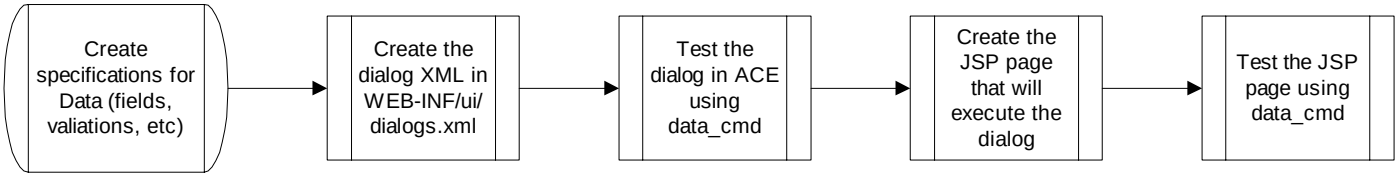
Netspective

Pattern: **[P01] Simple Data Dialog with DML embedded in XML (no Java required)**

Pros: Only requires basic XML and JSP knowlege and can be done very quickly
Cons: Embeds business logic, database access, and user interface in one object, little control
Uses: XML, com.xaf.form.Dialog, com.xaf.task.sql.DmlTask, com.xaf.navigate.taglib.DialogTag



sample Site/test-dialog.jsp

```
<jsp:directive.taglib prefix="app" uri="/WEB-INF/tld/page.tld"/>
<jsp:directive.taglib prefix="xaf" uri="/WEB-INF/tld/xaf.tld"/>

<app:page title="Test the dialog" heading="Test the dialog">

  <xaf:dialog source="Organization" name="org.registration"/>

</app:page>
```

Sections in dialogs.xml

1. Fields declaration (common to all dialogs)
2. Field population tasks (SQL will execute only for edit and delete modes not for add mode)
3. Dialog execution for add mode
4. Dialog execution for edit mode
5. Dialog execution for delete mode

sample Site/WEB-INF/ui/dialogs.xml

```
<xaf>
  <dialogs package="org">
    <dialog name="registration" heading="create-data-cmd-heading:Account" retain-params="org_id">
      <field.debug visible="no"/>
      <field.text name="org_code" caption="Account Code" required="yes"/>
      <field.text name="org_name" caption="Name" required="yes"/>
      <field.text name="org_abbrev" caption="Abbreviation"/>
      <field.select name="org_type" caption="Type" choices="schema-enum:Org_Type_Enum">
        <conditional action="apply-flag" flag="invisible" data-cmd="edit,delete"/>
      </field.select>
      <field.select name="org_industry" caption="Industry" choices="schema-enum:Org_Industry_Enum">
        <conditional action="apply-flag" flag="invisible" data-cmd="edit,delete"/>
      </field.select>
      <field.select name="ownership" caption="Ownership" choices="schema-enum:Org_Ownership"/>
      <field.text name="ticker_symbol" caption="Ticker Symbol"/>
      <field.integer name="employees" caption="Employees"/>
      <field.select name="time_zone" caption="Time Zone" choices="Central;Eastern;Mountain;Pacific"/>
```

1

```
      <populate-tasks data-cmd="edit,delete">
        <exec-statement report="none" store-type="row-fields" store="form:*">
          select * from org where org_id = ?
          <params>
            <param value="request:org_id"/>
          </params>
        </exec-statement>
      </populate-tasks>
```

2

```
      <execute-tasks data-cmd="add">
        <exec-transaction command="begin"/>
        <exec-dml command="insert" table="org" auto-inc="org_id,org_org_id_seq"
          auto-inc-store="request-attr:org_id"
          fields="org_code,org_name,org_abbrev,ownership,ticker_symbol,employees,time_zone"/>
        <exec-dml command="insert" table="org_industry" columns="org_id=request-attr:org_id,system_id=custom-sql:oind_system_id_seq.nextval"
          fields="org_industry"/>
        <exec-dml command="insert" table="org_type" columns="org_id=request-attr:org_id,system_id=custom-sql:otyp_system_id_seq.nextval"
          fields="org_type"/>
        <exec-transaction command="end"/>
        <exec-redirect url="config-expr:${create-app-url:/account/home.jsp}?org_id=${request-attr:org_id}"/>
      </execute-tasks>
```

3

```
      <execute-tasks data-cmd="edit">
        <exec-dml command="update" table="org" fields="org_code,org_name,org_abbrev,ownership,ticker_symbol,employees,time_zone" where="org_id = ?"
          where-bind="request:org_id"/>
        <exec-redirect url="config-expr:${create-app-url:/account/home.jsp}?org_id=${request:org_id}"/>
      </execute-tasks>
```

4

```
      <execute-tasks data-cmd="delete">
        <exec-transaction command="begin"/>
        <exec-dml command="delete" table="org_industry" where="org_id = ?" where-bind="request:org_id"/>
        <exec-dml command="delete" table="org_type" where="org_id = ?" where-bind="request:org_id"/>
        <exec-dml command="delete" table="org" where="org_id = ?" where-bind="request:org_id"/>
        <exec-transaction command="end"/>
        <exec-redirect url="create-app-url:/account"/>
      </execute-tasks>
    </dialog>
  </dialogs>
```

5

Netspective

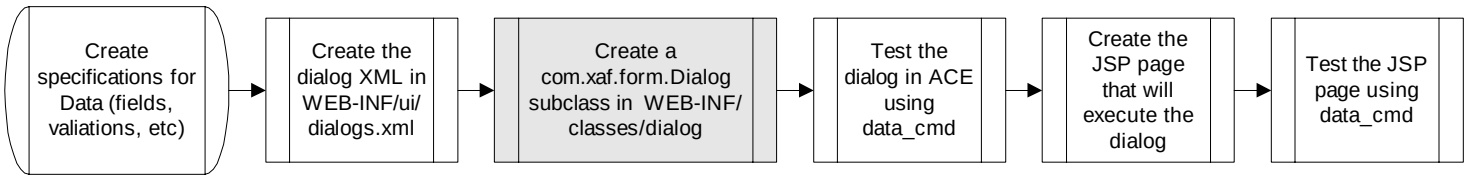
Pattern: **[P02] Data Dialog with DML in XML and Java-based validation**

Extends pattern: **[P01]**

Pros: Only requires basic XML and JSP knowlege and can be done very quickly with little Java

Cons: Embeds business logic, database access, and user interface in one object, little control

Uses: XML, com.xaf.form.Dialog, com.xaf.task.sql.DmlTask, com.xaf.navigate.taglib.DialogTag



sample Site/WEB-INF/ui/dialogs.xml is same as in Pattern **[P01]** except for first line in <dialog> tag

```
<xaf>
  <dialogs package="org">
    <dialog name="registration" class="dialog.org.Registration"
      heading="create-data-cmd-heading:Account" retain-params="org_id">
```

sample Site/WEB-INF/classes/dialog/org/Registration.java

```
package dialog.org;

public class Registration extends com.xaf.form.Dialog
{
  public boolean isValid(DialogContext dc) // override the com.xaf.form.Dialog.isValid
  {
     // the dc is a com.xaf.form.DialogContext object which contains the values of all the fields
     // and other runtime information

     // first make sure that default validation takes place
     if(! super.isValid(dc))
       return false;

     if(dc.addingData() && dc.hasValue("my_field_name"))
     {
        // add the error mesage to a specific field
        dc.addErrorMessage("my_field_name", "Something's wrong");
        return false;
     }

     if(dc.editingData() && dc.getValue("another_field").equals(dc.getValue("yet_another_field")))
     {
        // add the error mesage to the dialog (at the top)
        dc.addErrorMessage("Something's wrong");
        return false;
     }
  }
}
```

sample Site/test-dialog.jsp is same as in Pattern [P01]

```
  <jsp:directive.taglib prefix="app" uri="/WEB-INF/tld/page.tld"/>
  <jsp:directive.taglib prefix="xaf" uri="/WEB-INF/tld/xaf.tld"/>

  <app:page title="Test the dialog" heading="Test the dialog">
    <xaf:dialog source="Organization" name="org.registration"/>
  </app:page>
```

Netspective

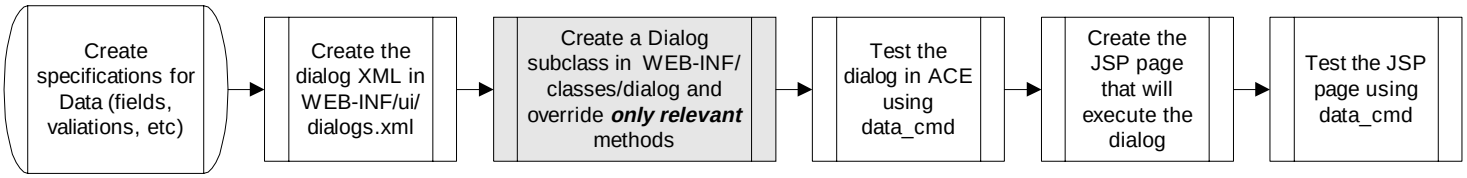
Pattern: **[P03] Data Dialog with DML and validation in Java**

Extends pattern: **[P01], [P02]**

Pros: Full control over transaction protection and other issues

Cons: Requires more java code

Uses: XML, com.xaf.form.Dialog, com.xaf.task.sql.DmlTask, com.xaf.navigate.taglib.DialogTag



sample Site/WEB-INF/classes/dialog/org/Registration.java

```
package dialog.org;

public class Registration extends com.xaf.form.Dialog
{
  public void populateValues(DialogContext dc, int formatType)
  {
      // if you call super.populateValues(dc, formatType) then you will process XML
      // <populate-tasks> as well; if you leave out super.populateValues(), then you're overrriding
      super.populateValues(dc, formatType);

      // you should almost always call dc.isInitialEntry() to ensure that you're not populating
      //  data unless the user is seeing the data for the first time
      if(! dc.isInitialEntry())
          return;

      // now do the populating using DialogContext methods
      if(dc.addingData()
        dc.populateValuesFromStatement("sql-pkg.sql-stmt");
      if(dc.editingData())
        dc.populateValuesFromStatement("sql-pkg.sql-stmt", new Object[] { param1, param2 });
  }

  /**
   * use this method to override/change field flags or values based on certain conditions
   */
  public void makeStateChanges(DialogContext dc, int stage)
  {
      super.makeStateChanges(dc, stage);

      // check some stuff in the environment and hide/show a field or something
      if(dc.hasValue("something"))
        dc.setFlag("my_field_name", com.xaf.form.field.DialogField.FLDFLAG_INVISIBLE);
  }

  /**
   * this is the class that you do your entire dialog validation with
   */
  public boolean isValid(DialogContext dc)
  {
      if(!super.isValid(dc))
        return false;

      // some common methods in dc are dc.getValue("field"), dc.addingData(), dc.editingData(), etc
      return true;
  }

  /**
   * This is where you would perform all your actions. Whatever you return as the function result will be shown in the HTML
   */
  public String execute(DialogContext dc)
  {
      // if you call super.execute(dc) then you would execute the <execute-tasks> in the XML; leave it out to override
      if(dc.addingData())
        dc.executeSqlInsert("table", "field1,field2", "column=request-attr:blah");
      if(dc.editingData())
        dc.executeSqlUpdate(...);
      return "Done with data management task";
  }
}
```