ORACLE

Integrating Siebel 8.0 Applications

Module 4: Integration Objects

4

Module Objectives

After completing this module you should be able to:

- Explain the role of internal and external integration objects
- Create an internal integration object
- Create an external integration object based on an XML schema definition (XSD)

Why you need to know:

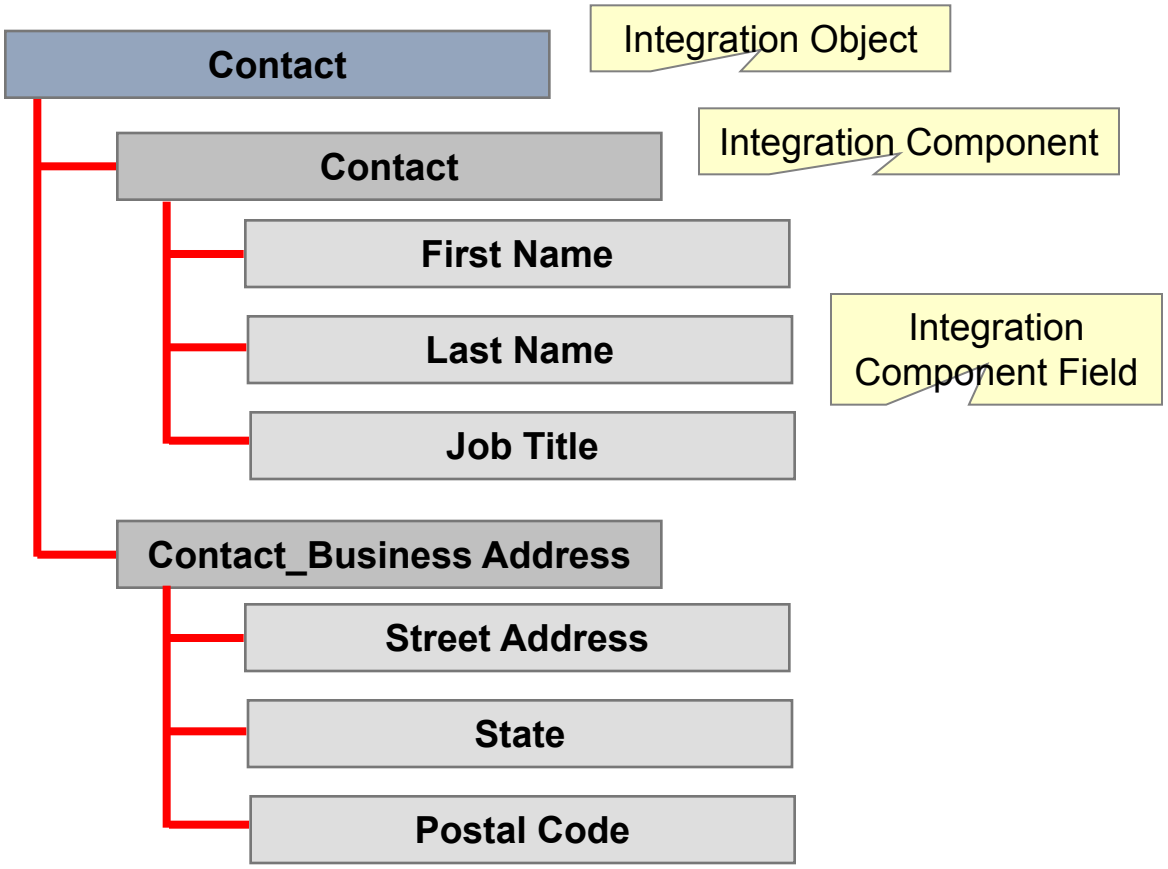
- Integration objects provide the templates for importing and exporting data from a Siebel application

Integration Object

- Defines the structure of data being exchanged between a Siebel and an external application
- Is used as a template to create memory-resident instances of data for processing by business services

Integration Object Continued

- Is designed to represent hierarchical data structures
- Consists of multiple integration components
  - Each integration component consists of multiple integration component fields

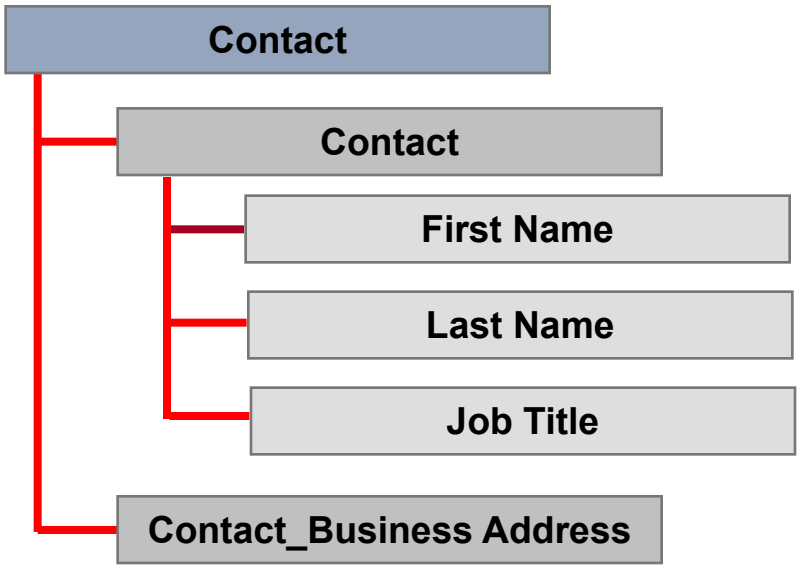


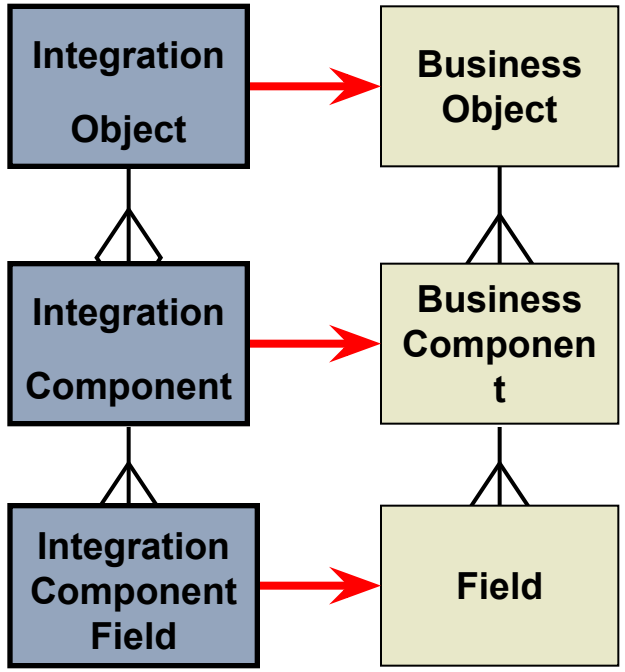
Types of Integration Objects

- Internal integration object
  - Specifies the structure of the data extracted from or inserted into the Siebel application
- External integration object
  - Specifies the structure of the data that is transported between the Siebel and the external application
  - Is required only when the external application is not able to handle data in Siebel format (“Siebel data”)
    - Additional processing is required within the Siebel application to convert the integration object instance

Internal Integration Object

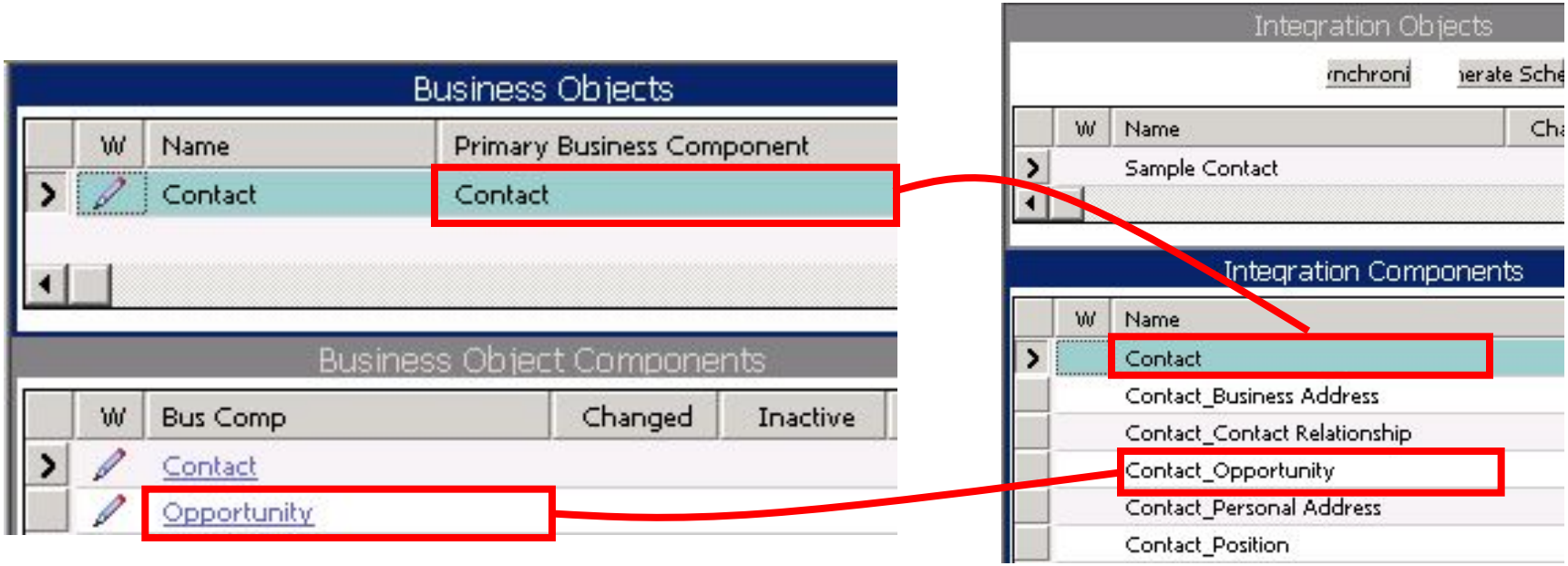
- Represents a subset of a Siebel business object
  - Integration components represent business components
  - Integration component fields represent business component fields





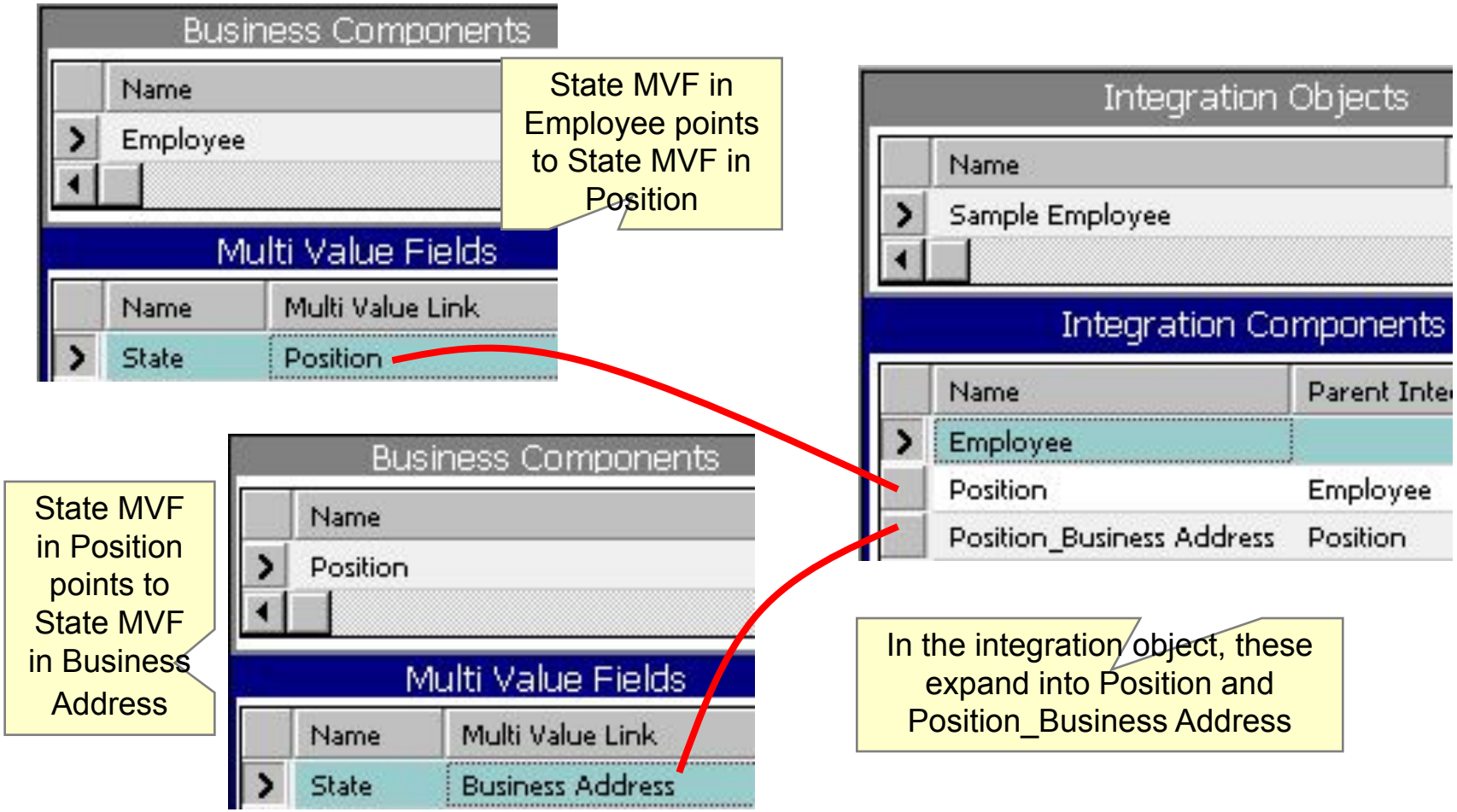
Integration Components

- An integration object includes the following:
  - One parent integration component corresponding to the primary business component in a business object
  - Zero or more child integration components corresponding to other business components in the business object



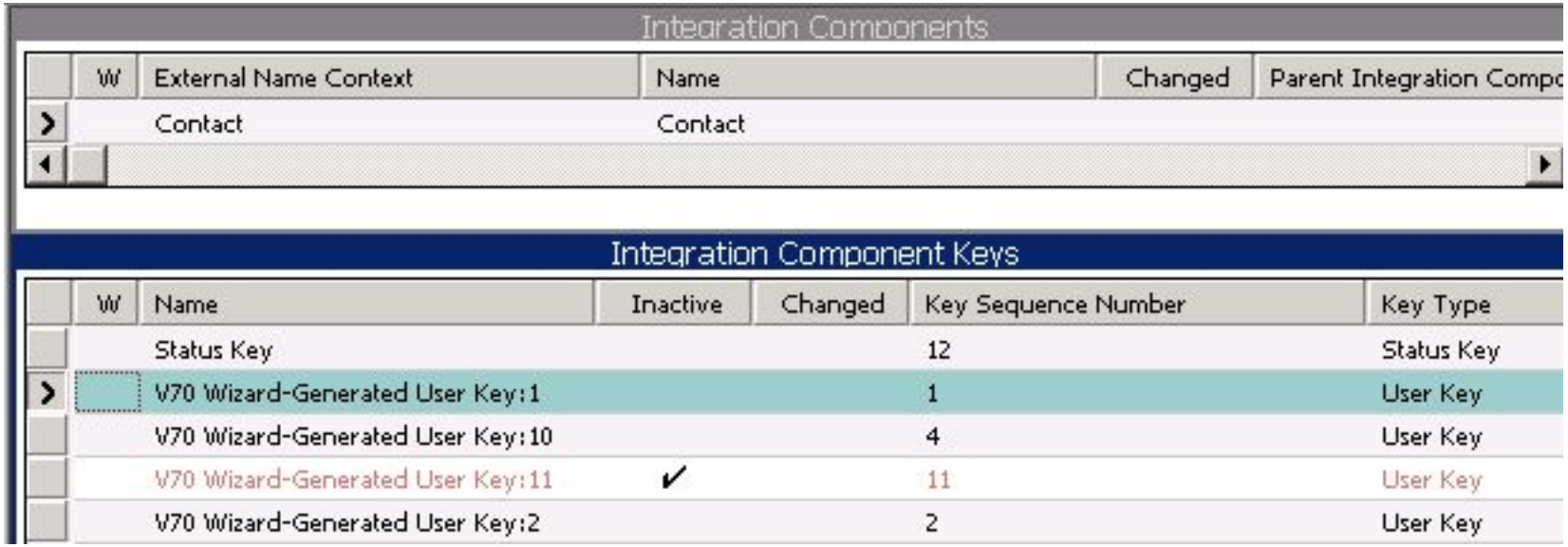
Integration Components Continued

- In addition, a child integration component represents the business component for each MVG in the primary and child business components



Integration Component Key

- Is a field or set of fields that uniquely identify a record being exchanged
- Is based on user keys for the tables referenced by the business components
- Is constructed by the Integration Object Wizard



Integration Components

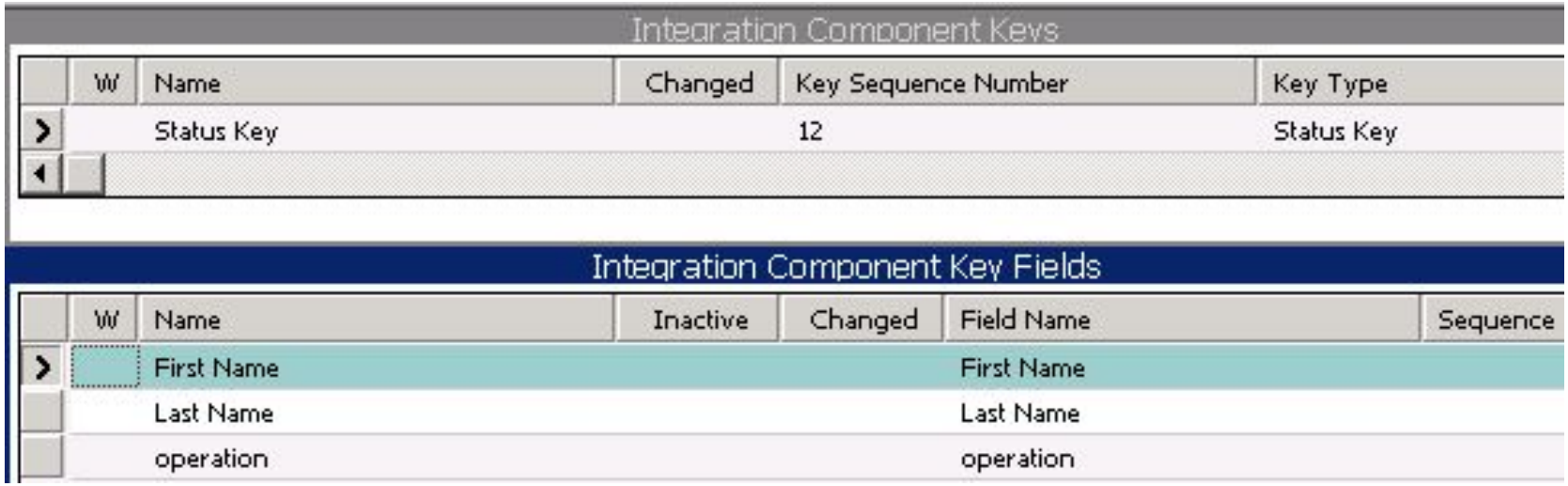
W	External Name Context	Name	Changed	Parent Integration Compo
>	Contact	Contact		

Integration Component Keys

W	Name	Inactive	Changed	Key Sequence Number	Key Type
	Status Key			12	Status Key
>	V70 Wizard-Generated User Key:1			1	User Key
	V70 Wizard-Generated User Key:10			4	User Key
	V70 Wizard-Generated User Key:11	✔		11	User Key
	V70 Wizard-Generated User Key:2			2	User Key

Status Key

- Is a special type of integration component key that is used to return the result of an operation, such as insert or update
  - Type is Status Key
  - Can be used to indicate:
    - Success or failure of an operation
    - Actual operation performed (for example during an upsert)
    - ROW_ID for a newly created record
  - Integration component key fields specify data to be returned when status is requested

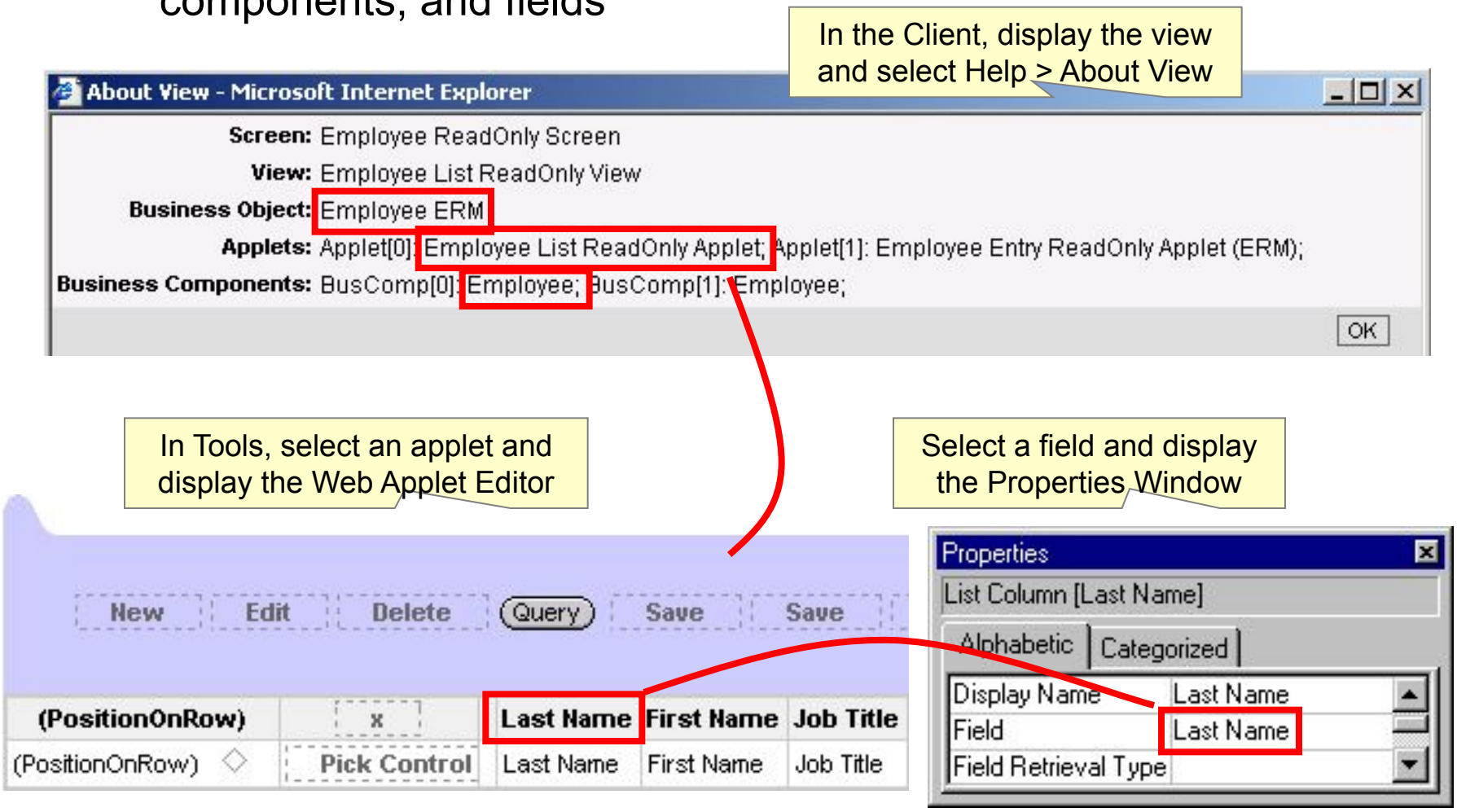


Creating an Internal Integration Object

1. Identify the Siebel Data to Be Exchanged
2. Specify the Business Object
3. Select the Integration Components
4. Inactivate Unneeded Integration Component Fields
5. Inactivate Unneeded Integration Component Keys

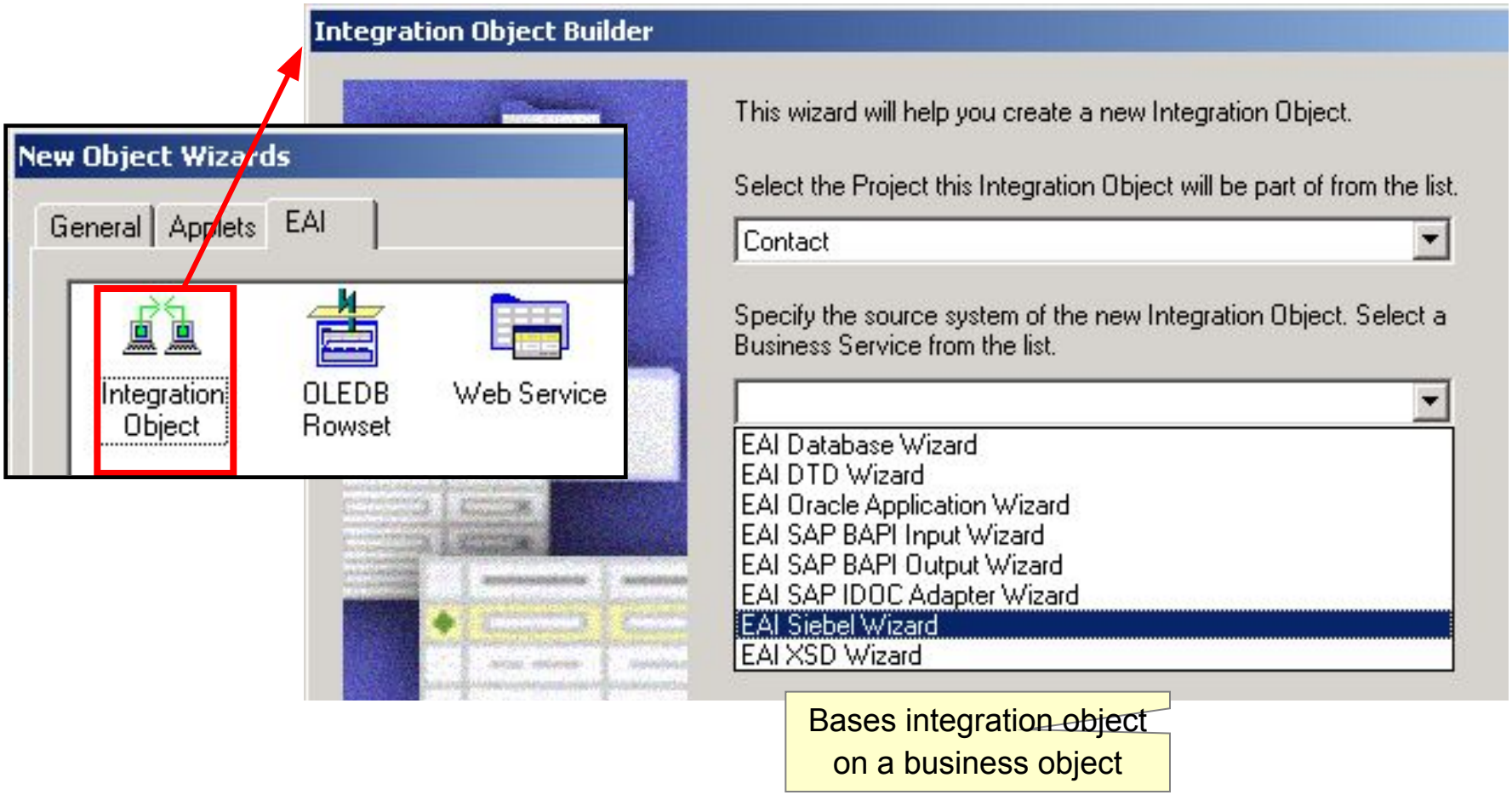
1. Identify the Siebel Data to Be Exchanged

- Identify the data in the Siebel UI to integrate
  - Determine the underlying Siebel business objects, business components, and fields



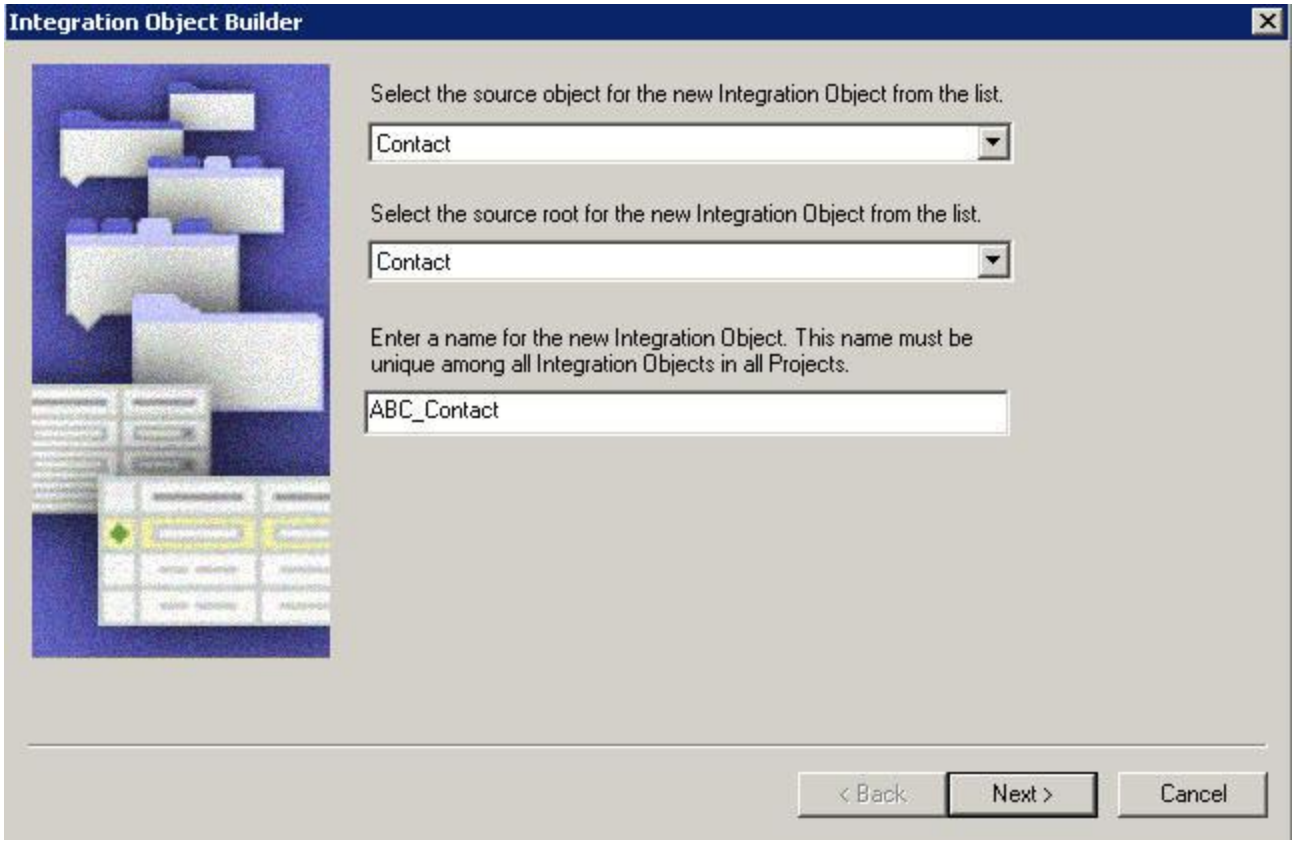
2. Specify the Business Object

- Use the Integration Object Builder to create an integration object based on a Siebel business object
  - Select the EAI Siebel Wizard business service



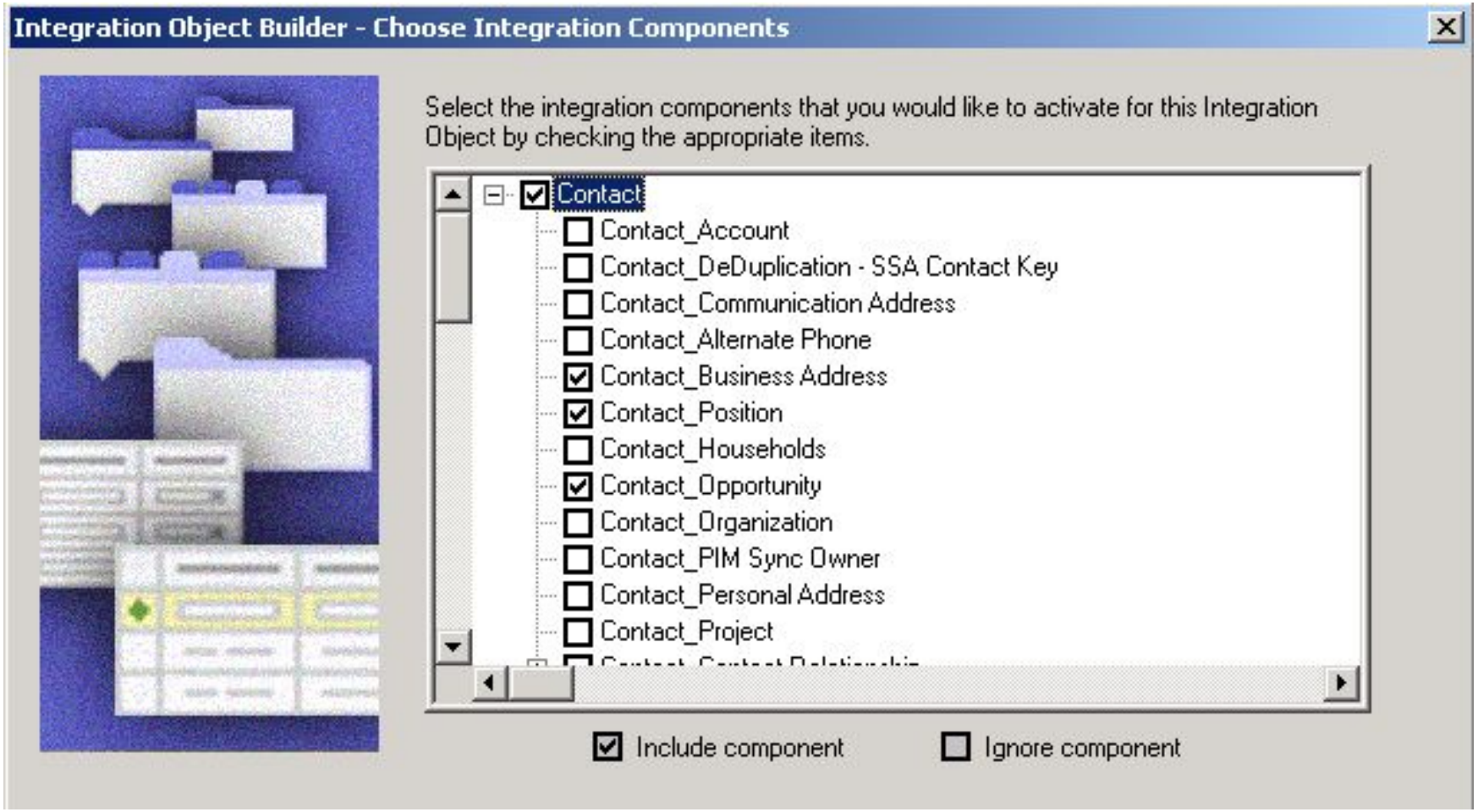
2. Specify the Business Object Continued

- Select the business object
- Provide a name for the integration object



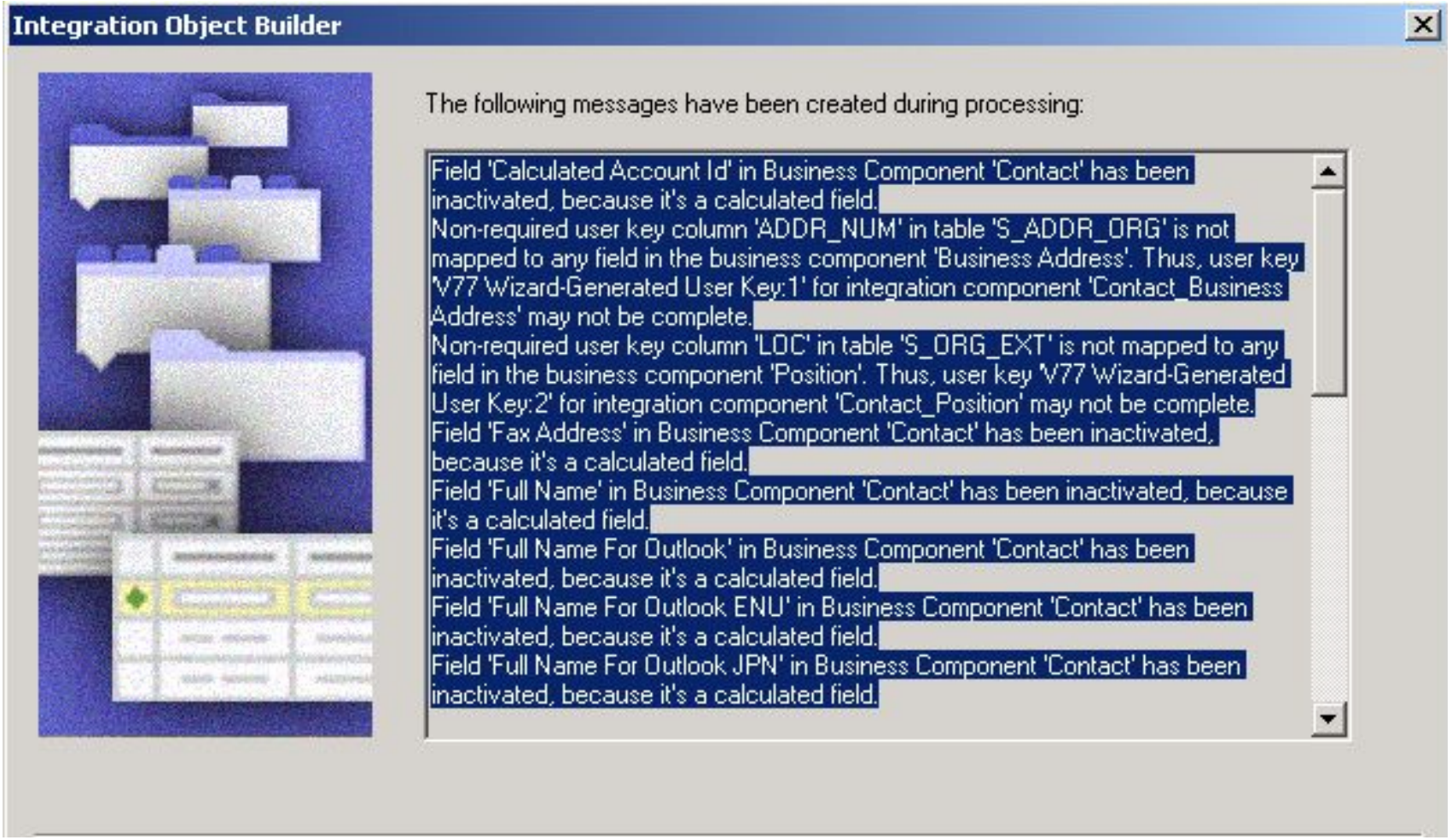
3. Select the Integration Components

- Select components to include in the integration object



3. Select the Integration Components Continued

- Review messages about fields made inactive
- Click Finish to configure the integration object

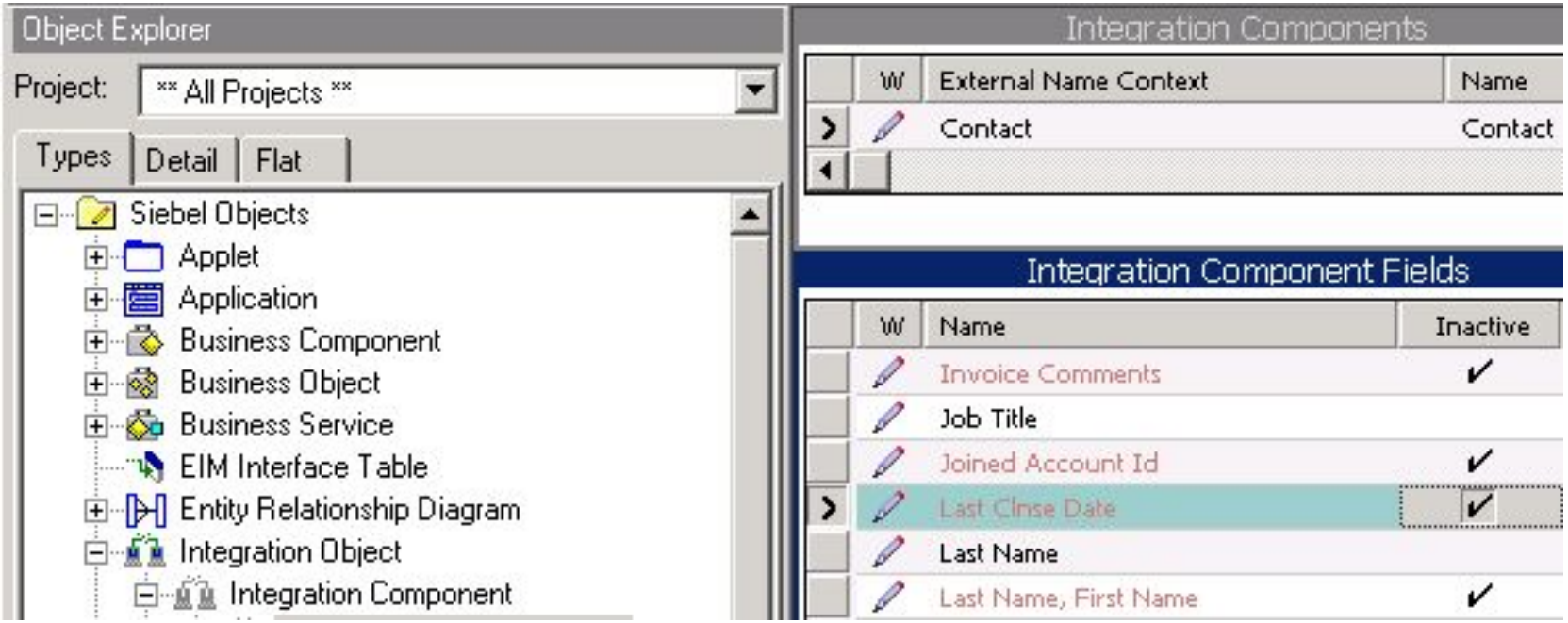


3. Select the Integration Components Continued

- Integration Object Wizard:
  - Finds all links, business components, and underlying tables
  - Identifies all business component fields that map to user keys in tables
  - Creates integration component user keys based on those fields
  - Notes any key columns that are not exposed in a business component
  - Creates calculated fields as integration component fields, but marks them inactive
    - Such fields cannot be updated

4. Inactivate Unneeded Integration Component Fields

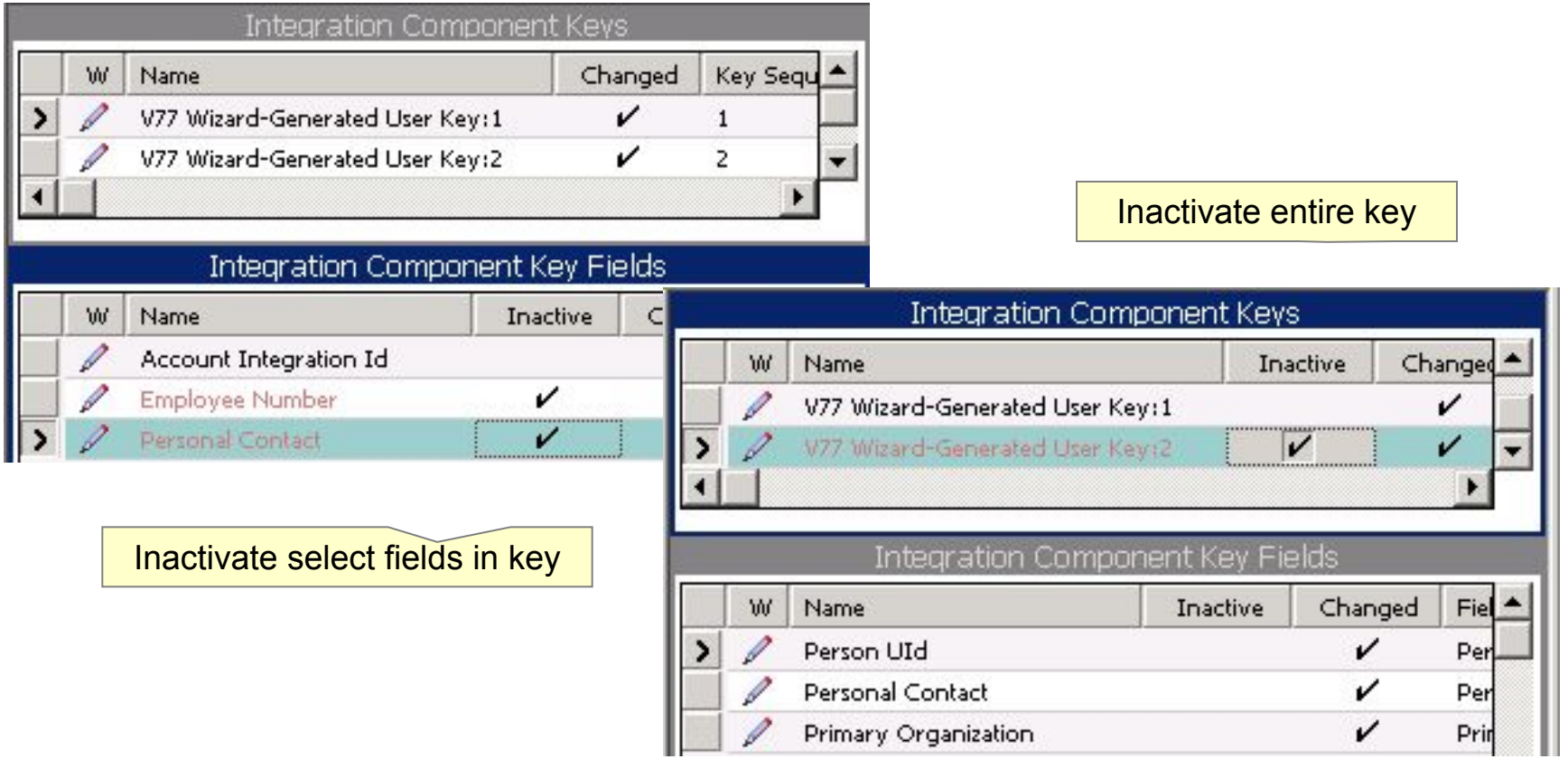
- Set unneeded fields to inactive in each integration component to improve performance
- Do not delete unneeded fields—deleted fields become active when the business object is upgraded



Field will not be included in the integration object

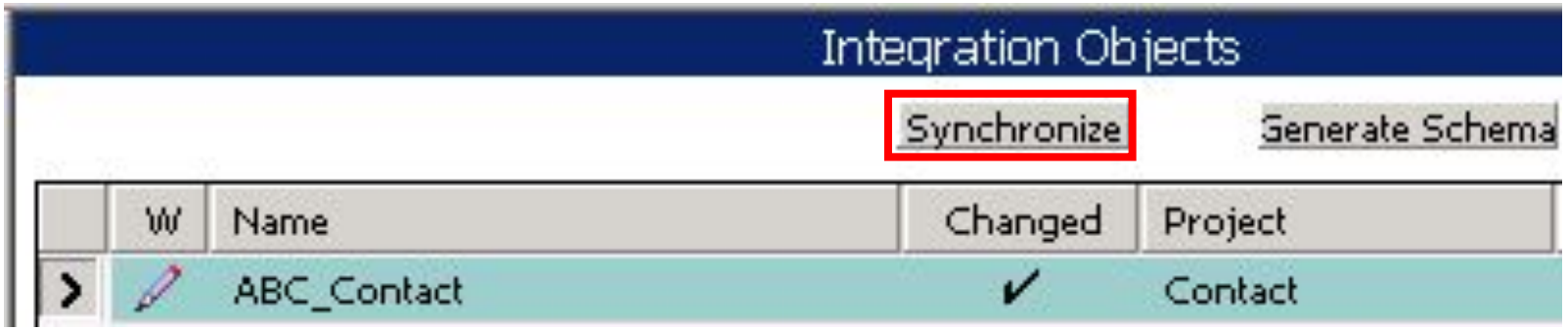
5. Inactivate Unneeded Integration Component Keys

- Make sure that the fields in the integration component keys are consistent with the component fields
  - Inactivate unneeded key fields in each integration component
  - Inactivate a key itself if it has no active fields



Synchronizing Integration Objects

- When business objects are modified or updated, it is necessary to make sure the integration object is still consistent
  - If business object changes are minor, click Synchronize to update the integration object
    - Example: adding a new single value field
  - If business object changes are extensive, delete and recreate the integration object
    - Example: creating a new MVG



Creating an External Integration Object

1. Obtain a Schema of the External Data
2. Create the External Integration Object
3. Select Integration Components
4. Verify the Integration Object

ORACLE

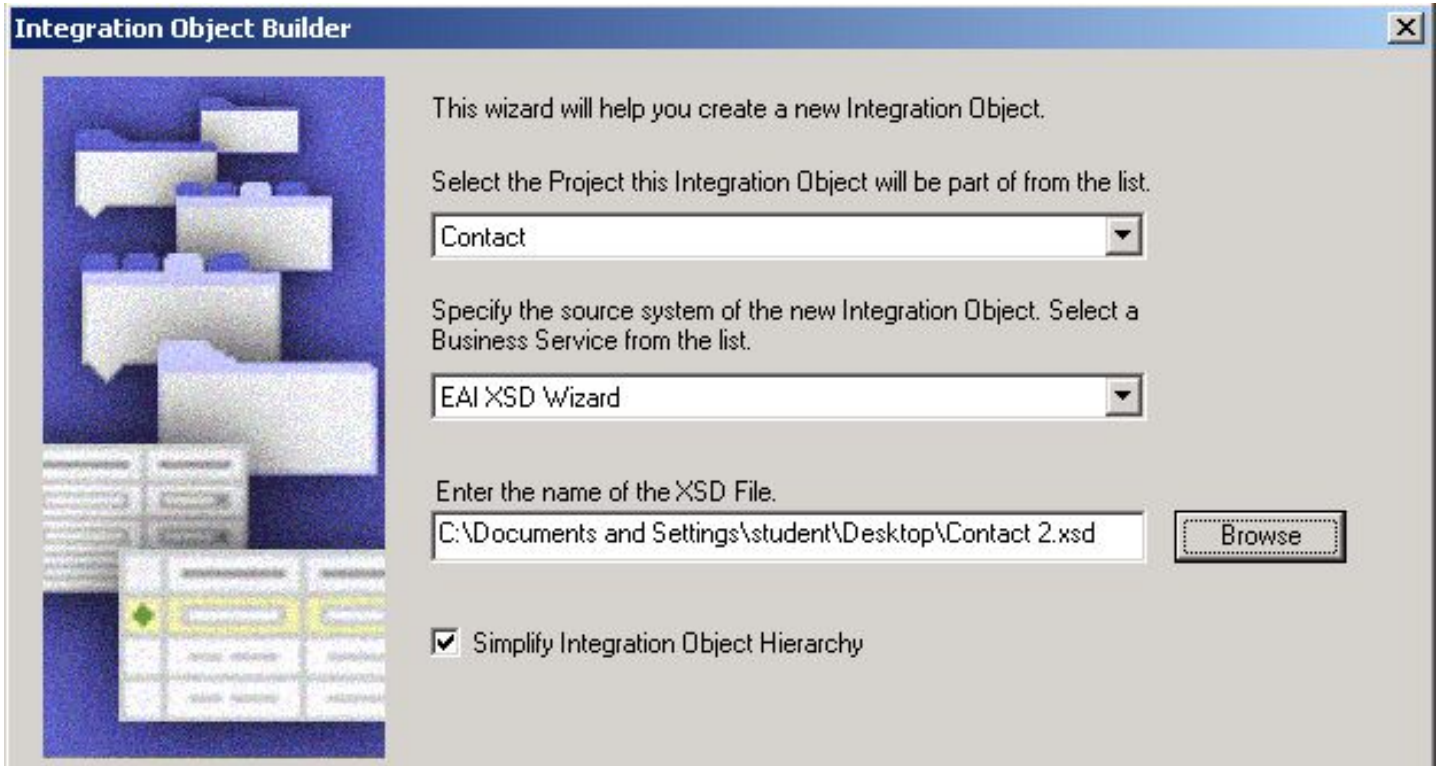
1. Obtain a Schema of the External Data

- Use the schema if published by the external application
- Alternatively, create a file containing a sample XML file received/sent by the external application
  - Use a third-party XML utility to generate the schema as an XSD

```
<?xml version="1.0" encoding="windows-1252" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns="http://www.OU.org"
            targetNamespace="http://www.OU.org"
            elementFormDefault="qualified">
  <xsd:element name="Contact">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="name" type="xsd:string"/>
        <xsd:element name="title" type="xsd:string"/>
        <xsd:element name="email">
          <xsd:complexType>
            <xsd:simpleContent>
              <xsd:extension base="xsd:string"/>
            </xsd:simpleContent>
          </xsd:complexType>
        </xsd:element>
```

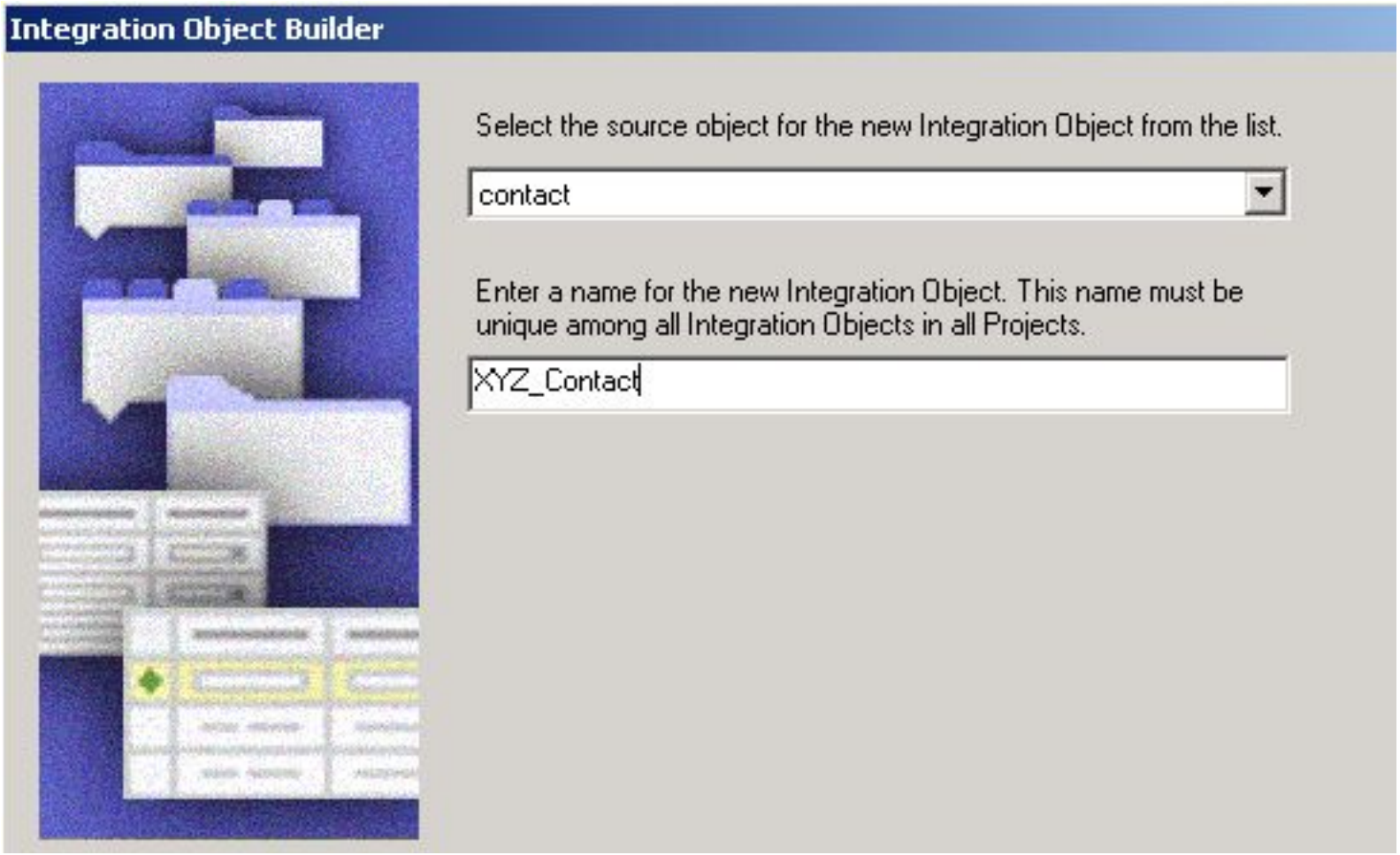
2. Create the External Integration Object

- Use the Integration Object Builder to create an integration object
  - Select the EAI XSD Wizard or EAI DTD Wizard business service
  - Check *Simplify Integration Hierarchy* to create an integration object with leaf elements as fields
    - Otherwise, all leaf elements are converted to integration components



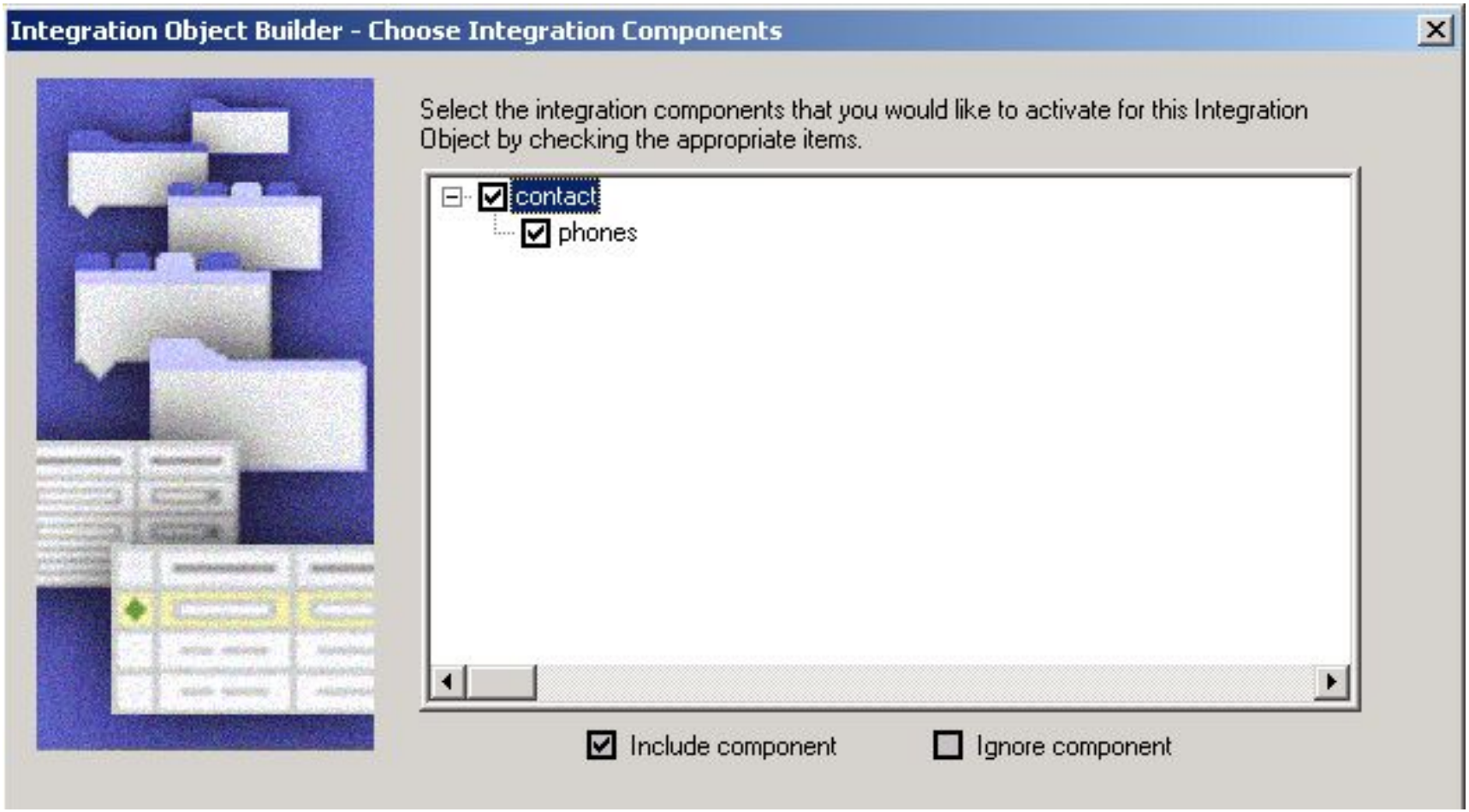
2. Create the External Integration Object Continued

- Specify the source object to serve as the root-level node



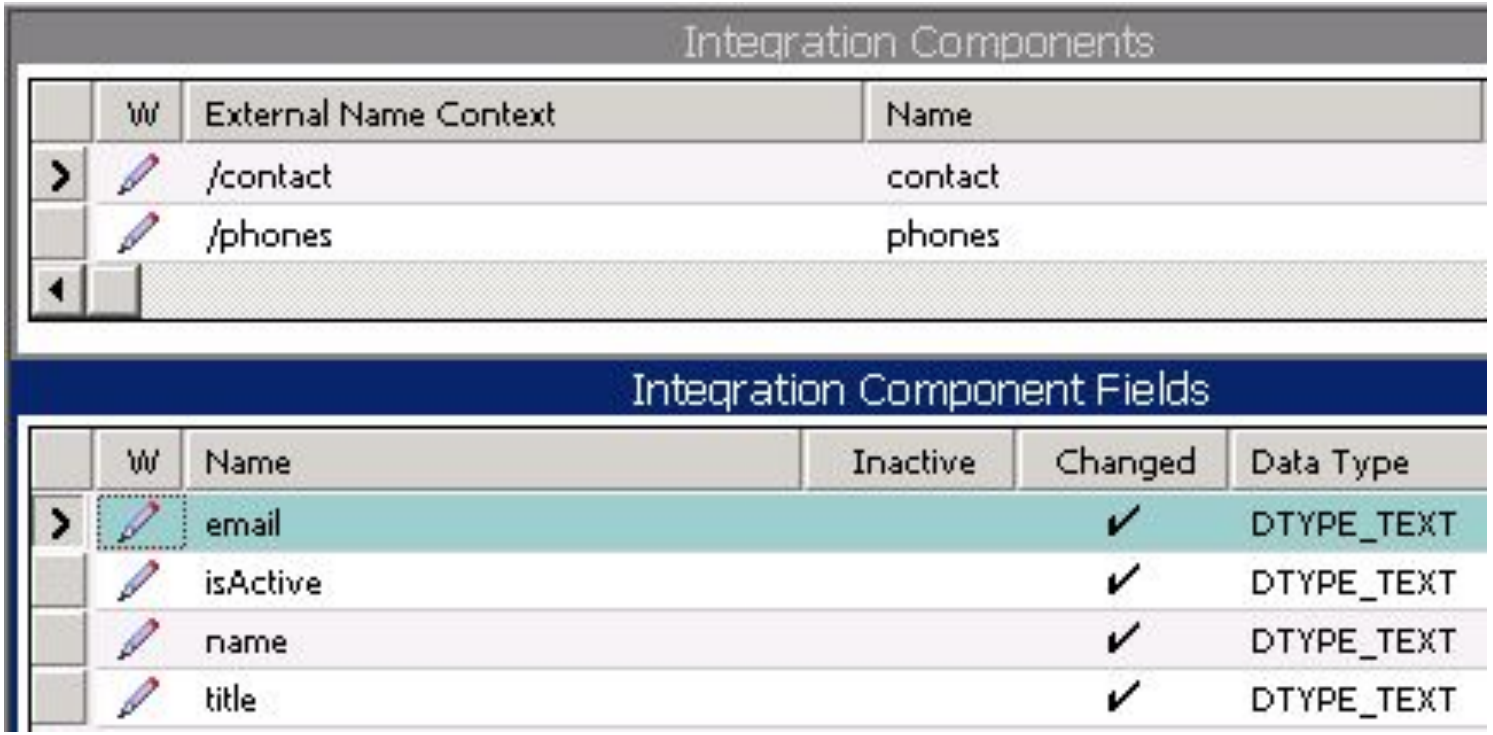
3. Select Integration Components

- Select integration components



4. Verify the Integration Object

- Examine the configured integration object that contains the desired components and fields
- Inactivate unneeded integration component fields as necessary



Module Highlights

- Integration objects define the structure of data being exchanged between a Siebel and an external application
  - Integration objects are used as templates to create memory-resident instances of data for processing by business services
- Internal integration objects are a subset of Siebel business objects
- Use the Siebel Wizard to configure an internal integration object
  - Requires inactivating multiple fields and user keys
- Use the EAI XSD Wizard to build an external integration object based on the schema associated with external data

Lab

- In the lab you will:
  - Create an internal integration object
  - Create an external integration object based on an external .xsd file