Functional Specification for Sequenced Objects, EclipseLink OXM

Project ID: Sequenced Objects

Version: November 15, 2007

Status: Review Draft

Author: Blaise Doughan

Project Manager: David Twelves

Change Log		
Version	**Reviewers**	**Changes**
2007/09/13	OXM Team	Initial Version
2007/11/15	EclipseLink committers	

Review and Approval

The reviewers should have the technical knowledge and authority to state that the project is doing the right thing. If they do not have the authority and knowledge to do this, then they should not be on a review team. (This restriction is for formal technical reviews that determine acceptance. The author is always free to call for informal peer reviews to obtain guidance without approval.) Recommended reviewers are specified by the Product Manager.

If there are external interface (anything in section 4 of this document) there should be at least one approver who will review the spec for manageability issues.

A formal technical review ends with accepting or not accepting the reviewed product. The review team accepts the document when they believe it is right. Right means the product has selected the correct alternatives, is complete, and is worth doing (satisfies requirements). Update the title page "Status" line to <*Approved*> when all issues are resolved and the document is ready for source control.

Table of Contents

1. Project Overview

1.1. Objective

Take the concept of Sequenced Data Objects from SDO 2.1 and expose the concept through EclipseLink OXM in a way that can be used by POJOs. In turn this feature will be used by our SDO implementation to enable our own handling of Sequenced Data Objects.

1.2. Brief Overview

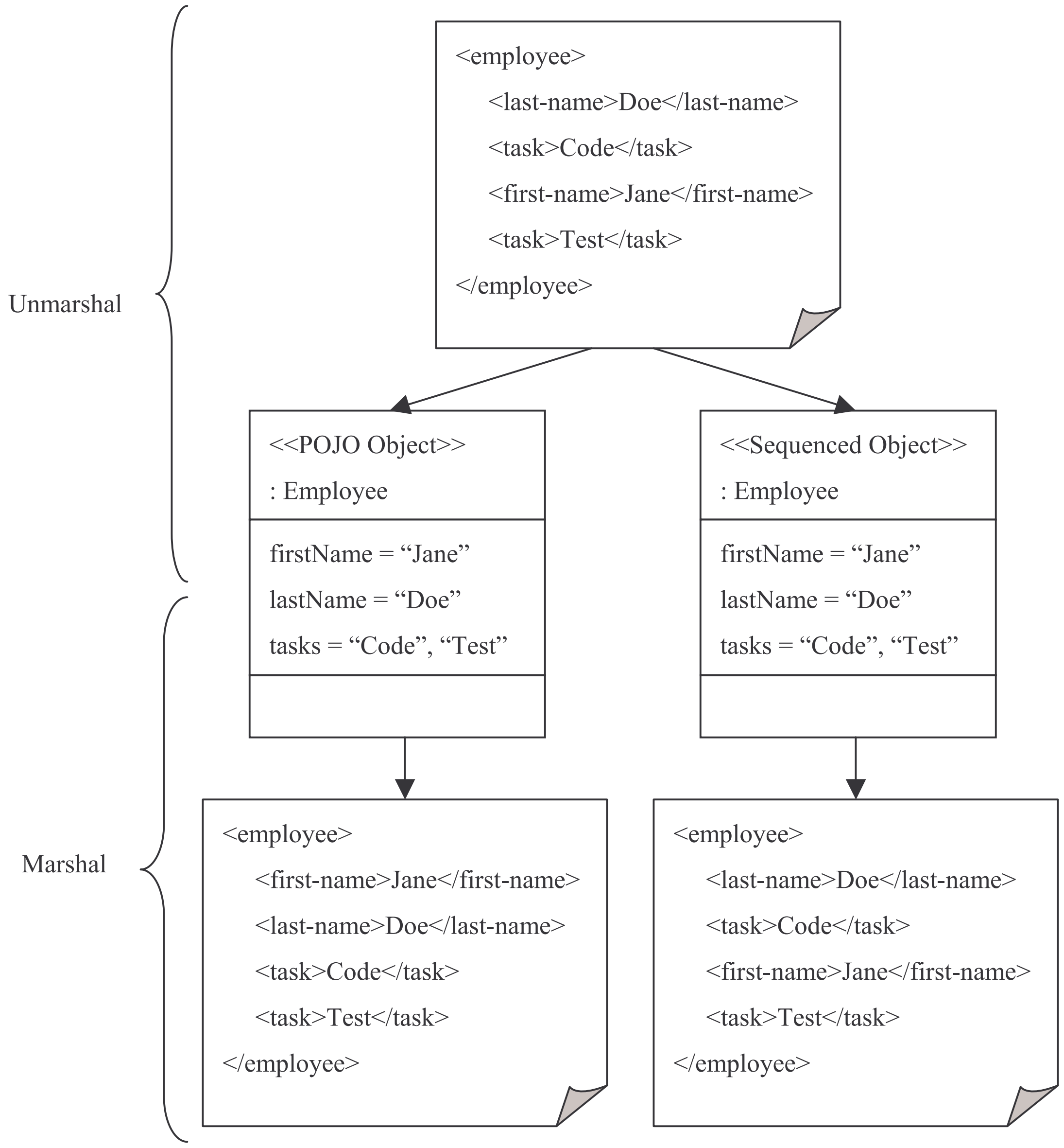
Sequenced objects are similar to POJO objects except that they maintain order across the properties when unmarshalling and marshalling.

1.3. Benefits

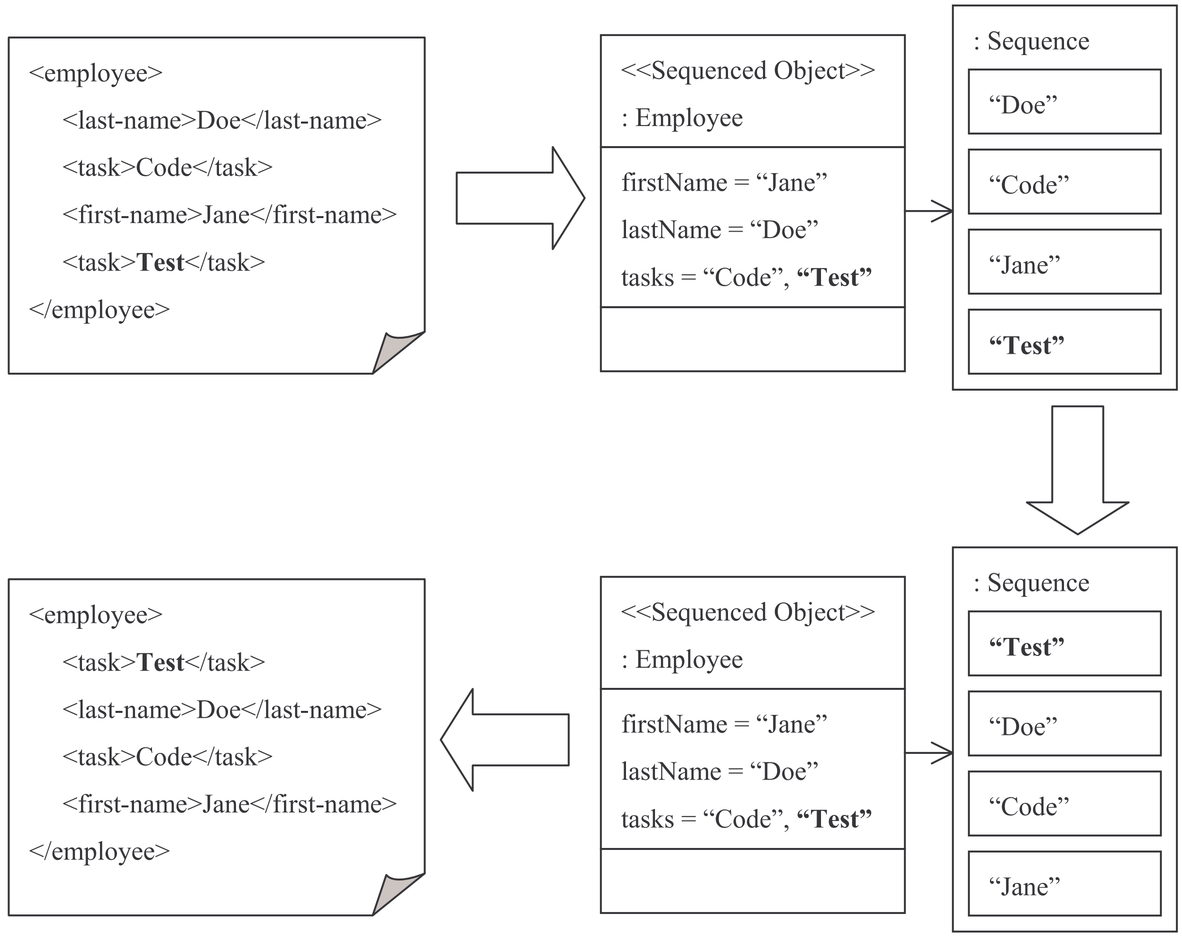
The user is given a way to affect the OXM mappings at an instance level.

2. Concepts

Sequenced objects are similar to POJO objects except that they maintain order across the properties when unmarshalling and marshalling. For regular POJOs the order of properties when marshalling is based on the order of mappings.



The user needs the ability to change the order of the sequence so this will need to be made available to the user.



3. Requirements

3.1. Scope

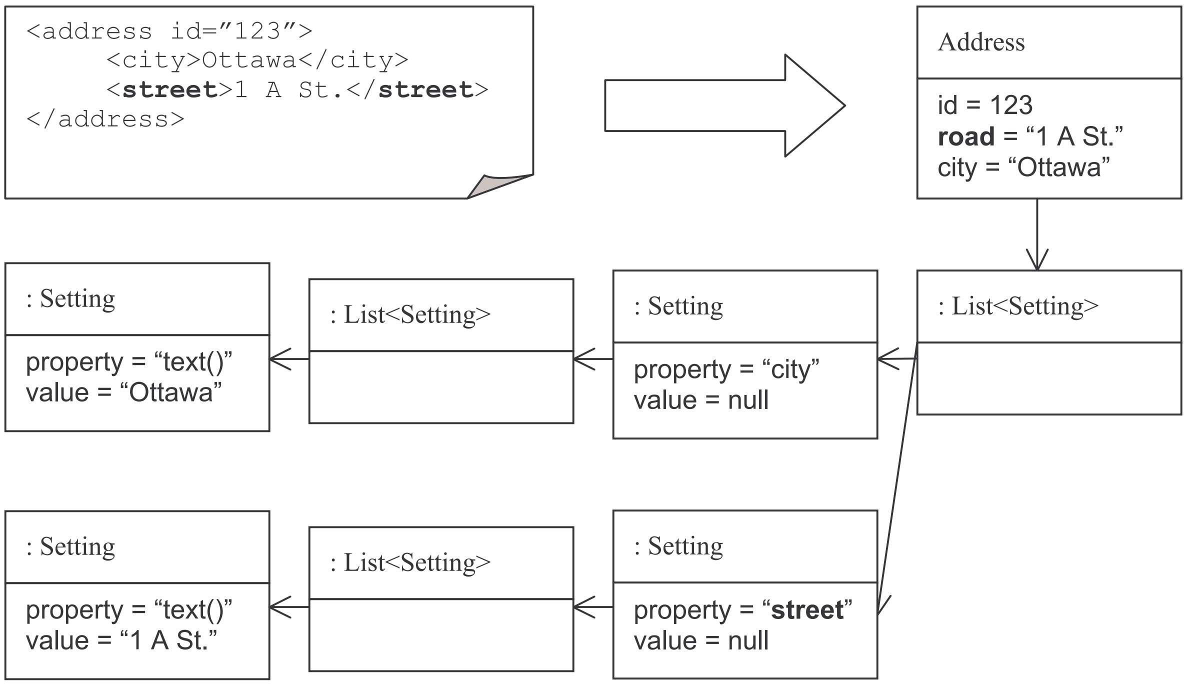
The graph can contain a mix of sequenced and non-sequenced objects. Therefore sequencing is scoped at the object level.

3.2. Functionality

3.2.1. Direct Mappings

Note:

- Sequences do not apply to object properties that are mapped to XML attributes.
- The name property on the Setting objects corresponds to the name of the XML node.

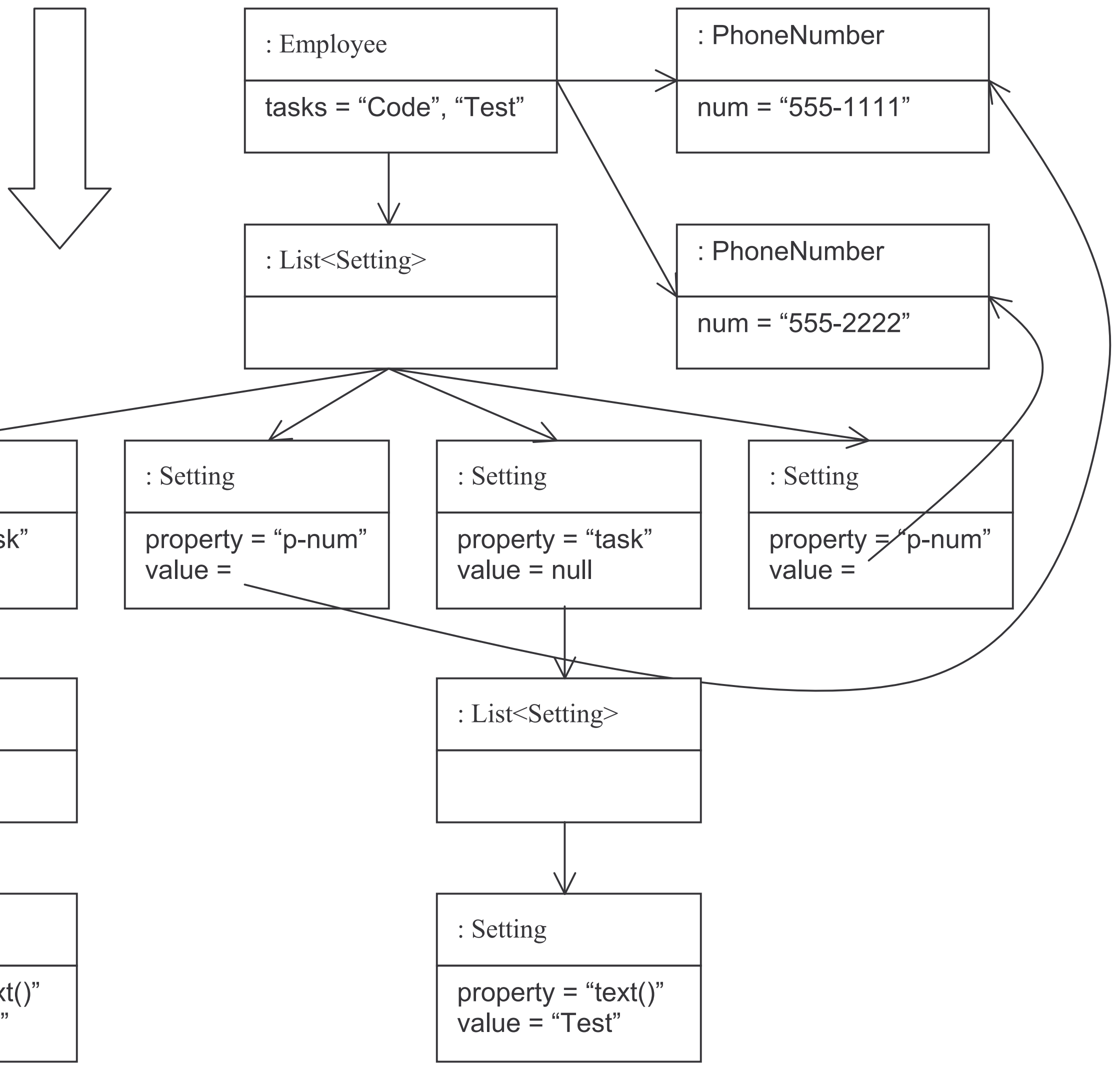


```
<address id="123">
     <city>Ottawa</city>
     <street>1 A St.</street>
</address>
```

3.2.2. Collection Mappings

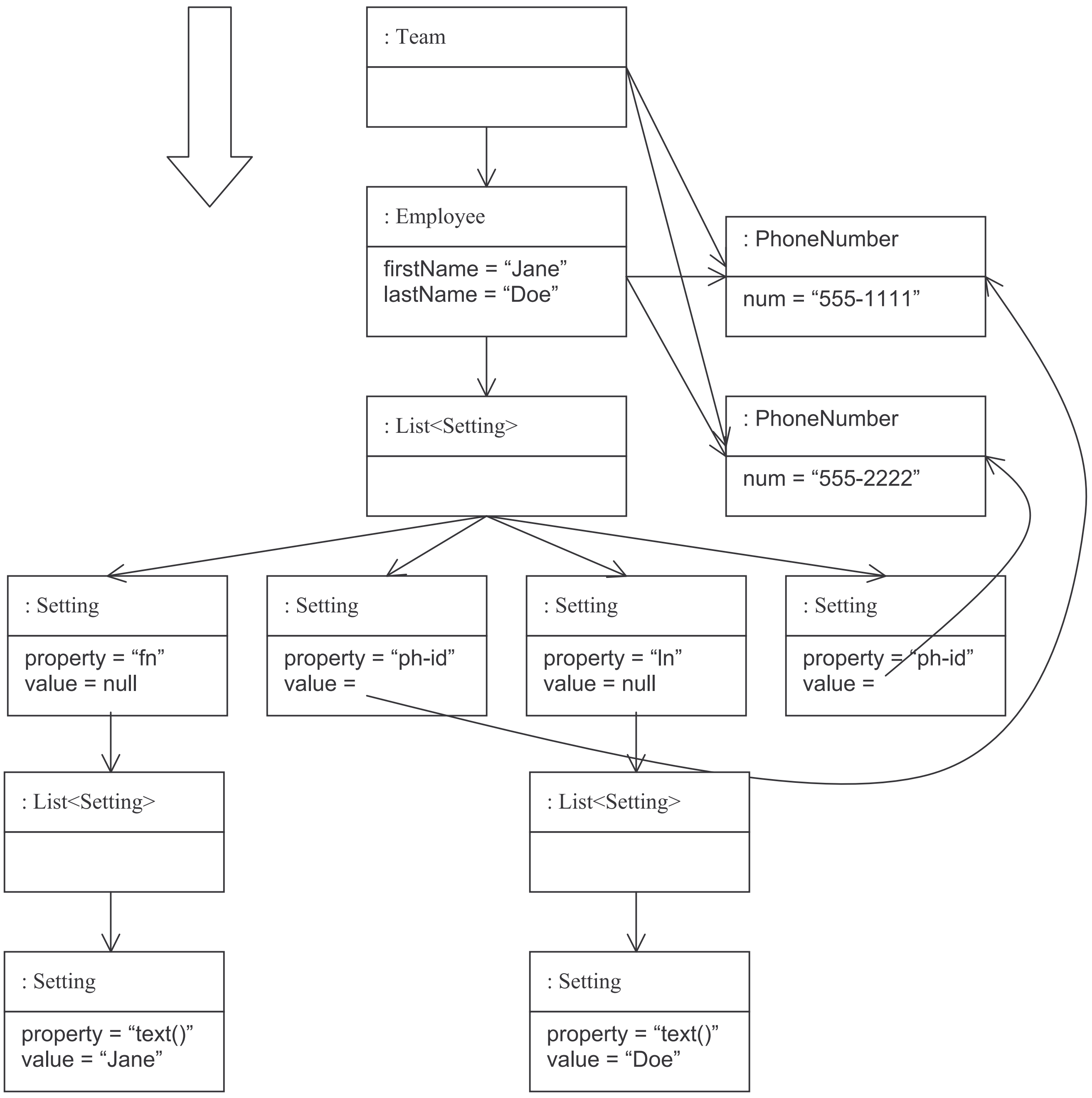
Each value from the collection results in an individual setting in the sequence.

```
<employee>
     <task>Code</task>
     <p-num type=”home”>
          <number>555-1111</number>
     </p-num>
     <task>Test</task>
     <p-num type=”work”>
          <number>555-2222</number>
     </p-num>
</employee>
```



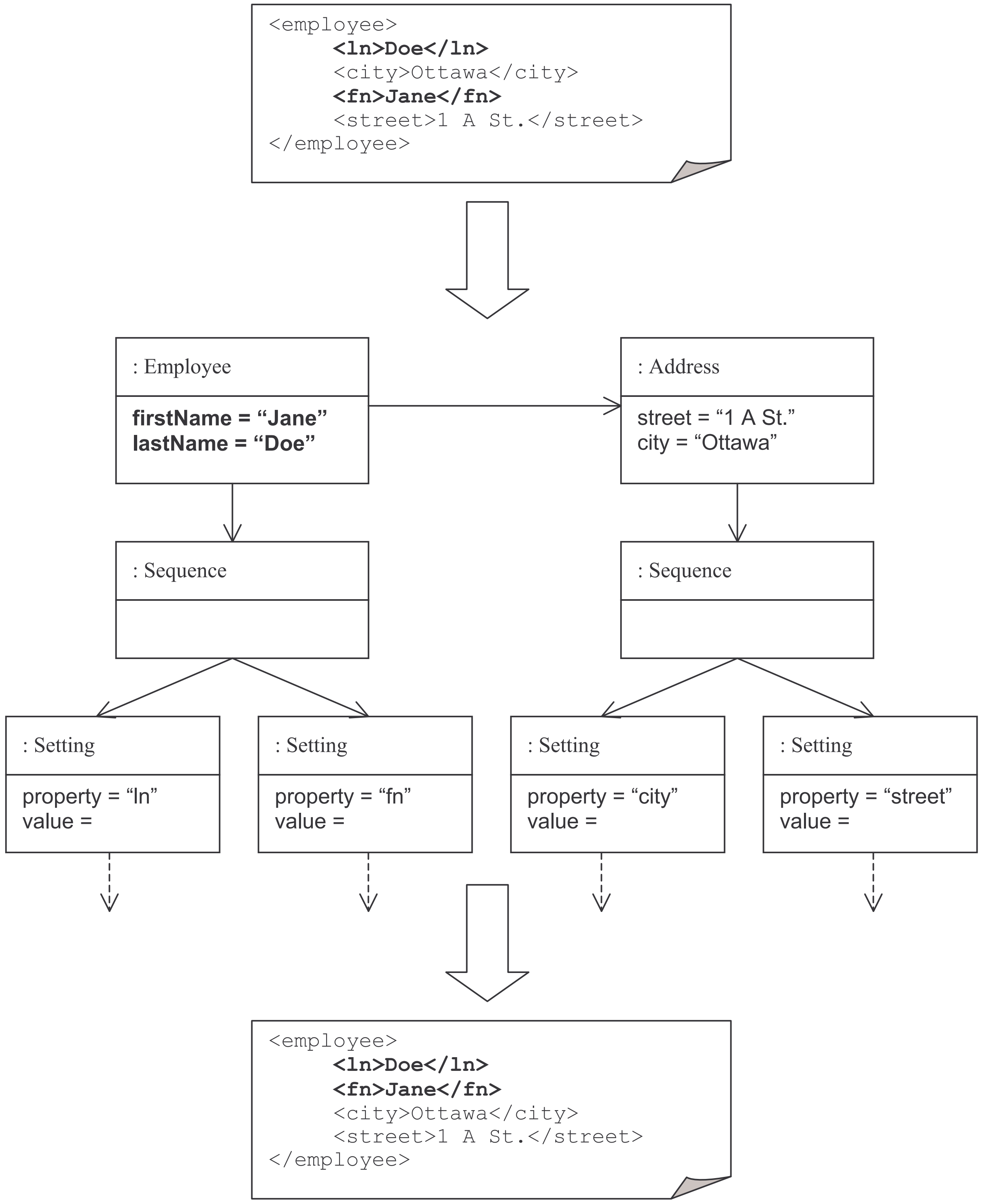
3.2.3. Reference Mappings

```
<team>
     <employee>
          <fn>Jane</fn>
          <ph-id>2</ph-id>
          <ln>Doe</ln>
          <ph-id>1</ph-id>
     </employee>
     <phone-number id=”1”>555-1111</phone-number>
     <phone-number id=”2”>555-2222</phone-number>
</team>
```



3.2.4. Self Mappings

If multiple objects map to the same record each object will have its own sequence.



3.2.5. Any Mapping

Sequences would support this type of mapping.

3.2.6. Any Attribute Mapping

Sequenced objects do not track XML attributes, so this mapping would not apply.

3.2.7. XML Fragment Mappings

These values would apply to sequences unless the node was mapped to an XML attribute.

3.2.8. Transformation Mappings

Transformation mappings are out of scope for this feature. Since the value can span multiple elements it would be difficult to represent it in sequence.

3.2.9. XML Binary Mappings

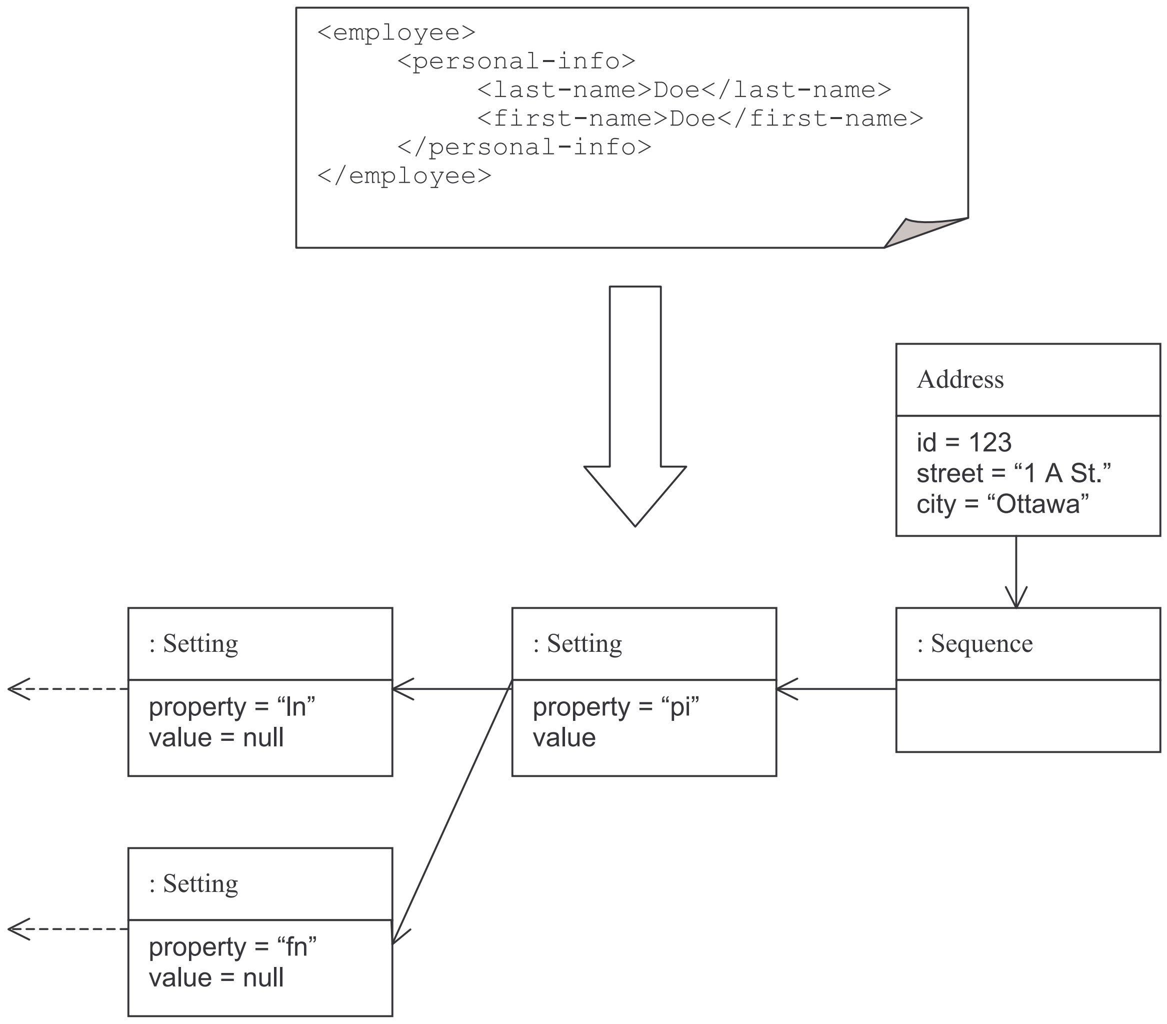
These values would apply to sequences unless the node was mapped to an XML attribute.

3.2.10. Inheritance

- Inherited mappings (except those to XML attributes) are included in the sequence.
- If the parent object is a sequenced object, this does not automatically make the child object a sequenced object.
- If the parent object is not a sequenced object, this does not prevent the child object from being a sequenced object.

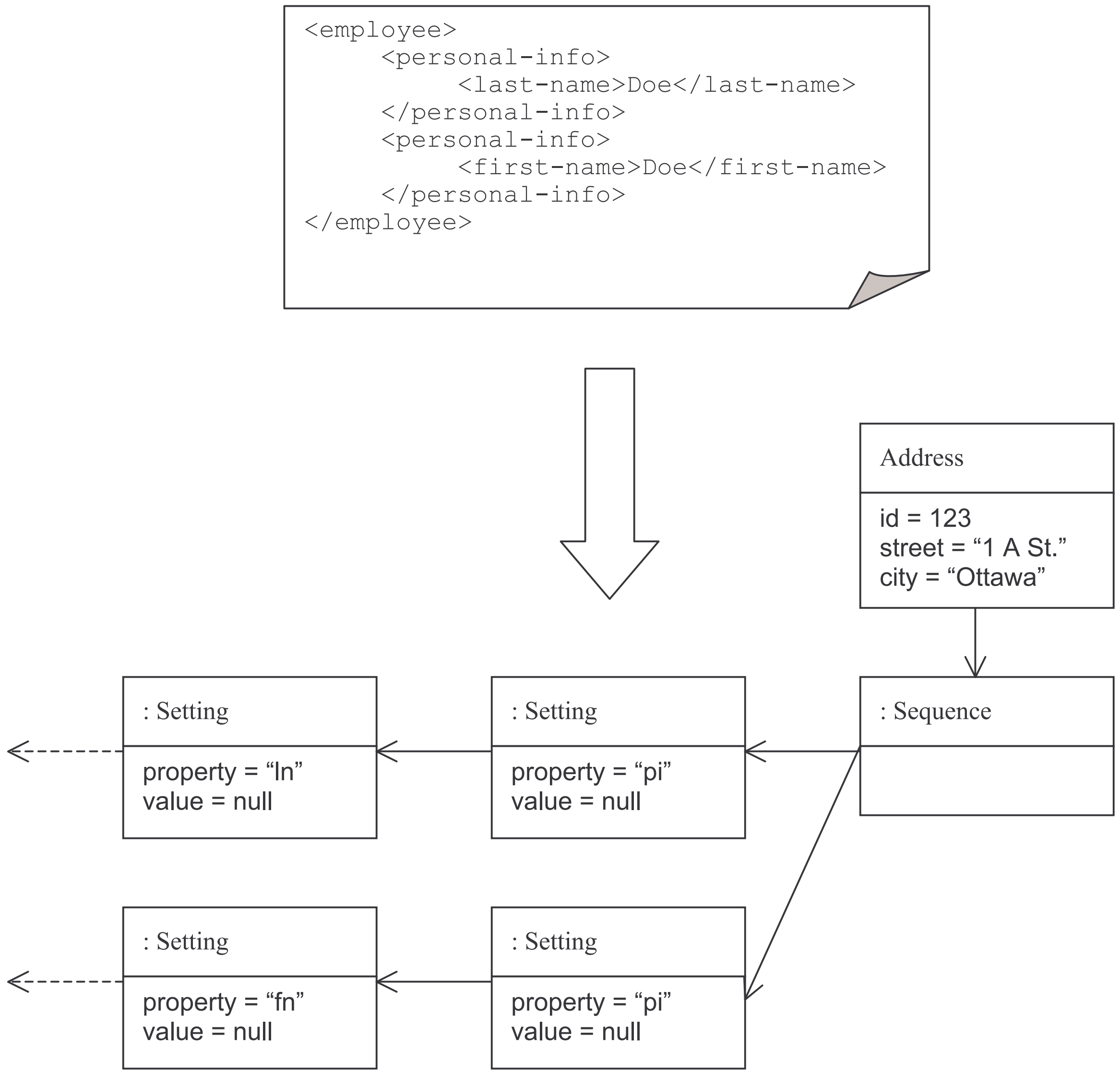
3.2.11. Path Based Mappings

3.2.11.1.Example # 1



```
<employee>
     <personal-info>
          <last-name>Doe</last-name>
          <first-name>Doe</first-name>
     </personal-info>
</employee>
```

3.2.11.2.Example # 2



```
<employee>
     <personal-info>
          <last-name>Doe</last-name>
     </personal-info>
     <personal-info>
          <first-name>Doe</first-name>
     </personal-info>
</employee>
```

3.3. Performance

3.3.1. Performance Attributes

3.3.1.1. Memory Usage

Sequenced objects are expected to use more memory than unsequenced objects, this is because they both require the same base POJO, but in addition the sequenced object must maintain an additional data structure to preserve the order of properties.

3.3.1.2. Marshal Performance

Once implemented, the time taken to marshal sequenced objects must not be noticeably different from the amount of time it takes to marshal unsequenced (regular POJO) objects.

3.3.1.3. Unmarshal Performance

It will take longer to unmarshal sequenced objects than unsequenced (regular POJO) objects. The extra time will be spent moving data between the object and the sequence.

3.3.2. Performance Tuning

Performance tuning is not an issue.

3.3.3. Who Should Tune

Customers are not expected to tune the performance of this project.

3.3.4. Performance Measurement

3.3.4.1. Memory Usage

Memory usage can be measured using a tool such as JProbe. The test should compare memory usage against an equivalent POJO model.

3.3.4.2. Marshal Performance

Marshal performance can be measured by adding additional test cases to the EclipseLink OXM performance test suite.

3.3.4.3. Unmarshal Performance

Unmarshal performance can be measured by adding additional test cases to the EclipseLink OXM performance test suite.

3.3.5. Administrative Interfaces

There is no requirement to implement administration function for performance tuning.

3.4. Availability

No new availability requirements are needed and none need to be changed for this project.

3.5. Scalability

No new scalability requirements are needed and none need to be changed for this project.

3.6. Design Constraints

The design must support the usage of this feature to implement Sequenced Data Object feature in SDO.

3.7. Manageability

Manageability is not an issue for this project.

3.8. Reliability

No new reliability requirements are needed and none needs to be changed for this project.

3.9. Portability and Platform-Specific Requirements

There are no portability requirements for this project.

3.10. Maintainability

There are not maintainability requirements for this project.

3.11. Security

There are no security requirements for this project.

3.12. Compatibility

3.12.1. Internationalization

New exceptions and the UI components will need to be internationalized.

3.13. Diagnosability

3.13.1. Integrity Checker

The Integrity Checker could verify that if the descriptor indicates that the object is sequenced that object implements the SequencedObject interface.

4. Client Interfaces

4.1. Command-line Tools

There are no command-line tools for this project.

4.2. GUI Design Elements

A mechanism needs to be made available to flag that the descriptor represents a sequenced object.

4.3. Public APIs/Classes

4.3.1. XMLDescriptor (MODIFIED)

```
/**
 * Return true if the object will be marshalled/unmarshalled as a
 * sequenced object, else false.  Note sequenced objects must implement
 * the SequencedObject interface.
 * @see org.eclipse.persistence.oxm.sequenced.SequencedObject
 */
public boolean isSequencedObject() {
      …
}

/**
 * Specify if the object will be marshalled/unmarshalled as a
 * sequenced object Note sequenced objects must implement the
 * SequencedObject interface.
 * @see org.eclipse.persistence.oxm.sequenced.SequencedObject
 */
public void setSequencedObject(boolean isSequencedObject) {
      …
}
```

4.3.2. SequencedObject (NEW)

```
package org.eclipse.persistence.oxm.sequenced;

import java.util.List;

/**
 * <p>For sequenced objects the property values correspond to the
 * setting and not the domain object.</p>
 */
public interface SequencedObject {

      List<Setting> getSettings();

}
```

4.3.3. Setting (NEW)

```
package org.eclipse.persistence.oxm.sequenced;

/**
 * <b>Example 1</b>
 * <pre>
 * Setting piSetting = new Setting();
 * piSetting.setName("personal-info");
 *
 * Setting fnSetting = new Setting();
 * fnSetting.setName("first-name");
 * piSetting.getSequence().add(fnSetting);
 *
 * Setting fnTextSetting = new Setting();
 * fnTextSetting.setName("text()");
 * fnTextSetting.setValue("Jane");
 * fnSetting.getSequence().add(fnTextSetting);
 *
 * Setting lastNameSetting = new Setting();
 * lnSetting.setName("last-name");
 * piSetting.getSequence().add(lnSetting);
 *
 * Setting lnTextSetting = new Setting();
 * lnTextSetting.setName("text()");
 * lnTextSetting.setValue("Doe");
 * lnSetting.getSequence().add(lnTextSetting);
 * </pre>
 * <pre>
 * &lt;personal-info&gt;
 *      &lt;first-name&gt;Jane&lt;/first-name&gt;
 *      &lt;last-name&gt;Doe&lt;/last-name&gt;
 * &lt;/personal-info&gt;
 * </pre>
 * <b>Example 2</b>
 * <pre>
 * Setting fnpiSetting = new Setting();
 * fnpiSetting.setName("personal-info");
 *
 * Setting fnSetting = new Setting();
 * fnSetting.setName("first-name");
 * fnpiSetting.getSequence().add(fnSetting);
 *
 * Setting fnTextSetting = new Setting();
 * fnTextSetting.setName("text()");
 * fnTextSetting.setValue("Jane");
 * fnSetting.getSequence().add(fnTextSetting);
 *
 * Setting lnpiSetting = new Setting();
 * lnpiSetting.setName("personal-info");

 * Setting lastNameSetting = new Setting();
 * lnSetting.setName("last-name");
 * lnpiSetting.getSequence().add(lnSetting);
 *
 * Setting lnTextSetting = new Setting();
 * lnTextSetting.setName("text()");
 * lnTextSetting.setValue("Doe");
 * lnSetting.getSequence().add(lnTextSetting);
 * </pre>
 * <pre>
 * &lt;personal-info&gt;
 *      &lt;first-name&gt;Jane&lt;/first-name&gt;
 * &lt;/personal-info&gt;
 * &lt;personal-info&gt;
 *      &lt;last-name&gt;Doe&lt;/last-name&gt;
 * &lt;/personal-info&gt;
 * </pre>
 */
public class Setting {

      /**
       * <p>Return the name of the setting.  The name of the setting
       * corresponds to a fragment of an XPath in an object-to-XML
       * mapping.</p>
       * <b>Example</b>
       * <p>For the XPath personal-info/first-name/text() would
       * correspond to 3 Setting objects with names "personal-info",
       * "first-name", and "text()" </p>
       */
      public String getName() {
            …
      }

      /**
       * <p>Specify the name of the setting.  The name of the setting
       * corresponds to a fragment of an XPath in an object-to-XML
       * mapping.</p>
       * <b>Example</b>
       * <p>For the XPath personal-info/first-name/text() would
       * correspond to 3 Setting objects with names "personal-info",
       * "first-name", and "text()"</p>
       */
      public void setName(String name) {
            …
      }

      /**
       * Return the namespace URI that qualifies the name of the Setting
       * (if there is one).
       */
      public String getNamespaceURI() {
            return namespaceURI;
```

```
        }

        /**
         * Specify the namespace URI that qualifies the name of the
         * Setting (if there is one).
         */
        public void setNamespaceURI(String namespaceURI) {
              this.namespaceURI = namespaceURI;
        }

        /**
         * Return the value corresponding to this setting.  For sequenced
         * object the property values correspond to the setting and not
         * the domain object.
         */
        public Object getValue() {
              return value;
        }

        /**
         * Specify the value corresponding to this setting.  For sequenced
         * object the property values correspond to the setting and not
         * the domain object.
         */
        public void setValue(Object value) {
              this.value = value;
        }

        /**
         * Get the ObjectUpdatePolicy for this Setting.
         */
        public void getObjectUpdatePolicy() {
              return this.objectUpdatePolicy;
        }

        /**
         * Specify an ObjectUpdatePolicy for this Setting.
         */
        public void setObjectUpdatePolicy(ObjectUpdatePolicy
              objectUpdatePolicy) {
              this.objectUpdatePolicy = objectUpdatePolicy;
        }

        /**
         * Return the child Setting objects.
         */
        public List<Setting> getSettings() {
              …
        }

  }
```

4.3.4. ObjectUpdatePolicy

```
package org.eclipse.persistence.oxm.sequenced;

/**
 * The ObjectUpdatePolicy provides the opportunity to specify logic
 * responsible for setting the corresponding value on the POJO.
 */
public interface ObjectUpdatePolicy {

      void updateObject(Object value);

}
```

4.4. Configuration Files

4.4.1. Deployment XML

The following element will be added to the OXM descriptor complex type in deployment XML.

```
<xsd:complexType name="xml-class-mapping-descriptor">
      <xsd:complexContent>
            <xsd:extension base="eclipselink:class-mapping-descriptor">
                  <xsd:sequence>
                        …
                        <xsd:element name=”sequenced-object”
                              type=”xsd:boolean”
                              default=”false”
                              minOccurs=”0”/>
                  </xsd:sequence>
            </xsd:extension>
      <xsd:complexContent>
<xsd:complexType>
```

4.4.2. Sessions XML

No changes are required to Sessions XML.

4.4.3. EclipseLink Workbench

The EclipseLink Workbench metadata will need to be updated to handle this setting.

5. Dependencies and Effects

5.1. Software

Table 6–1 Software Dependencies and Effects

Within EclipseLink Dependency		
	Effects	**Support Level**
Multiple tables		NOT APPLICABLE
Aggregates		NOT APPLICABLE
Inheritance	Mappings inherited from parent objects need to factor into the sequence.	FULL SUPPORT (OXM)
Interfaces		NOT APPLICABLE
Direct mappings	Mappings will need to populate the sequence.	FULL SUPPORT (OXM)
Relationship mappings	Mappings will need to populate the sequence.	FULL SUPPORT (OXM)
Aggregate mappings	Mappings will need to populate the sequence.	FULL SUPPORT (OXM)
Variable mappings		NOT APPLICABLE
OXM	This is an OXM feature and will require changes to OXM.	FULL SUPPORT
Object Relational		NOT APPLICABLE
EIS	This feature could apply to EIS, but is out of scope for this project.	NO SUPPORT
Indirection		NOT APPLICABLE
Returning		NOT APPLICABLE
Stored procedures		NOT APPLICABLE
Sequencing		NOT APPLICABLE
Optimistic locking		NOT APPLICABLE
Expressions		NOT APPLICABLE
Sub-queries		NOT APPLICABLE
Queries		NOT APPLICABLE
Report queries		NOT APPLICABLE
EJB-QL		NOT APPLICABLE

Cursors		NOT APPLICABLE
Joining, partial objects		NOT APPLICABLE
Batch reading		NOT APPLICABLE
Caching		NOT APPLICABLE
Cache synchronization, invalidation		NOT APPLICABLE
Unit of work		NOT APPLICABLE
Server sessions		NOT APPLICABLE
Session broker		NOT APPLICABLE
Remote sessions		NOT APPLICABLE
Historical sessions		NOT APPLICABLE
Connection pooling		NOT APPLICABLE
JTS integration		NOT APPLICABLE
EJB - CMP		NOT APPLICABLE
JDO		NOT APPLICABLE
Events		NOT APPLICABLE
Logging		NOT APPLICABLE
SDO	Sequenced Data Objects will be implemented using this feature.	FULL SUPPORT

5.2. User Documentation

User documentation impacts used to be recorded in the functional specification. Rather than track this information in the functional specification, it is now kept in ST Project. Please log in to ST project and create cross-references between your project and user documents for the appropriate release.

5.3. Internal Documentation

There is not impact on other internal documents.

6. Compatibility & Migration

6.1. Deployment XML changes

Metadata is required at the descriptor level to indicate that the object is sequenced. Past versions of OXM metadata must be read in as representing unsequenced objects.

6.2. Sessions XML Changes

There are no changes required to Sessions XML.

6.3. Internal MW Project Format Changes

Metadata will be required by the internal MW project to determine if a descriptor represents a sequenced object. This behaviour must match the deployment XML changes.

A. References

Table 6–1 References

[SDO]	Service Data Object Specification, currently at release 2.1. http://www.osoa.org/display/Main/Service+Data+Objects+Specifications

C. Open Issues

Issue Log			
Issue #	**Name**	**Description**	**Status**
1			OPEN/CLOSED
2			
3			

D. Decisions

Decision Log		
Decision #	**Name**	**Description**
1		
2		
3		