

Cross-Referencing for Master Data Management with Oracle Application Integration Architecture Foundation Pack

An Oracle White Paper

Summary	3
Oracle Application Integration Architecture.....	3
Process Integration Packs.....	4
Foundation Pack	4
Master Data Management Overview.....	4
MDM with One-Way Synchronization	4
MDM with Two-Way Synchronization.....	5
Cross-Referencing for Master Data Management.....	5
Cross-Referencing in the Hub / Application Layer	5
Cross-Referencing in the Application Integration Architecture	6
Scenarios	7
Scenario A: Cross-Referencing in the AIA Layer Only	7
Characteristics.....	7
Maintenance of Cross-References	8
Scenario B: Cross-Referencing in the Hub and AIA Layer.....	8
Characteristics.....	8
Maintenance of Cross-References	10
Scenario C: Cross-Referencing in the Hub Layer Only	11
Characteristics.....	11
Maintenance of Cross-References	12
Recommendations.....	12

SUMMARY

Cross-referencing is a key concept when integrating systems and applications. It enables an integration layer to identify correlated information across independent applications and, therefore, is required in almost every integration scenario. Cross-referencing is usually implemented as a generic service on the integration layer level and the same is true for Oracle Application Integration Architecture (AIA).

In the context of Master Data Management (MDM), there is also a cross-referencing concept on an application level within the data hubs to identify source systems for master records such as customers or products.

The arising question is when to use which kind of cross-referencing capabilities or even both at the same time. This whitepaper will discuss the possible scenarios in the MDM context when integrating with Oracle AIA Foundation Pack and will come up with recommendations for consistent and manageable implementations.

ORACLE APPLICATION INTEGRATION ARCHITECTURE

With companies relying on many different application systems to manage mission-critical business functions, application landscapes today are complex and rigid infrastructures. In order for IT to become a strategic partner to the business – helping to drive innovation, growth, and competitive advantage – creating better alignment across applications through flexible, adaptable business processes integrations is mandatory.

To help organizations achieve their business goals, Oracle has developed a new approach to unifying applications, Oracle Application Integration Architecture.

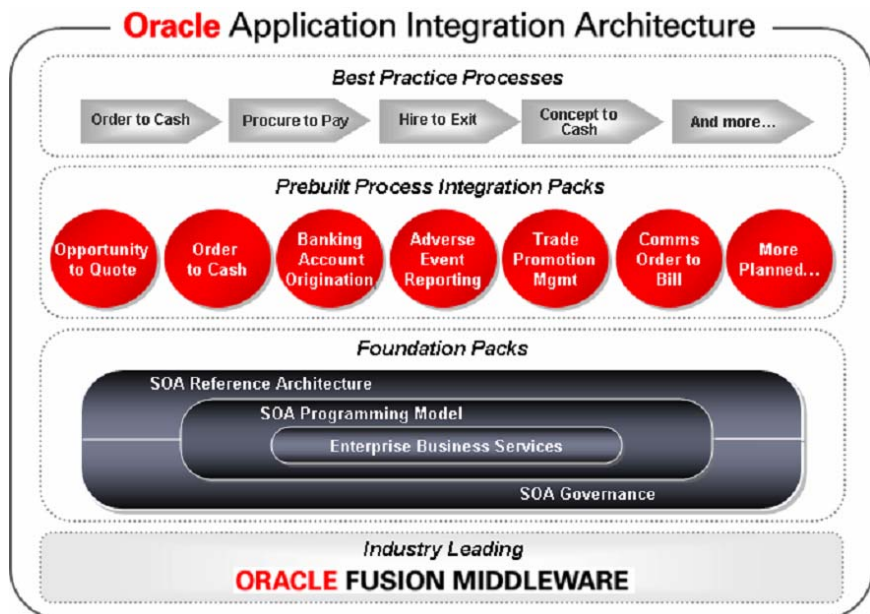


Figure 1: Oracle Applications Integration Architecture

Oracle Application Integration Architecture is a comprehensive set of products that delivers sustainable business-process-based integrations across Oracle, third-party, and custom applications.

Process Integration Packs

Oracle Process Integration Packs deliver prepackaged, end-to-end business process integrations across Oracle Applications. Based on the Application Integration Architecture Foundation Pack, these out-of-the-box process integrations include everything needed to rapidly enable service-oriented applications, from business processes to common objects and services to SOA Governance, at greatly reduced cost. These process integrations are designed to be easily extended and evolved as an organization's business changes, allowing you to respond to customer and market requirements with greater agility and flexibility.

Foundation Pack

Oracle Application Integration Architecture Foundation Pack helps create custom process integrations more quickly and cost-effectively than traditional approaches. Every business is going to have unique integration requirements, whether it is a non-standard integration or an integration with a custom-developed solution, there will be a need to develop individual integrations.

To ensure integration success, Oracle Application Integration Architecture Foundation Pack provides a proven reference architecture and reusable web services. This solution includes our application-independent objects, services, methodology, and tools, allowing the creation of individual Process Integration Packs leveraging a service-oriented architecture.

MASTER DATA MANAGEMENT OVERVIEW

This section will quickly give an overview about typical implementation patterns for master data management (MDM). Mainly, we will distinguish between implementations requiring one-way or two-way synchronization between the hub and the connected applications.

MDM with One-Way Synchronization

If a master data hub is designed in a way that the management of master data only happens within the hub, a one-way synchronization is fully sufficient. In this case, the hub propagates created or updated records to the middleware for further distribution to target systems. As an example, Product Data Hubs are usually implemented in this way. Also, Customer Data Hubs are sometimes implemented following this pattern when the maintenance of the customer information is only allowed in the master hub and the connected applications are pure consumers of the information provided by the hub.

MDM with Two-Way Synchronization

Whenever the maintenance of master data is also possible in the connected applications, a two-way synchronization between these applications and the hub is required to synchronize master information. If a change happens in the hub itself, the synchronization happens in the same way as for one-way synchronization. If the change happens in one of the so-called sources, the following needs to happen:

1. The application sends the created or updated record to the hub.
2. The hub decides how to handle the change based on configured rules.
3. The new or updated “golden record” is propagated to all connected or relevant applications. This also includes the application where the change happened.

Typical examples for this implementation pattern are customer hubs (UCM or CDH).

CROSS-REFERENCING FOR MASTER DATA MANAGEMENT

This section will quickly summarize the main characteristics of cross-referencing in the hub and AIA layer.

Cross-Referencing in the Hub / Application Layer

There are obvious functional requirements to have cross-references within a data hub such as a Customer Data Hub or a Product Data Hub.

The data hub is the place where information from different systems are validated and consolidated to the so-called “single blended record.” This happens both during initial data loads when setting up a new hub and also during the regular maintenance of data when the spoke systems are allowed to create or change master data.

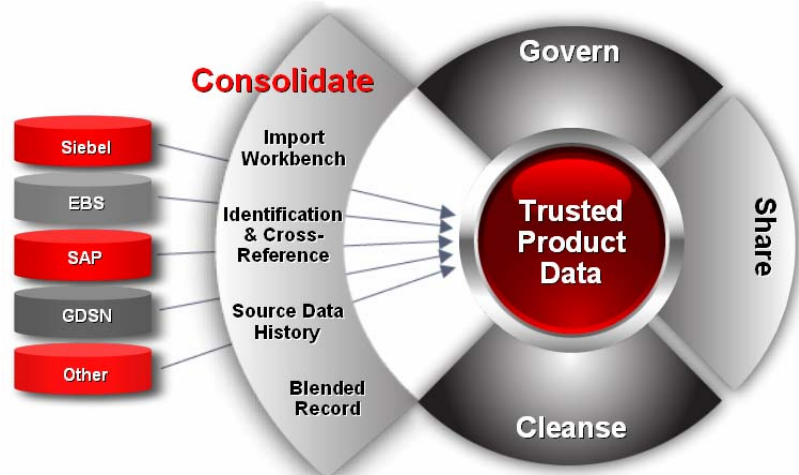


Figure 2: Cross-referencing within a data hub (Product Data Hub)

To manage this consolidation process, information about the source of particular records or partitions of records must be stored. This requires the hubs to define source systems, as well as criteria about how to handle information coming from each particular source. With this information, the hub is able to track which piece of information came from which source system. The key information here are the cross-references to the base records in the source systems.

Having this in place, the hub can also apply predefined rules about how to handle data from source systems. For example, predefined precedence rules for the case of conflicts in a customer data hub.

Cross-Referencing in the Application Integration Architecture

For almost every established integration scenario, cross-referencing is a key requirement to map entities between integrated systems. A central cross-referencing table keeps track of the mappings of records and the integration layer needs to take care to maintain the cross-referencing table. For instance, a customer record in the Universal Customer Master (UCM) with key “SEBL_007” can be matched with the customer record with key “EBS_00G” within an instance of the Oracle E-Business Suite.

UCM	E-Business Suite
SEBL_002	EBS_00B
SEBL_007	EBS_00G

Figure 3: Example of cross-referencing in the integration layer

Implementing the AIA approach to have a system-independent representation of data within the application layer requires an additional application-independent key identifying a record, called the common key. With this, the cross-referencing table could look like this:

Common	UCM	E-Business Suite
COMMON_A	SEBL_002	EBS_00B
COMMON_B	SEBL_007	EBS_00G

Figure 4: Example of cross-referencing in AIA

So, from an AIA perspective, a hub is yet another application and, regarding cross-referencing, AIA treats the hub in exactly the same way as it handles any other application.

Within a typical data flow from a requester system to a provider system, cross-referencing needs to happen twice:

- From the requester application key value to the common value during the transformation from the Application Business Message (ABM) to the canonical Enterprise Business Message (EBM).
- From the common key within the EBM to the provider's application ABM within the transformation in the provider Application Business Connector (ABC) service.

In order to maintain the cross-references and especially to create new mappings, the AIA artifacts need to take care to create the references whenever new records are created. For instance, the AIA artifacts to handle the customer creation within the UCM need to create the cross-references for the new records. Note that this requires a response containing the application's key from all target systems that receive a new record.

Subsequent updates on records that were cross-referenced before in the integration layer can then be handled by querying the cross-referencing table accordingly. Having this in place, the AIA layer has no need to call back to the applications to determine whether a record already exists in a target system or not.

Also note that cross-referencing tables need to be filled initially when establishing new integrations between established systems in order to guarantee successful synchronizations.

SCENARIOS

The following will outline the possible cross-referencing scenarios when integrating a data hub with AIA.

Scenario A: Cross-Referencing in the AIA Layer Only

Characteristics

There are implementation scenarios for which the functionality to have references to the source systems within the Data Hub is not required or not desired. In these cases, only a one-way synchronization from the hub to the target systems is required.

For instance, if changes to customer records only happen within the Customer Hub instance and the information is propagated to connected applications in a one-way synchronization process. In this rare case, there is no functional requirement to maintain cross-references within the data hub.

A more common use case for this pattern are product data hubs, as their design usually allows changes to products only within the hub application and the connected applications are pure consumers of the provided information.

Of course, the integration layer needs to maintain cross-references to be able to map incoming updates from the hubs to the appropriate records in the target systems. Also, during the propagation of new records from the hubs to the target

system, the AIA artifacts need to take care to create new cross-references accordingly.

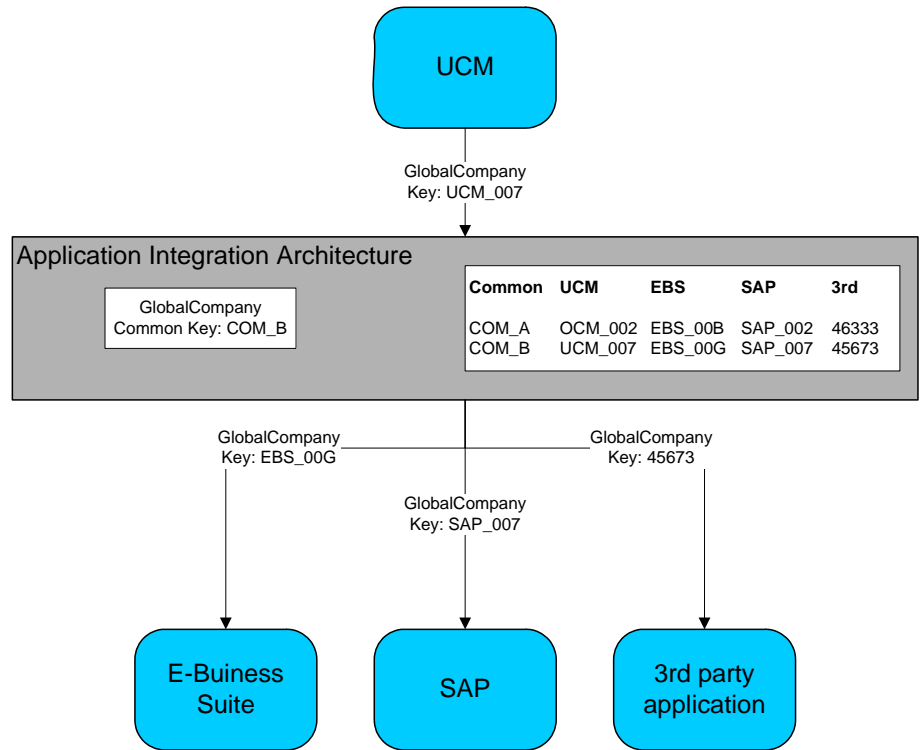


Figure 5: Example of cross-referencing in AIA only

Figure 5 shows a typical case in which a customer record is updated in the UCM application and the changes need to be propagated to three integrated applications. The UCM key identifying the record is matched with the common key to achieve a system-independent representation of the message (EBM). The common key is then referenced against the target-system-specific application keys for E-Business Suite, SAP, and some third-party application.

Maintenance of Cross-References

In the case of creating a new customer record, the target applications feed back the application keys of the newly generated records in the AIA layer and the AIA infrastructure takes care to store the new references in the central cross-referencing table. In particular, the ABC services need to handle the creation of cross-reference entries in AIA. The ABC services for UCM creates the mapping from the UCM key to the common key in AIA. The ABC services for the particular applications need to create the references from the common key to the application keys.

Scenario B: Cross-Referencing in the Hub and AIA Layer

Characteristics

If the hub implementation requires defining source systems and their specific settings, such as precedence rules for changes on particular attributes, cross-referencing can take place both at the hub layer and at the AIA layer.

The cross-referencing information on both layers has overlapping content. While the AIA layer needs to handle cross-references to all integrated systems to be able to appropriately propagate information to them, the hub only cares about systems relevant to the functional hub logic.

Furthermore, it is in the nature of the hub to only care for references in the context of the hub. However, if there is a need for the integration of other business entities, such as orders or invoices, that require the references to the customer data, only cross-referencing in the integration layer can support this as the hub is not involved in these integrations.

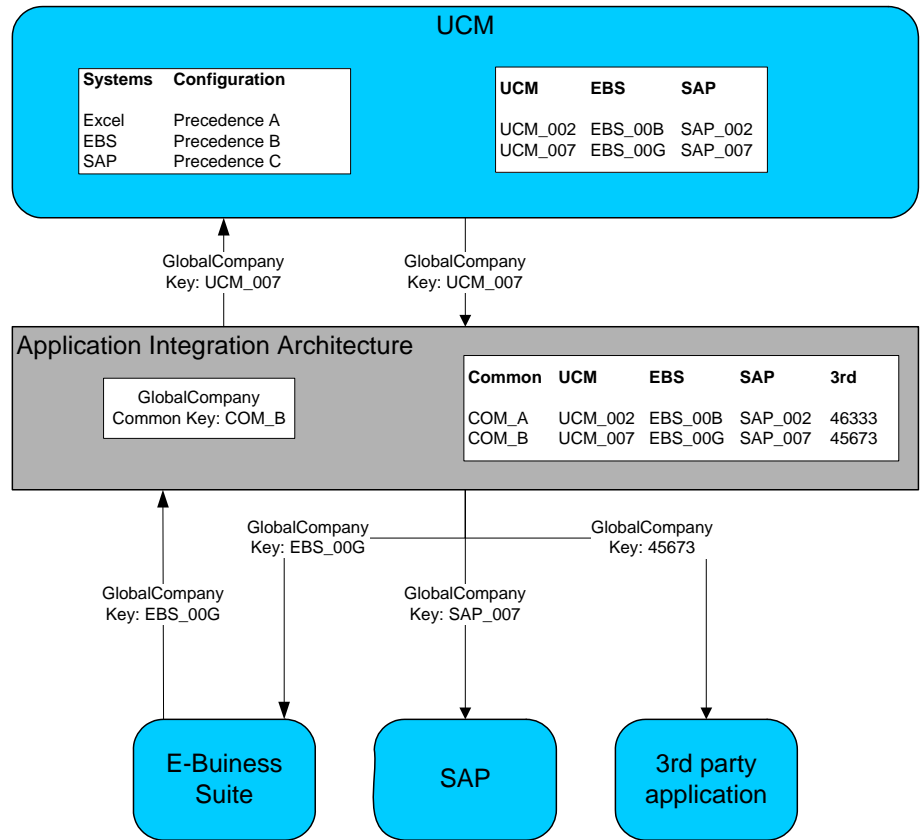


Figure 6: Example of cross-referencing both in AIA and the hub

Figure 6 gives an example of a customer data hub scenario involving cross-referencing both at the hub and in AIA. In this example, changes to customer information can happen in three different places:

- UCM
- E-Business Suite
- SAP

The third-party application is just a subscriber to the changes in the hub; it is not allowed to update customer records itself, it is only consuming the events coming from the hub.

Now, let's assume an update to customer "GlobalCompany" with key EBS_00G happening in the E-Business Suite. This event is captured by the AIA infrastructure and propagated to the UCM. On its way to the UCM, two cross-referencing actions occur when transforming the EBS key EBS_00G to the canonical representation in the EBM (COM_B) and then to the UCM-specific ABM (UCM_007).

Depending on the setup in the UCM, the updates coming from EBS can be relevant, partially relevant, or not relevant at all. This decision is based on the UCM configuration that defines which kinds of changes within EBS are also changing the "golden record" within UCM.

Let's consider that this change is relevant to the master record in UCM. This changed information needs to be propagated back to all source systems. In addition, note that a propagation back to the origin of the change (EBS) may be required as the UCM logic defines whether the update is pertinent or not. Obviously, for this propagation to the target systems, the cross-referencing within the AIA layer applies as before.

Maintenance of Cross-References

For the maintenance of the cross-references, the AIA layer needs to maintain the cross-references that are relevant for the integration itself. A subset of these also has a functional meaning and therefore also need to be populated into the hub. So, whenever a new record is created in one of the spoke systems, the new application key is stored in the AIA cross-referencing table, as well as in a respective source record in the hub. By doing this, both layers have accurate cross-referencing information.

In addition to that, mainly customer data hubs have application features that influence the cross-referencing information while maintaining the data in the hub. For instance, the merging and unmerging of customer records have direct impact on the cross-references. In the case of merges, the external references for the merged customer need to be associated with the survivor of the merge. As an example, UCM_002 and UCM_007 are merged and UCM_007 is the survivor in the scenario above. This changes the cross-references in the hub in the following way as the references to UCM_002 are now assigned to the survivor UCM_007:

UCM	E-Business Suite	SAP
SEBL_007	EBS_00B, EBS_00G	SAP_002, SAP_007

These changes in the hub need to be propagated to the AIA layer to be able to keep the cross-references in sync. This can either be done in an event-driven way whenever a merge or a unmerge occurs, or it can also be done in a batch run in certain intervals. It is important to note that the propagation of the changed references is not fully provided out-of-the box by the current hub solutions.

Scenario C: Cross-Referencing in the Hub Layer Only

Characteristics

With the assumption that the hub is able to maintain all cross-references, it is technically possible to enrich messages coming from UCM with all key attributes required for the propagation to all target systems.

In this scenario, the AIA layer needs to consider the keys of the target applications as being a part of the message payload. In this way, AIA would not actively query and maintain any cross-referencing database.

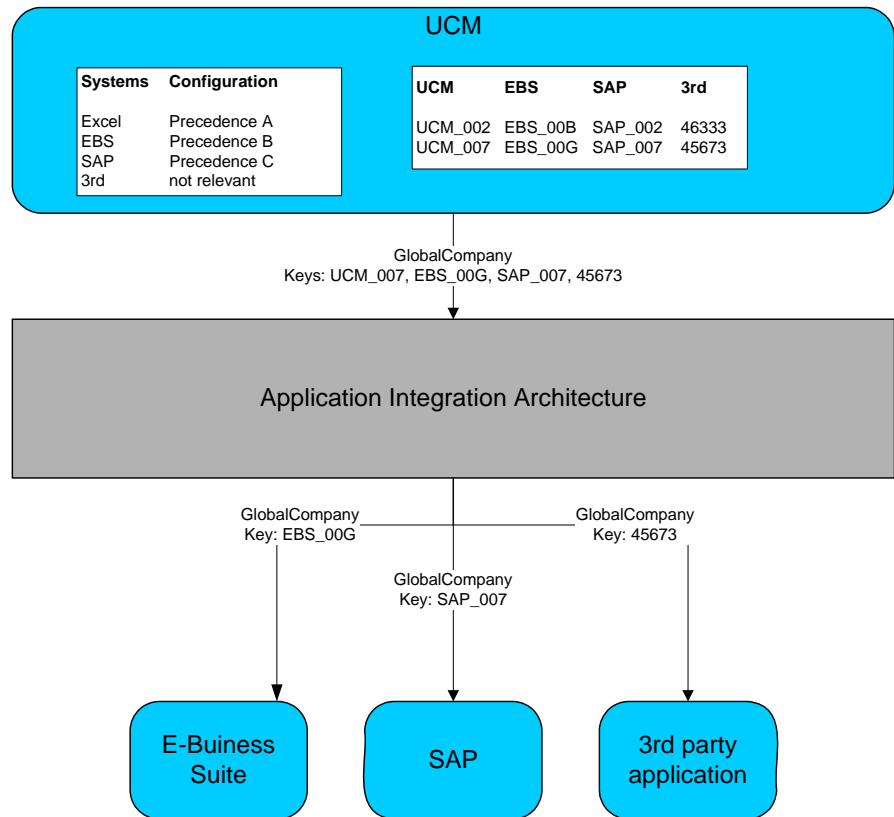


Figure 7: Example of cross-referencing in the application layer only

In the example in Figure 7, an update to a customer record occurs in UCM and the change is propagated to the target systems E-Business Suite, SAP, and some third-party application. The payload contains the key values for these applications and this information must be maintained within the AIA layer in order to be able to construct the specific ABMs for each application containing the respective key.

Note that in this scenario, integration-related configurations must be set up without any application meaning. In our example, the definition of the source system “3rd party” is required to hold cross-references for this application. However, this system is not relevant from a hub perspective as it is a pure consumer of the master data and not allowed to change any customer data itself.

Also, the integration needs to ensure that the hub provides all required keys as a part of any update messages it sends to the integration layer. This could result in the requirement to extend the standard outgoing message structures provided by the hub.

Maintenance of Cross-References

To maintain cross-references in this scenario, the integration needs to ensure that application keys are propagated back to the UCM after the creation of customer records in the spoke systems in the same way as for scenarios A and B.

RECOMMENDATIONS

We clearly recommend following scenarios A or B when implementing Master Data Management with Oracle Application Integration Architecture. Both scenarios are valid approaches for integrating systems with AIA in the MDM context. Depending on the functional requirements in the hub, the hub needs to have a certain set of cross-references that have a functional meaning. Cross-references, being only relevant from an integration perspective, reside only in the AIA layer. Scenario A fully allows loose coupling, as the targets are not known to the hub. Scenario B also allows for loose coupling to a certain extent. Pure consumer applications can be plugged into the architecture at any time without affecting the hub.

We consider scenario C a technically feasible approach that we usually would not recommend for several reasons. Most importantly, integration layer functionality should not be moved into an application. A clear distinction between applications and integration is a prerequisite for a flexible and adoptable architecture. By using an application for an integration purpose such as cross-referencing, the landscape becomes harder to maintain and difficult to adopt to changes such as the replacement of an application. Also, loose coupling becomes impossible in this approach, as the hub must be aware of any connected system.

The following table summarizes the advantages and disadvantages of each scenario:

Scenario	Characteristics	Advantages / Disadvantages	Recommended
A	Cross-referencing in AIA	Pro: Cross-referencing only in one place Pro: Allows for loose coupling Con: Only applicable for one-way synchronization	Yes

Scenario	Characteristics	Advantages / Disadvantages	Recommended
		<p>scenarios</p> <p>Con: Merging and unmerging not possible due to lack of information in the hub</p>	
B	Cross-referencing in AIA and the hub	<p>Pro: Clear concept: hub and AIA have the references they require and nothing else</p> <p>Pro: Allows for loose coupling</p> <p>Con: Maintenance of cross-references in two places and synchronization needs to be ensured</p>	Yes
C	Cross-referencing in the hub	<p>Pro: Cross-referencing only in one place</p> <p>Con: Messages structures possibly need to be extended to include reference information</p> <p>Con: Mixtures of concepts (application and integration)</p> <p>Con: Could require “faked” setup within the hub to cover integration-related information and functionality</p> <p>Con: Prevents from loosely coupling systems</p>	Partially

Scenario	Characteristics	Advantages / Disadvantages	Recommended
		Con: Integration between systems for other integration scenarios not possible due to lack of information	



Title

April 2008

Author: Gerhard Drasch

Contributing Authors: Dave Wong

Oracle Corporation

World Headquarters

500 Oracle Parkway

Redwood Shores, CA 94065

U.S.A.

Worldwide Inquiries:

Phone: +1.650.506.7000

Fax: +1.650.506.7200

www.oracle.com

Copyright © 2008, Oracle. All rights reserved.

This document is provided for information purposes only and the contents hereof are subject to change without notice.

This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.