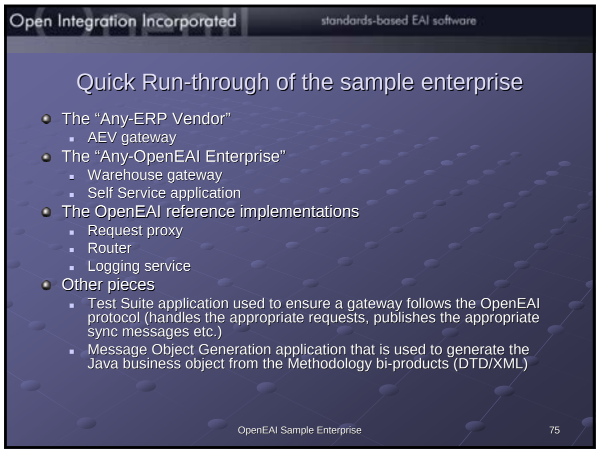
OpenEAI APIs 73


Open Integration Incorporated standards-based EAI services and training

OpenEAI Sample Enterprise

- Allows people to download and run OpenEAI based applications resulting in an integrated sample enterprise. This gives them the opportunity to see how the pieces fit together
- Uses several OpenEAI reference implementations
- Several applications and gateways developed strictly for the sample enterprise
- Developed using all Open Source software:
  - MySQL
  - OpenJMS
- Will evolve into a full treatment of all OpenEAI concepts with concrete examples

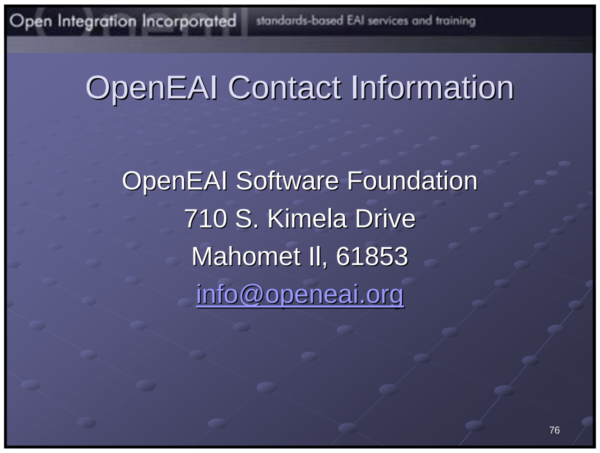
OpenEAI Sample Enterprise 74


Open Integration Incorporated standards-based EAI software

Quick Run-through of the sample enterprise

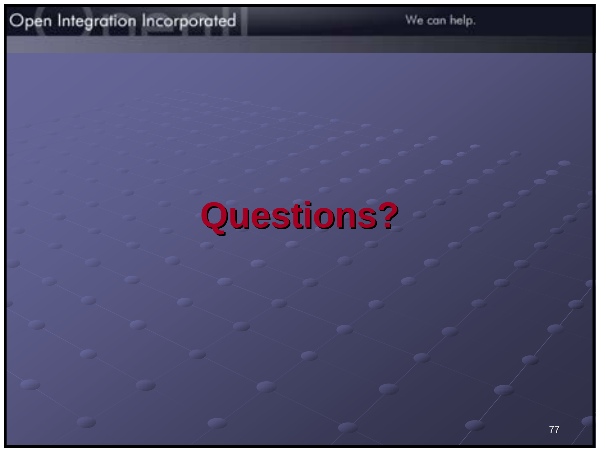
- The "Any-ERP Vendor"
  - AEV gateway
- The "Any-OpenEAI Enterprise"
  - Warehouse gateway
  - Self Service application
- The OpenEAI reference implementations
  - Request proxy
  - Router
  - Logging service
- Other pieces
  - Test Suite application used to ensure a gateway follows the OpenEAI protocol (handles the appropriate requests, publishes the appropriate sync messages etc.)
  - Message Object Generation application that is used to generate the Java business object from the Methodology bi-products (DTD/XML)

OpenEAI Sample Enterprise 75



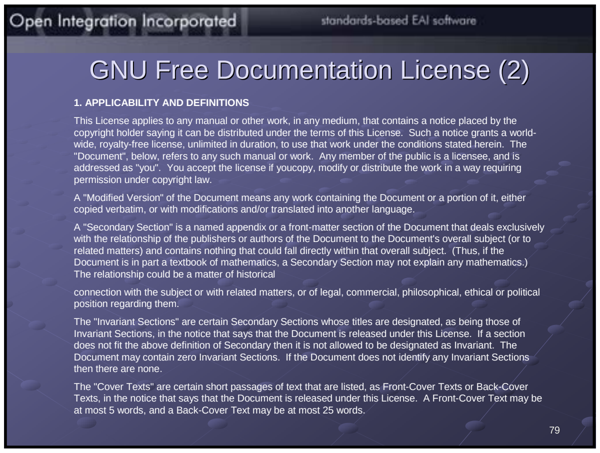
OpenEAI Contact Information

OpenEAI Software Foundation
710 S. Kimela Drive
Mahomet Il, 61853
info@openeai.org



Questions?

GNU Free Documentation License (1)



Open Integration Incorporated standards-based EAI software

GNU Free Documentation License (2)

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if youcopy, modify or distribute the work in a way requiring permission under copyright law.

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.

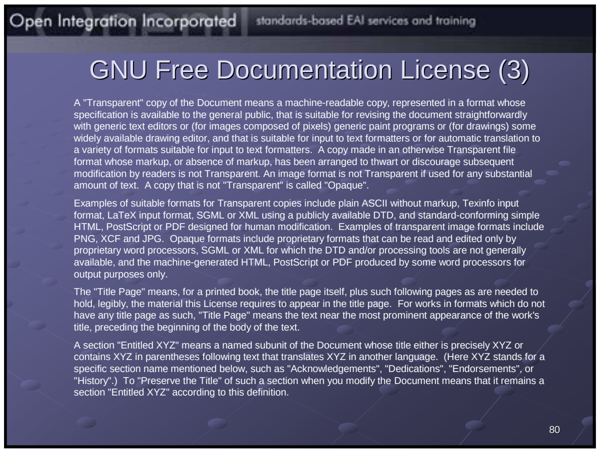
A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, 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. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

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 Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

79



Open Integration Incorporated standards-based EAI services and training

GNU Free Documentation License (3)

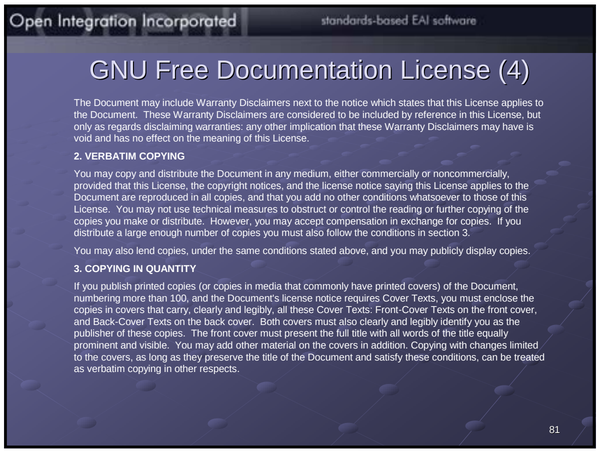
A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for 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, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. 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, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include 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, PostScript or PDF 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.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

80



Open Integration Incorporated standards-based EAI software

GNU Free Documentation License (4)

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

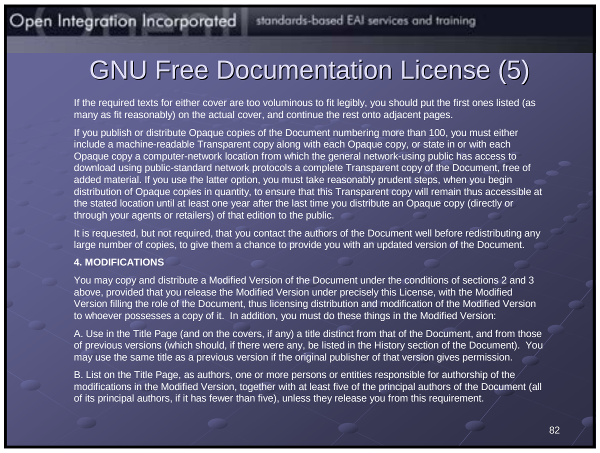
You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those 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 (or copies in media that commonly have printed covers) 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.

81



GNU Free Documentation License (5)

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 computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. 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.

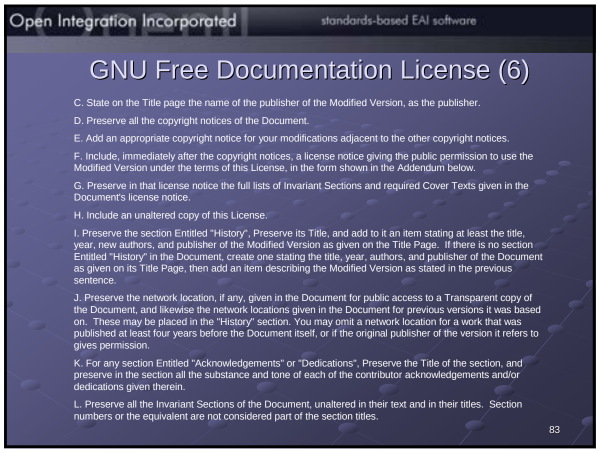
It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

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 fewer than five), unless they release you from this requirement.



GNU Free Documentation License (6)

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.

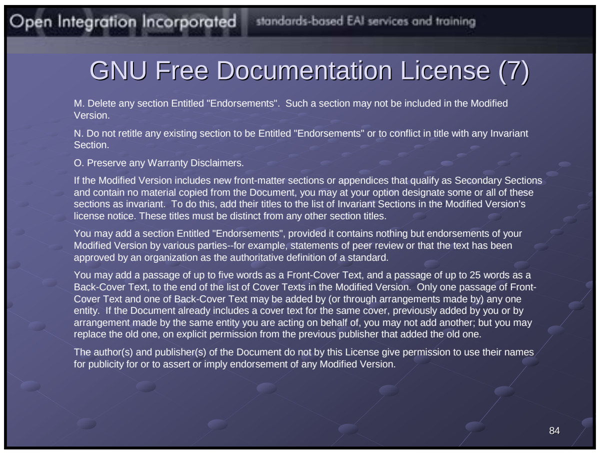
H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve 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. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, 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.



GNU Free Documentation License (7)

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

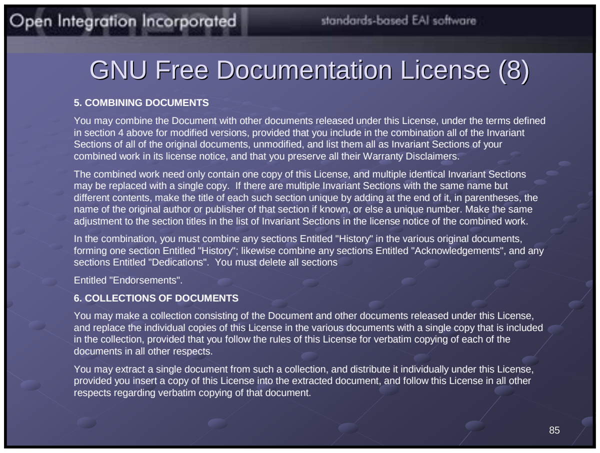
O. Preserve any Warranty Disclaimers.

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.


Open Integration Incorporated standards-based EAI software

GNU Free Documentation License (8)

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, and that you preserve all their Warranty Disclaimers.

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

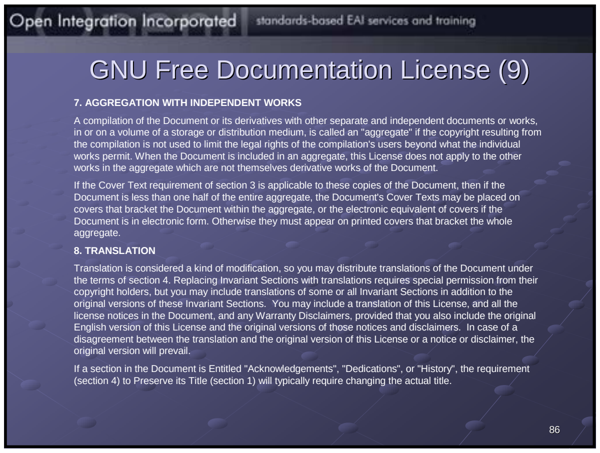
Entitled "Endorsements".

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.

85


Open Integration Incorporated standards-based EAI services and training

GNU Free Documentation License (9)

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, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

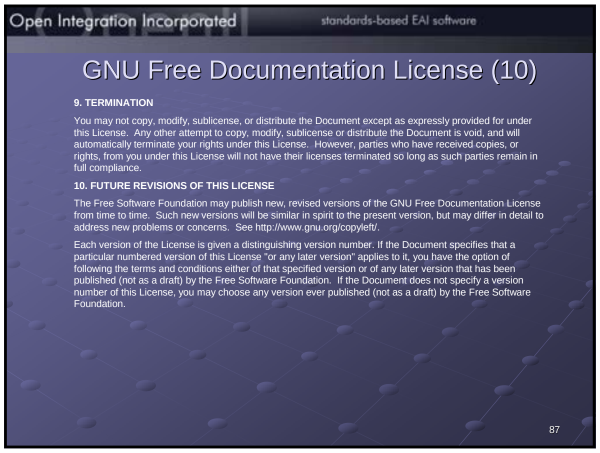
If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket 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, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

86


Open Integration Incorporated standards-based EAI software

GNU Free Documentation License (10)

9. TERMINATION

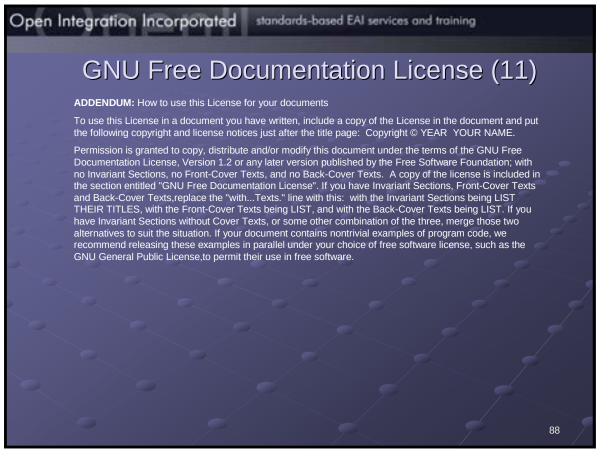
You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies,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

87



Open Integration Incorporated standards-based EAI services and training

GNU Free Documentation License (11)

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page: Copyright © YEAR YOUR NAME.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License". If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts,replace the "with...Texts." line with this: with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation. If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License,to permit their use in free software.

88