MAX

Adobe

Mate Flex Framework

Using Flex Frameworks to Build Data-Driven Applications

What is Mate?

<MXML>

Event Handling

Dependency Injection

Sample Projects

- InsyncMate1
  - Exact copy of original app with a model and central event handling
- InsyncMate2
  - Logic moved to ContactManager
- InsyncMate3
  - Better architecture with Presentation Model pattern
  - Modularized application
- InsyncMate4
  - Modularized application

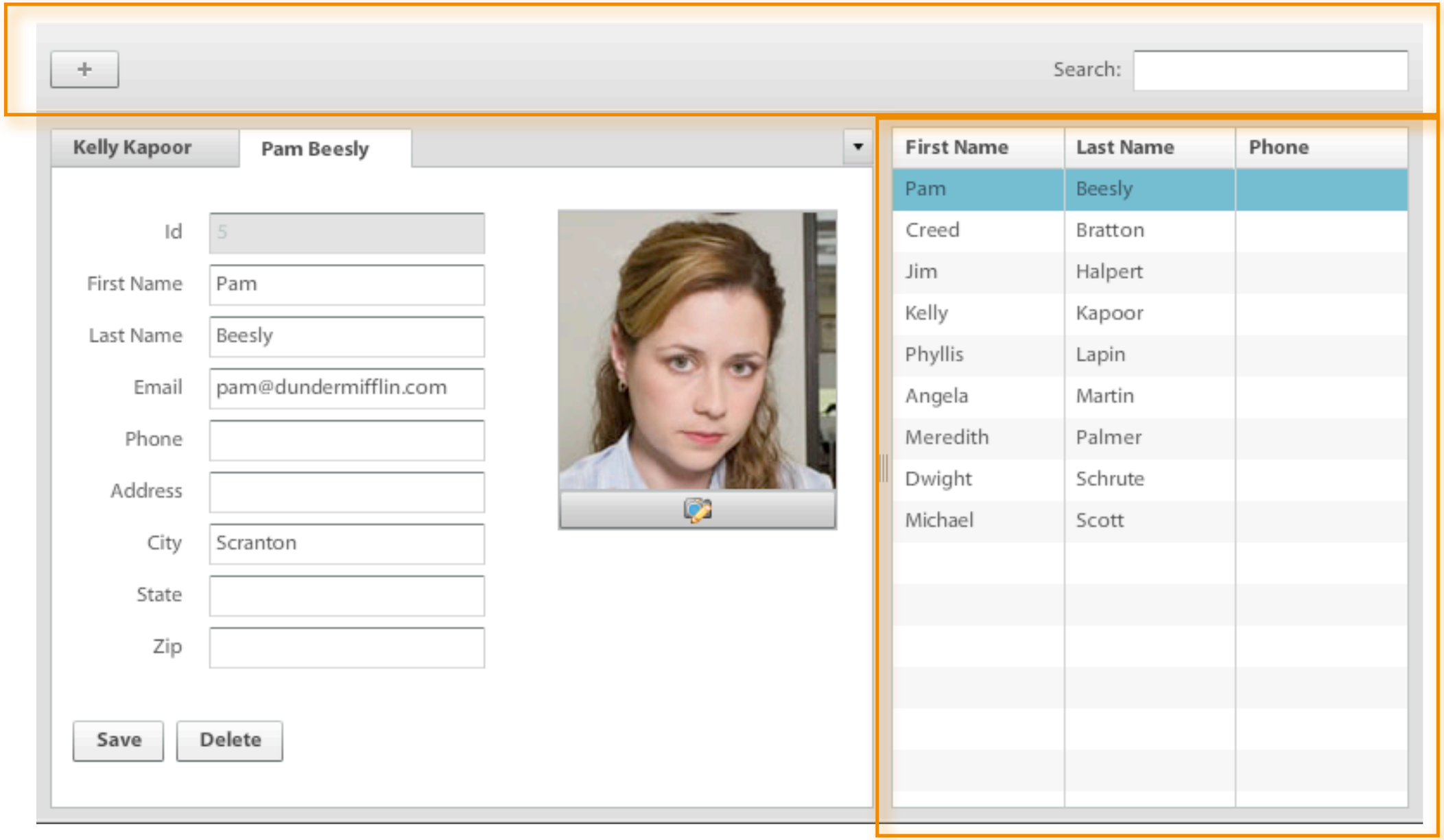
MAX

Before & After

Contact List

ContactList.mxml

Toolbar.mxml



ContactList.mxml

ContactList.mxml - Plain version

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml">

<mx:Script>
        [Bindable]
        public var contacts:ArrayCollection;
        private var lastSearchStr:String;


         public function search(searchStr:String):void {
             service.getContactsByName(searchStr);
             lastSearchStr = searchStr;
         }

         private function getContacts_result(event:ResultEvent):void {
              contacts = event.result as ArrayCollection;
         }

    </mx:Script>


    <mx:RemoteObject id="service" destination="contacts" endpoint="http://localhost:8400/
    messagebroker/amf">
            <mx:method name="getContactsByName" result="getContacts_result(event)"/>
    </mx:RemoteObject>



    <mx:DataGrid id="dg" dataProvider="{ contacts }"
            doubleClick="dispatchEvent(
                  new ContactEvent( ContactEvent.EDIT, dg.selectedItem as Contact) )"  ... />

</mx:Canvas>
```

ContactList.mxml - Plain version

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml">

<mx:Script>
        [Bindable]
        public var contacts:ArrayCollection;
        private var lastSearchStr:String;

        public function search(searchStr:String):void {
            service.getContactsByName(searchStr);
            lastSearchStr = searchStr;
        }

        private function getContacts_result(event:ResultEvent):void {
            contacts = event.result as ArrayCollection;
        }

    </mx:Script>


    <mx:RemoteObject id="service" destination="contacts" endpoint="http://localhost:8400/
    messagebroker/amf">
            <mx:method name="getContactsByName" result="getContacts_result(event)"/>
    </mx:RemoteObject>



    <mx:DataGrid id="dg" dataProvider="{ contacts }"
            doubleClick="dispatchEvent(
                    new ContactEvent( ContactEvent.EDIT, dg.selectedItem as Contact) )"  ... />

</mx:Canvas>
```

ContactList.mxml - Plain version

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml">

<mx:Script>
        [Bindable]
        public var contacts:ArrayCollection;
        private var lastSearchStr:String;

        public function search(searchStr:String):void {
            service.getContactsByName(searchStr);
            lastSearchStr = searchStr;
        }

        private function getContacts_result(event:ResultEvent):void {
            contacts = event.result as ArrayCollection;
        }

    </mx:Script>


    <mx:RemoteObject id="service" destination="contacts" endpoint="http://localhost:8400/
    messagebroker/amf">
            <mx:method name="getContactsByName" result="getContacts_result(event)"/>
    </mx:RemoteObject>



    <mx:DataGrid id="dg" dataProvider="{ contacts }"
            doubleClick="dispatchEvent(
                    new ContactEvent( ContactEvent.EDIT, dg.selectedItem as Contact) )"  ... />

</mx:Canvas>
```

called by parent
executes service call

ContactList.mxml - Plain version

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml">

<mx:Script>
        [Bindable]
        public var contacts:ArrayCollection;
        private var lastSearchStr:String;


         public function search(searchStr:String):void {
             service.getContactsByName(searchStr);
             lastSearchStr = searchStr;
         }

         private function getContacts_result(event:ResultEvent):void {
              contacts = event.result as ArrayCollection;
         }

    </mx:Script>

    <mx:RemoteObject id="service" destination="contacts" endpoint="http://localhost:8400/
    messagebroker/amf">
            <mx:method name="getContactsByName" result="getContacts_result(event)"/>
    </mx:RemoteObject>


    <mx:DataGrid id="dg" dataProvider="{ contacts }"
            doubleClick="dispatchEvent(
                  new ContactEvent( ContactEvent.EDIT, dg.selectedItem as Contact) )"  ... />

</mx:Canvas>
```

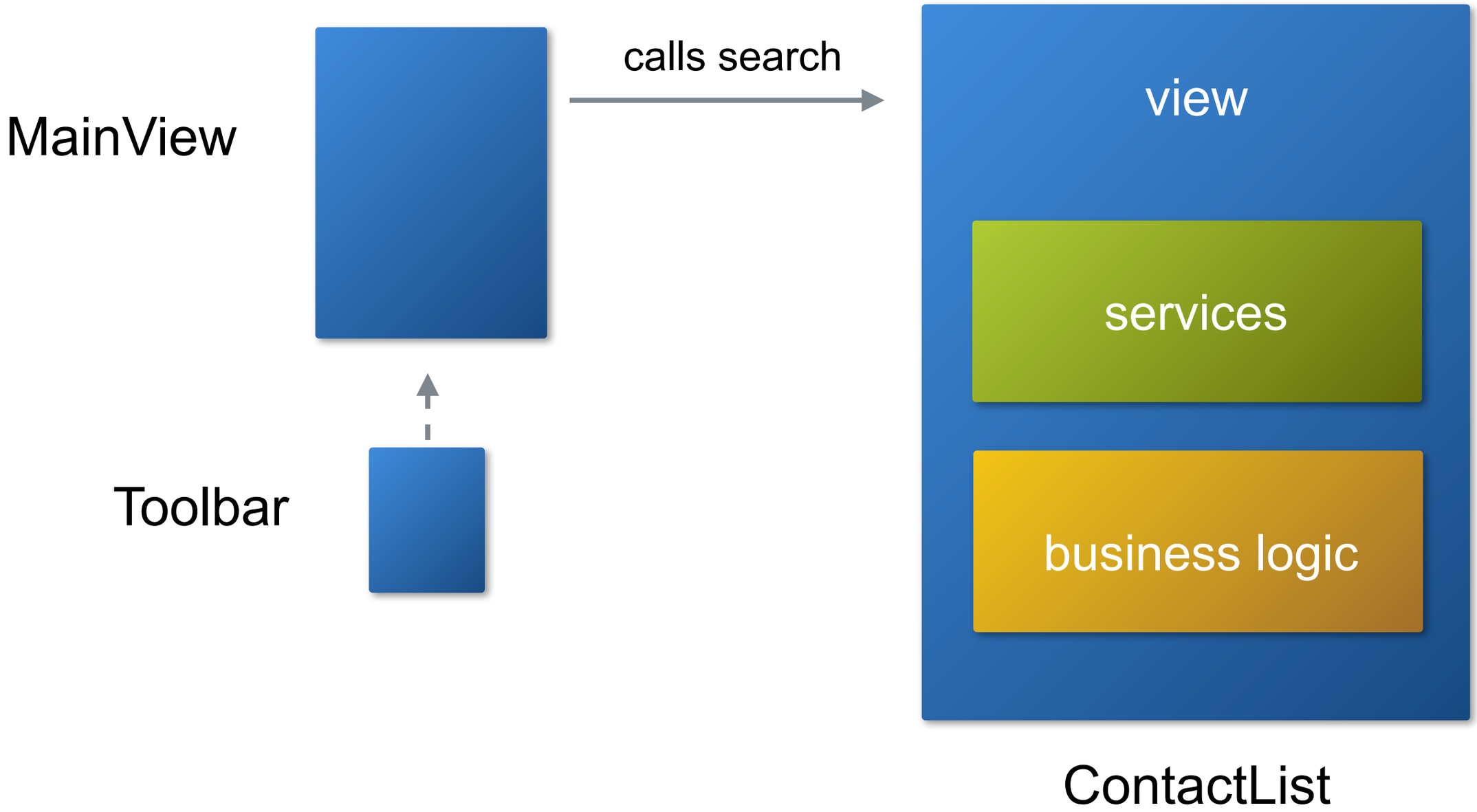
ContactList.mxml - Plain version

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml">

<mx:Script>
        [Bindable]
        public var contacts:ArrayCollection;
        private var lastSearchStr:String;


         public function search(searchStr:String):void {
             service.getContactsByName(searchStr);
             lastSearchStr = searchStr;
         }

         private function getContacts_result(event:ResultEvent):void {
              contacts = event.result as ArrayCollection;
         }
    </mx:Script>


    <mx:RemoteObject id="service" destination="contacts" endpoint="http://localhost:8400/
    messagebroker/amf">
            <mx:method name="getContactsByName" result="getContacts_result(event)"/>
    </mx:RemoteObject>



    <mx:DataGrid id="dg" dataProvider="{ contacts }"
            doubleClick="dispatchEvent(
                  new ContactEvent( ContactEvent.EDIT, dg.selectedItem as Contact) )"  ... />

</mx:Canvas>
```

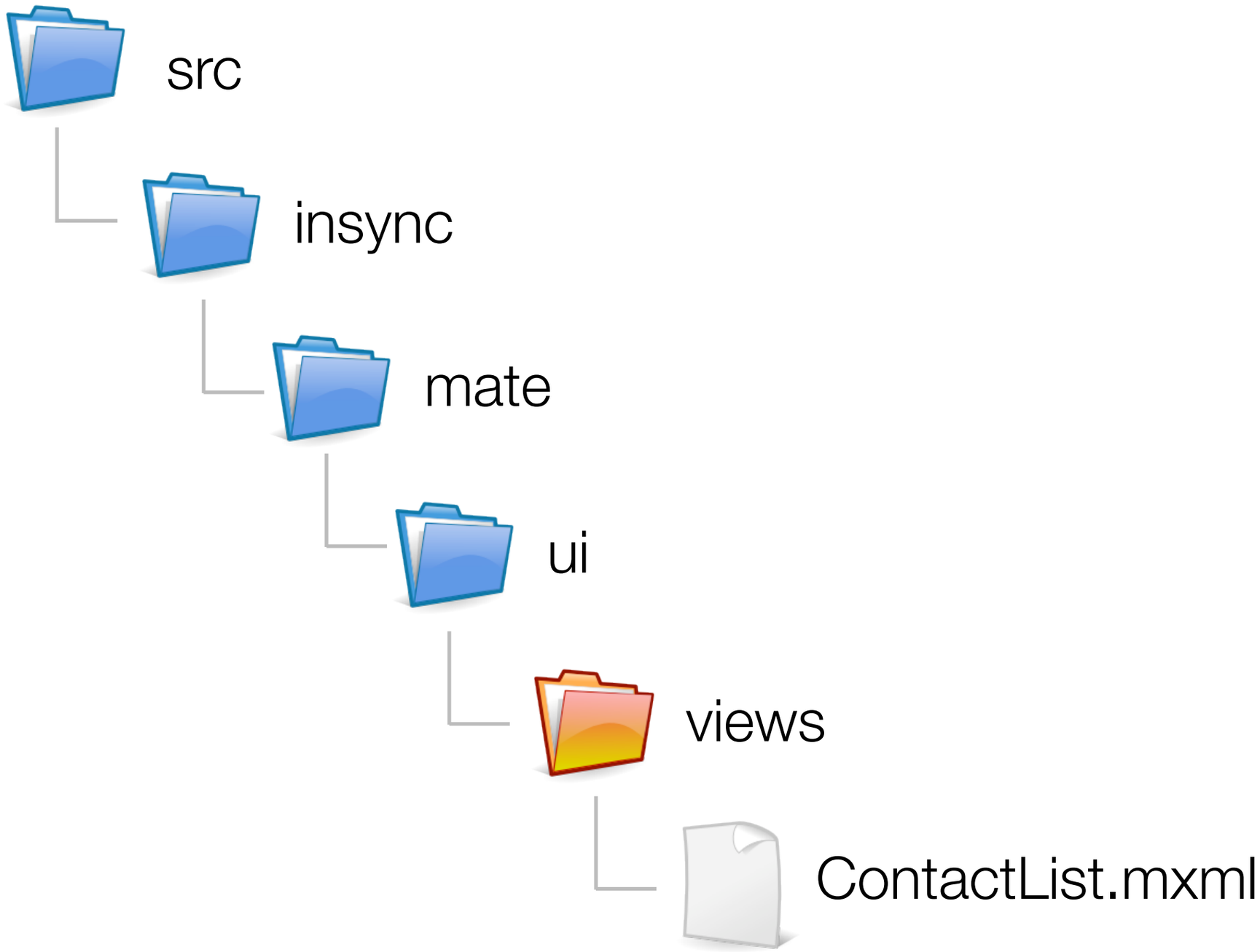
ContactList.mxml - Plain version

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml">

<mx:Script>
        [Bindable]
        public var contacts:ArrayCollection;
        private var lastSearchStr:String;


         public function search(searchStr:String):void {
             service.getContactsByName(searchStr);
             lastSearchStr = searchStr;
         }

         private function getContacts_result(event:ResultEvent):void {
              contacts = event.result as ArrayCollection;
         }

    </mx:Script>


    <mx:RemoteObject id="service" destination="contacts" endpoint="http://localhost:8400/
    messagebroker/amf">
            <mx:method name="getContactsByName" result="getContacts_result(event)"/>
    </mx:RemoteObject>


    <mx:DataGrid id="dg" dataProvider="{ contacts }"
            doubleClick="dispatchEvent(
                    new ContactEvent( ContactEvent.EDIT, dg.selectedItem as Contact) )"  ... />

</mx:Canvas>
```

Plain version



Open file (InsyncMate1):



ContactList.mxml - Mate version 1

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" width="100%" height="100%">

    <mx:Metadata>
        [Event(name="editContact", type="insync.mate.events.ContactEvent")]
    </mx:Metadata>

    <mx:Script>

            import mx.collections.ArrayCollection;
            import insync.mate.model.Contact;
            import insync.mate.events.ContactEvent;

            [Bindable]
            public var contacts:ArrayCollection;

    </mx:Script>


    <mx:DataGrid id="dg" dataProvider="{ contacts }"
        top="0" left="0"
        right="0" bottom="0"
        doubleClickEnabled="true"
        doubleClick="dispatchEvent( new ContactEvent( ContactEvent.EDIT, dg.selectedItem
as Contact) )">

        <mx:columns>
```

ContactList.mxml - Mate version 1

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" width="100%" height="100%">

    <mx:Metadata>
        [Event(name="editContact", type="insync.mate.events.ContactEvent")]
    </mx:Metadata>

    <mx:Script>

            import mx.collections.ArrayCollection;
            import insync.mate.model.Contact;
            import insync.mate.events.ContactEvent;

        [Bindable]
        public var contacts:ArrayCollection;

    </mx:Script>


    <mx:DataGrid id="dg" dataProvider="{ contacts }"
        top="0" left="0"
        right="0" bottom="0"
        doubleClickEnabled="true"
        doubleClick="dispatchEvent( new ContactEvent( ContactEvent.EDIT, dg.selectedItem
as Contact) )">

        <mx:columns>
```

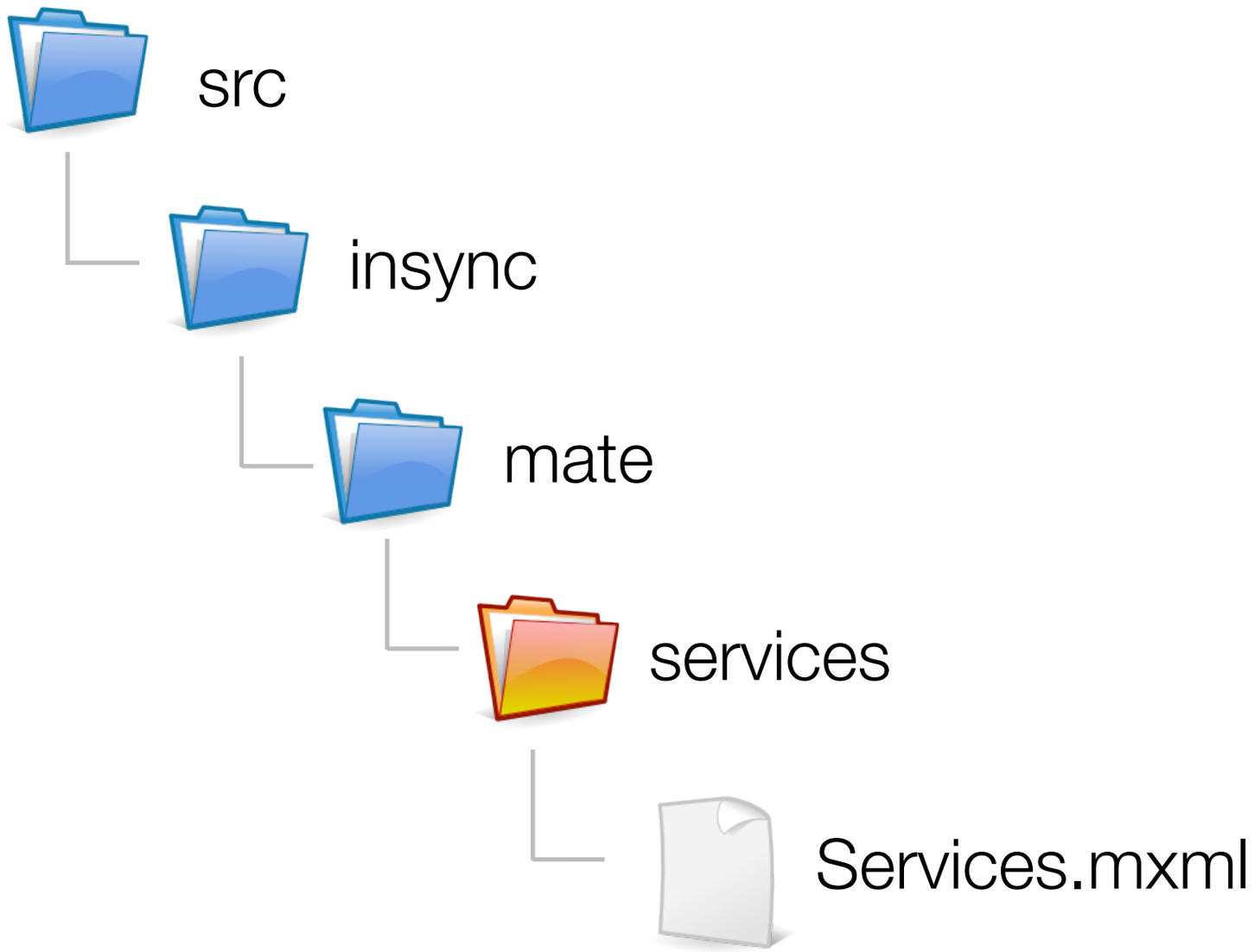
provided by injection

ContactList.mxml - Mate version 1

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" width="100%" height="100%">

    <mx:Metadata>
        [Event(name="editContact", type="insync.mate.events.ContactEvent")]
    </mx:Metadata>

    <mx:Script>

            import mx.collections.ArrayCollection;
            import insync.mate.model.Contact;
            import insync.mate.events.ContactEvent;

            [Bindable]
            public var contacts:ArrayCollection;

    </mx:Script>


    <mx:DataGrid id="dg" dataProvider="{ contacts }"
        top="0" left="0"
        right="0" bottom="0"
        doubleClickEnabled="true"
        doubleClick="dispatchEvent( new ContactEvent( ContactEvent.EDIT, dg.selectedItem
as Contact) )">

        <mx:columns>
```

- no business logic
- not dependent on service
- independent from Mate

Open file (InsyncMate1):



Services.mxml - Mate version 1

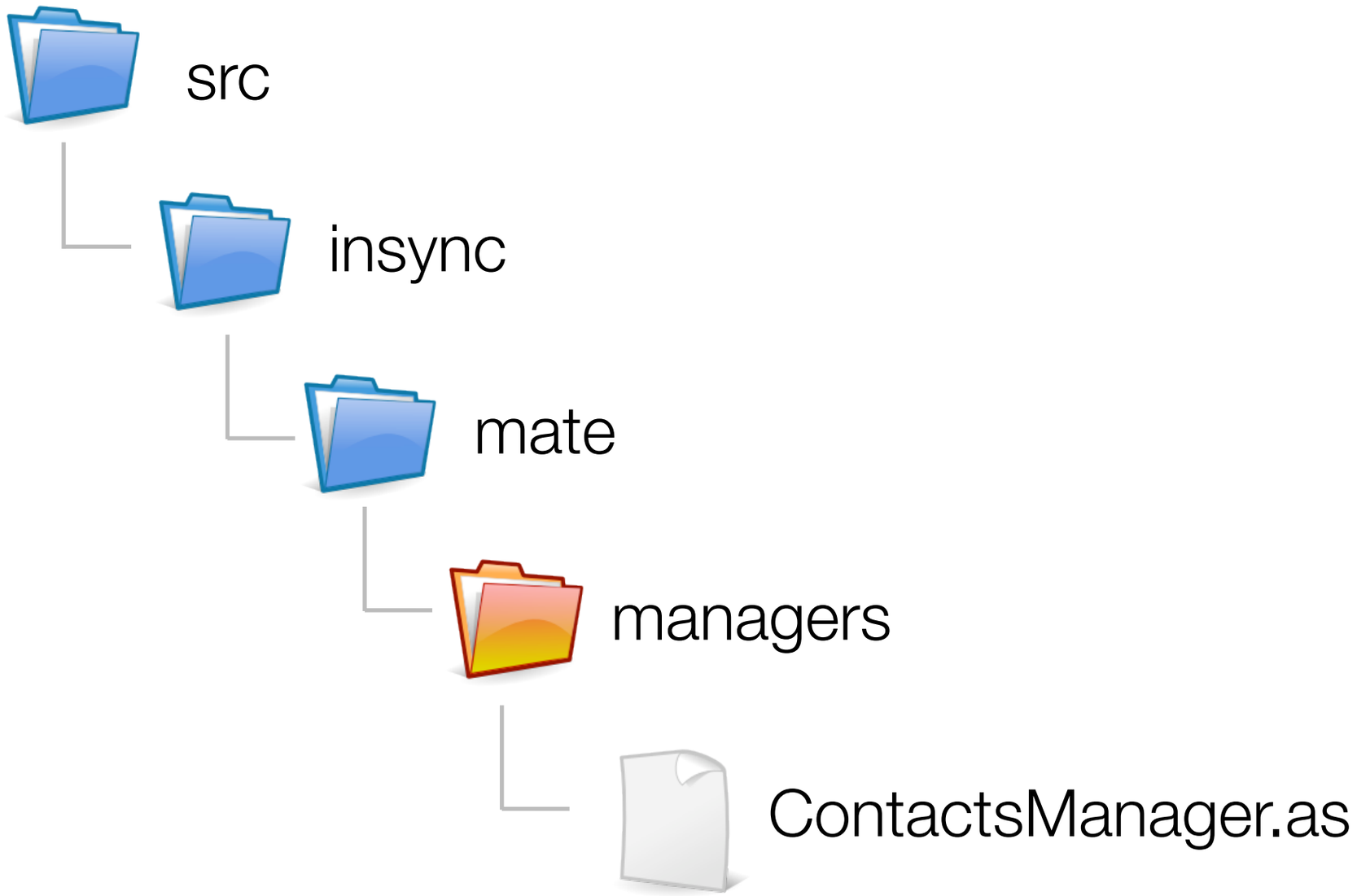
```
<Object xmlns="*" xmlns:mx="http://www.adobe.com/2006/mxml">

    <mx:RemoteObject id="contacts" destination="contacts"
    endpoint="http://localhost:8400/messagebroker/amf" />

</Object>
```

centralized services

Services.mxml - Mate version 1

```
<Object xmlns="*" xmlns:mx="http://www.adobe.com/2006/mxml">

    <MockRemoteObject id="contacts" mockGenerator="{ MockContactService }" delay="3">
            <methods>
                <MockMethod name="search" dataUrl="assets/xml/contacts.xml"/>
            </methods>
    </MockRemoteObject>

</Object>
```

Open file (InsyncMate1):



ContactsManager.as - Mate version 1

```
public class ContactsManager extends EventDispatcher {

        // property to store the last search keyword
        public var lastSearch:String = "";

        // Read-only public variable used to access the current contacts to show in list
        private var _contacts:ArrayCollection;
        [Bindable(event="contactsChanged")]
        public function get contacts():ArrayCollection
        {
            return _contacts;
        }

        // Stores given contacts and the current search keyword
        public function saveContacts( list:ArrayCollection, searchKey:String  ):void
        {
            _contacts = list;
            //trigger binding by dispatching change event
            dispatchEvent( new Event( "contactsChanged" ) );

            lastSearch = searchKey;
        }
}
```

ContactsManager.as - Mate version 1

```
public class ContactsManager extends EventDispatcher {

        // property to store the last search keyword
        public var lastSearch:String = "";

        // Read-only public variable used to access the current contacts to show in list
        private var _contacts:ArrayCollection;
        [Bindable(event="contactsChanged")]
        public function get contacts():ArrayCollection
        {
            return _contacts;
        }

        // Stores given contacts and the current search keyword
        public function saveContacts( list:ArrayCollection, searchKey:String  ):void
        {
            _contacts = list;
            //trigger binding by dispatching change event
            dispatchEvent( new Event( "contactsChanged" ) );

            lastSearch = searchKey;
        }
}
```

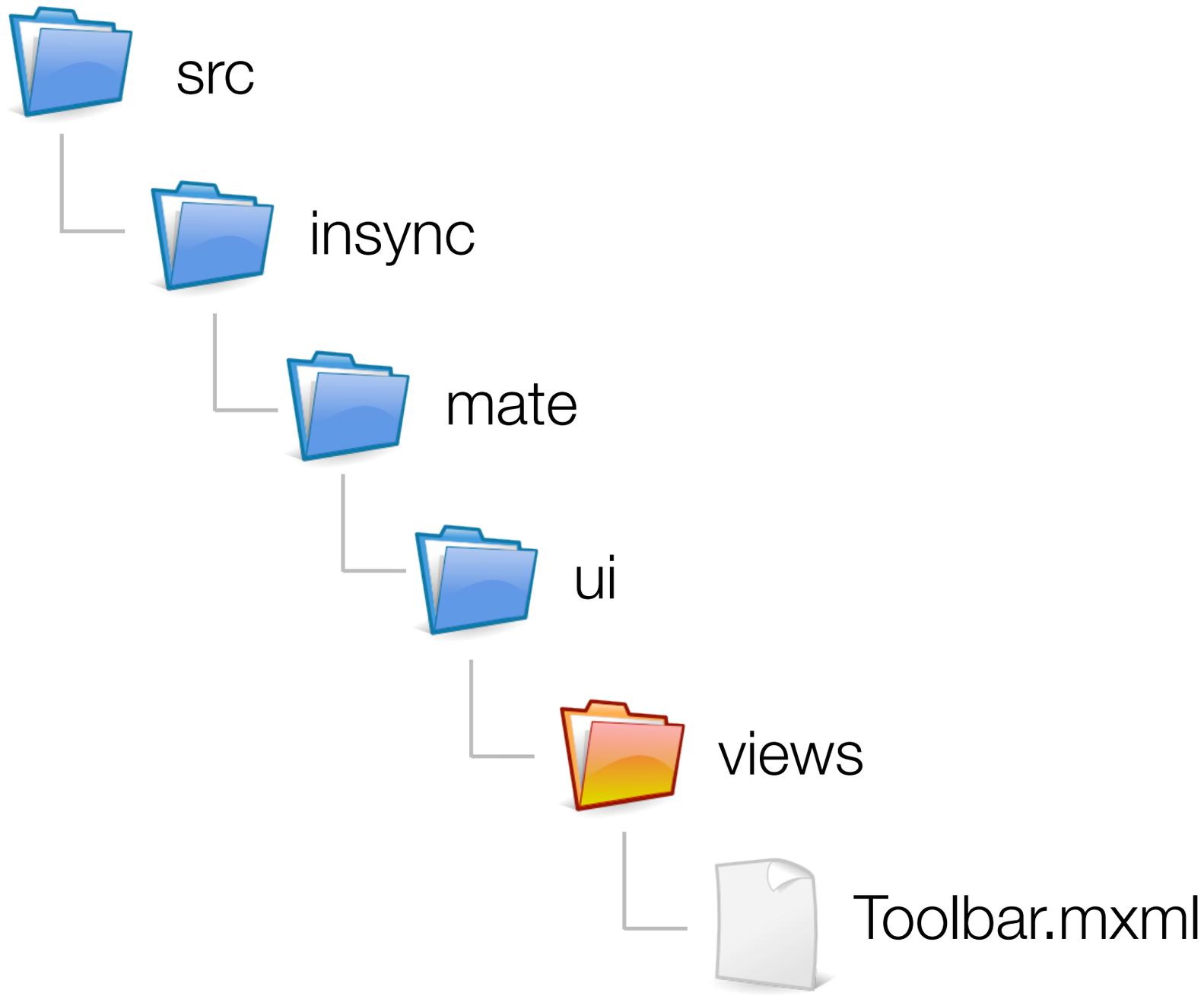
ContactsManager.as - Mate version 1

```
public class ContactsManager extends EventDispatcher {

        // property to store the last search keyword
        public var lastSearch:String = "";

        // Read-only public variable used to access the current contacts to show in list
        private var _contacts:ArrayCollection;
        [Bindable(event="contactsChanged")]
        public function get contacts():ArrayCollection
        {
            return _contacts;
        }

        // Stores given contacts and the current search keyword
        public function saveContacts( list:ArrayCollection, searchKey:String  ):void
        {
            _contacts = list;
            //trigger binding by dispatching change event
            dispatchEvent( new Event( "contactsChanged" ) );

            lastSearch = searchKey;
        }
}
```

independent from services

easy to mock and test

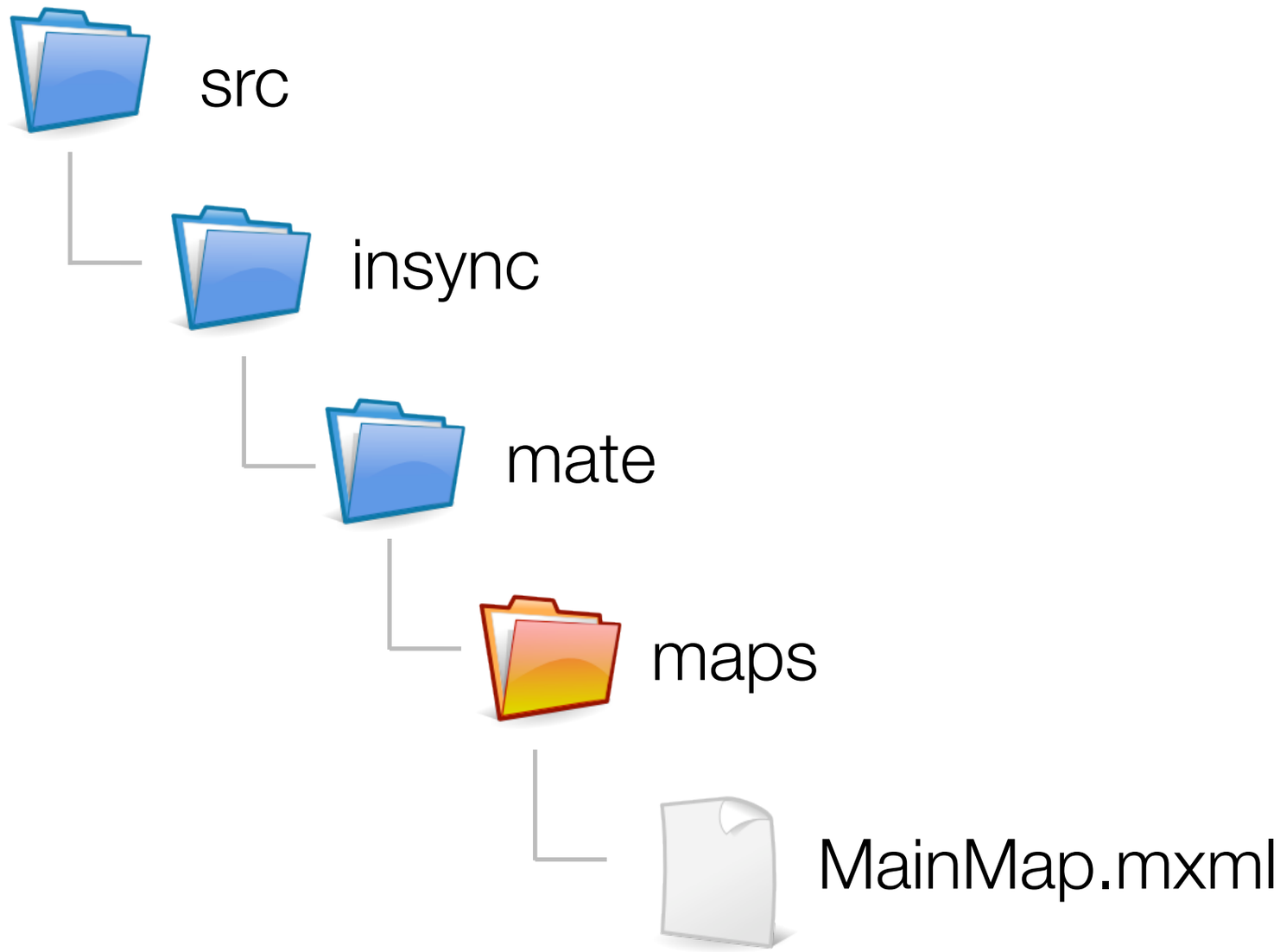
Open file (InsyncMate1):



Toolbar.mxml - Mate version 1

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" width="100%">

    <mx:Metadata>
        [Event(name="search", type="insync.plain.events.SearchEvent")]
        [Event(name="addContact", type="insync.plain.events.ContactEvent")]
    </mx:Metadata>

    <mx:Script>
        <![CDATA[
            import insync.plain.events.SearchEvent;
            import insync.plain.events.ContactEvent;
        ]]>
    </mx:Script>

    <mx:Button toolTip="Add Contact"
        click="dispatchEvent(new ContactEvent(ContactEvent.ADD))"/>

    <mx:Label text="Search:" />

    <mx:TextInput id="searchBox"
        change="dispatchEvent(new SearchEvent(SearchEvent.SEARCH, searchBox.text))"/>

</mx:Canvas>
```

Open file (InsyncMate1):



MainMap.mxml - Mate version 1

```
<EventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/"
xmlns:services="insync.mate.services.*">

    <services:Services id="services" />


    <EventHandlers type="{ SearchEvent.SEARCH }">

        <RemoteObjectInvoker instance="{ services.contacts }"
                method="getContactsByName"
                arguments="{ event.searchStr }">

            <resultHandlers>
                <MethodInvoker generator="{ ContactsManager }"
                        method="saveContacts"
                        arguments="{ [resultObject, event.searchStr] }"/>
            </resultHandlers>

        </RemoteObjectInvoker>

    </EventHandlers>


</EventMap>
```

MainMap.mxml - Mate version 1

```
<EventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/"
xmlns:services="insync.mate.services.*">

    <services:Services id="services" />


    <EventHandlers type="{ SearchEvent.SEARCH }">

        <RemoteObjectInvoker instance="{ services.contacts }"
                method="getContactsByName"
                arguments="{ event.searchStr }">

            <resultHandlers>
                <MethodInvoker generator="{ ContactsManager }"
                        method="saveContacts"
                        arguments="{ [resultObject, event.searchStr] }"/>
            </resultHandlers>

        </RemoteObjectInvoker>

    </EventHandlers>


</EventMap>
```

MainMap.mxml - Mate version 1

```
<EventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/"
xmlns:services="insync.mate.services.*">

    <services:Services id="services" />


    <EventHandlers type="{ SearchEvent.SEARCH }">

        <RemoteObjectInvoker instance="{ services.contacts }"
                method="getContactsByName"
                arguments="{ event.searchStr }">

            <resultHandlers>
                <MethodInvoker generator="{ ContactsManager }"
                        method="saveContacts"
                        arguments="{ [resultObject, event.searchStr] }"/>
            </resultHandlers>

        </RemoteObjectInvoker>

    </EventHandlers>


</EventMap>
```

MainMap.mxml - Mate version 1

```
<EventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/"
xmlns:services="insync.mate.services.*">

    <services:Services id="services" />


    <EventHandlers type="{ SearchEvent.SEARCH }">

        <RemoteObjectInvoker instance="{ services.contacts }"
                method="getContactsByName"
                arguments="{ event.searchStr }">

            <resultHandlers>
                <MethodInvoker generator="{ ContactsManager }"
                        method="saveContacts"
                        arguments="{ [resultObject, event.searchStr] }"/>
            </resultHandlers>

        </RemoteObjectInvoker>

    </EventHandlers>


</EventMap>
```

MainMap.mxml - Mate version 1

```
<EventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/"
xmlns:services="insync.mate.services.*">

    <services:Services id="services" />


    <EventHandlers type="{ SearchEvent.SEARCH }">

        <RemoteObjectInvoker instance="{ services.contacts }"
                method="getContactsByName"
                arguments="{ event.searchStr }">

            <resultHandlers>
                <MethodInvoker generator="{ ContactsManager }"
                        method="saveContacts"
                        arguments="{ [resultObject, event.searchStr] }"/>
            </resultHandlers>

        </RemoteObjectInvoker>

    </EventHandlers>


</EventMap>
```

MainMap.mxml - Mate version 1

```
<EventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/"
xmlns:services="insync.mate.services.*">

    <services:Services id="services" />


    <EventHandlers type="{ SearchEvent.SEARCH }">

        <RemoteObjectInvoker instance="{ services.contacts }"
                method="getContactsByName"
                arguments="{ event.searchStr }">

            <resultHandlers>
                <MethodInvoker generator="{ ContactsManager }"
                        method="saveContacts"
                        arguments="{ [resultObject, event.searchStr] }"/>
            </resultHandlers>

        </RemoteObjectInvoker>

    </EventHandlers>


</EventMap>
```

MainMap.mxml - Mate version 1

```
<EventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/"
xmlns:services="insync.mate.services.*">

    <services:Services id="services" />


    <EventHandlers type="{ SearchEvent.SEARCH }">

        <RemoteObjectInvoker instance="{ services.contacts }"
                method="getContactsByName"
                arguments="{ event.searchStr }">

            <resultHandlers>
                <MethodInvoker generator="{ ContactsManager }"
                        method="saveContacts"
                        arguments="{ [resultObject, event.searchStr] }"/>
            </resultHandlers>

        </RemoteObjectInvoker>

    </EventHandlers>


</EventMap>
```

service methods separated from view and business logic

Open file (InsyncMate1):

InsyncMate1.mxml (version 1)

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application
    xmlns:mx="http://www.adobe.com/2006/mxml"
    xmlns:views="insync.mate.ui.views.*"
    xmlns:maps="insync.mate.maps.*" xmlns:mate="http://mate.asfusion.com/">

    <!-- Styles ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <mx:Style source="assets/styles/styles.css" />

    <!-- EventMaps ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <maps:MainMap />

    <!-- UI ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <views:MainView/>

    <!-- Debugger ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <mate:Debugger level="{Debugger.ALL}" />

</mx:Application>
```

InsyncMate1.mxml (version 1)

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application
    xmlns:mx="http://www.adobe.com/2006/mxml"
    xmlns:views="insync.mate.ui.views.*"
    xmlns:maps="insync.mate.maps.*" xmlns:mate="http://mate.asfusion.com/">

    <!-- Styles ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <mx:Style source="assets/styles/styles.css" />

    <!-- EventMaps ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <maps:MainMap />

    <!-- UI ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <views:MainView/>

    <!-- Debugger ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <mate:Debugger level="{Debugger.ALL}" />

</mx:Application>
```

InsyncMate1.mxml (version 1)

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application
    xmlns:mx="http://www.adobe.com/2006/mxml"
    xmlns:views="insync.mate.ui.views.*"
    xmlns:maps="insync.mate.maps.*" xmlns:mate="http://mate.asfusion.com/">

    <!-- Styles ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <mx:Style source="assets/styles/styles.css" />

    <!-- EventMaps ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <maps:MainMap />

    <!-- UI ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <views:MainView/>

    <!-- Debugger ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <mate:Debugger level="{Debugger.ALL}" />

</mx:Application>
```

Debugging Output

```
<EventHandlers (started)    type="SearchEvent.SEARCH" (search)   priority="0"
useCapture="false"   useWeakReference="true"   dispatcherType="inherit"   scope="[object
Scope]">

    <RemoteObjectInvoker   instance="[RemoteObject  destination="contacts"
channelSet="[ChannelSet null ]"]"   arguments="Pam"   showBusyCursor="false"
makeObjectsBindable="true"   method="getContactsByName"   requestTimeout="0"
debug="false"/>

</EventHandlers (end)    type="SearchEvent.SEARCH" (search)>
```

Debugging Output

```
<EventHandlers (started)    type="SearchEvent.SEARCH" (search)   priority="0"
useCapture="false"   useWeakReference="true"   dispatcherType="inherit"   scope="[object
Scope]">

    <RemoteObjectInvoker   instance="[RemoteObject  destination="contacts"
channelSet="[ChannelSet null ]"]"   arguments="Pam"   showBusyCursor="false"
makeObjectsBindable="true"   method="getContactsByName"   requestTimeout="0"
debug="false"/>

</EventHandlers (end)    type="SearchEvent.SEARCH" (search)>
```

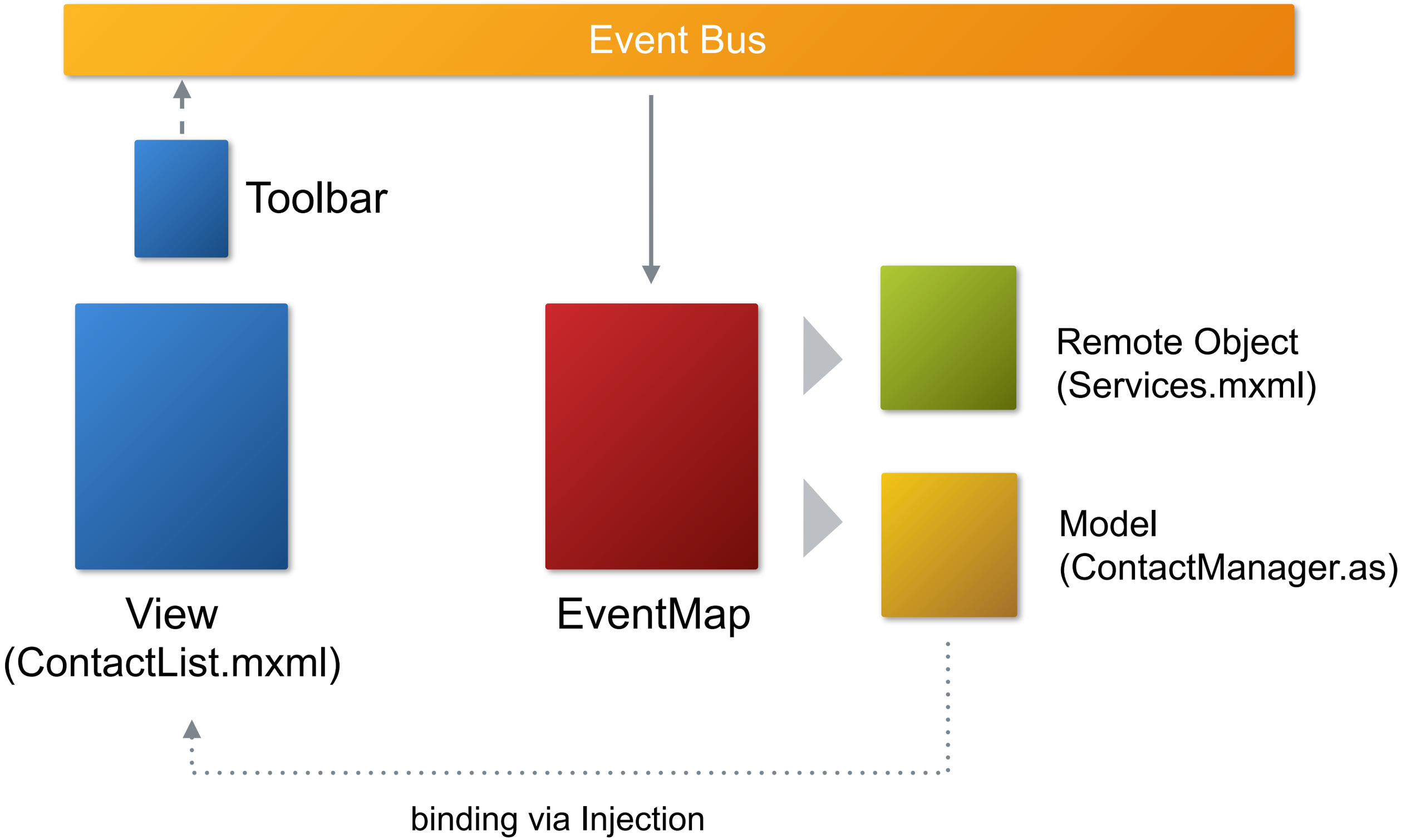
```
<RemoteObjectInvoker instance="{ services.contacts }"
          method="getContactsByName" debug="true"
          arguments="{ event.searchStr }">
```

Debugging Output

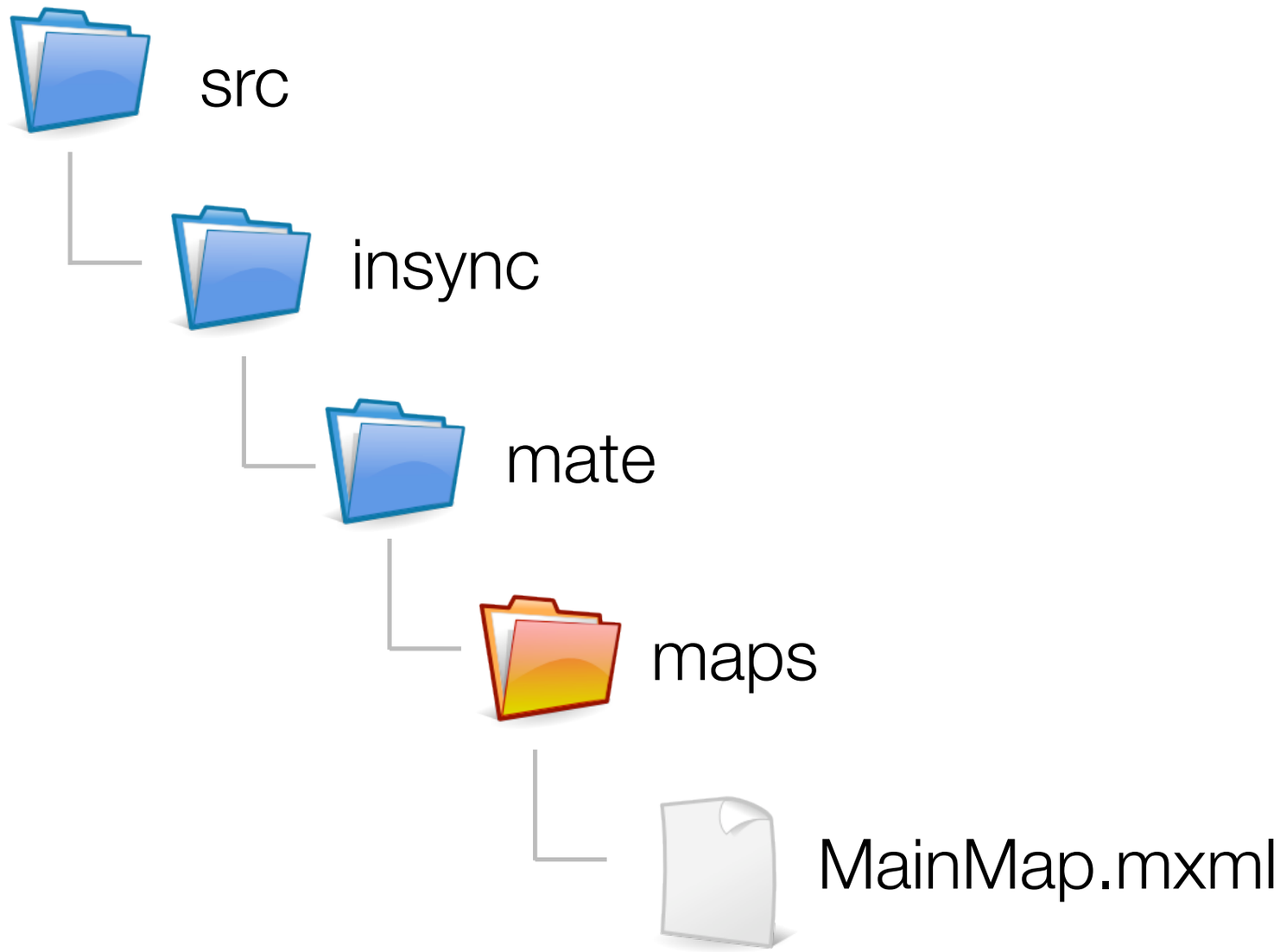
```
<EventHandlers (started)    type="SearchEvent.SEARCH" (search)   priority="0"
useCapture="false"   useWeakReference="true"   dispatcherType="inherit"   scope="[object
Scope]">

    <RemoteObjectInvoker   instance="[RemoteObject  destination="contacts"
channelSet="[ChannelSet null ]"]"   arguments="Pam"   showBusyCursor="false"
makeObjectsBindable="true"   method="getContactsByName"   requestTimeout="0"
debug="false"/>

</EventHandlers (end)    type="SearchEvent.SEARCH" (search)>
```

```
<RemoteObjectInvoker instance="{ services.contacts }"
          method="getContactsByName" debug="true"
          arguments="{ event.searchStr }">

<ServiceHandlers (started)    type="SearchEvent.SEARCH" (search)   token="[object
AsyncToken]"   priority="0"   useCapture="false"   useWeakReference="true"
scope="[object ServiceScope]"   dispatcherType="local">

    <MethodInvoker   method="saveContacts"   arguments="[ [object Contact], Pam ]"
generator="[class ContactsManager]"   cache="inherit"   registerTarget="true"/>

</ServiceHandlers (end)    type="SearchEvent.SEARCH" (search)>
```

Event Handling



Open file (InsyncMate1):



MainMap.mxml - Mate version 1

```
<EventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/">

    <Injectors target="{ ContactList }">
        <PropertyInjector targetKey="contacts"
            source="{ ContactsManager }" sourceKey="contacts"/>
    </Injectors>

</EventMap>
```

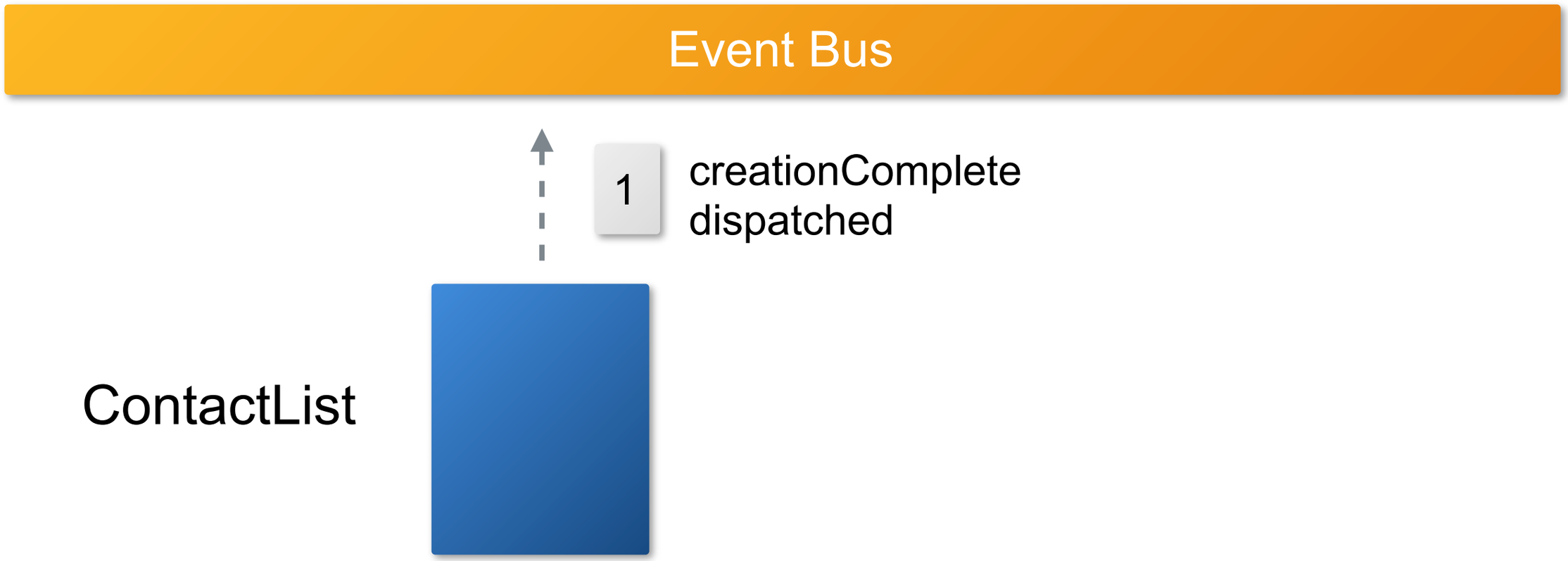
Takes advantage of bindings

Injection

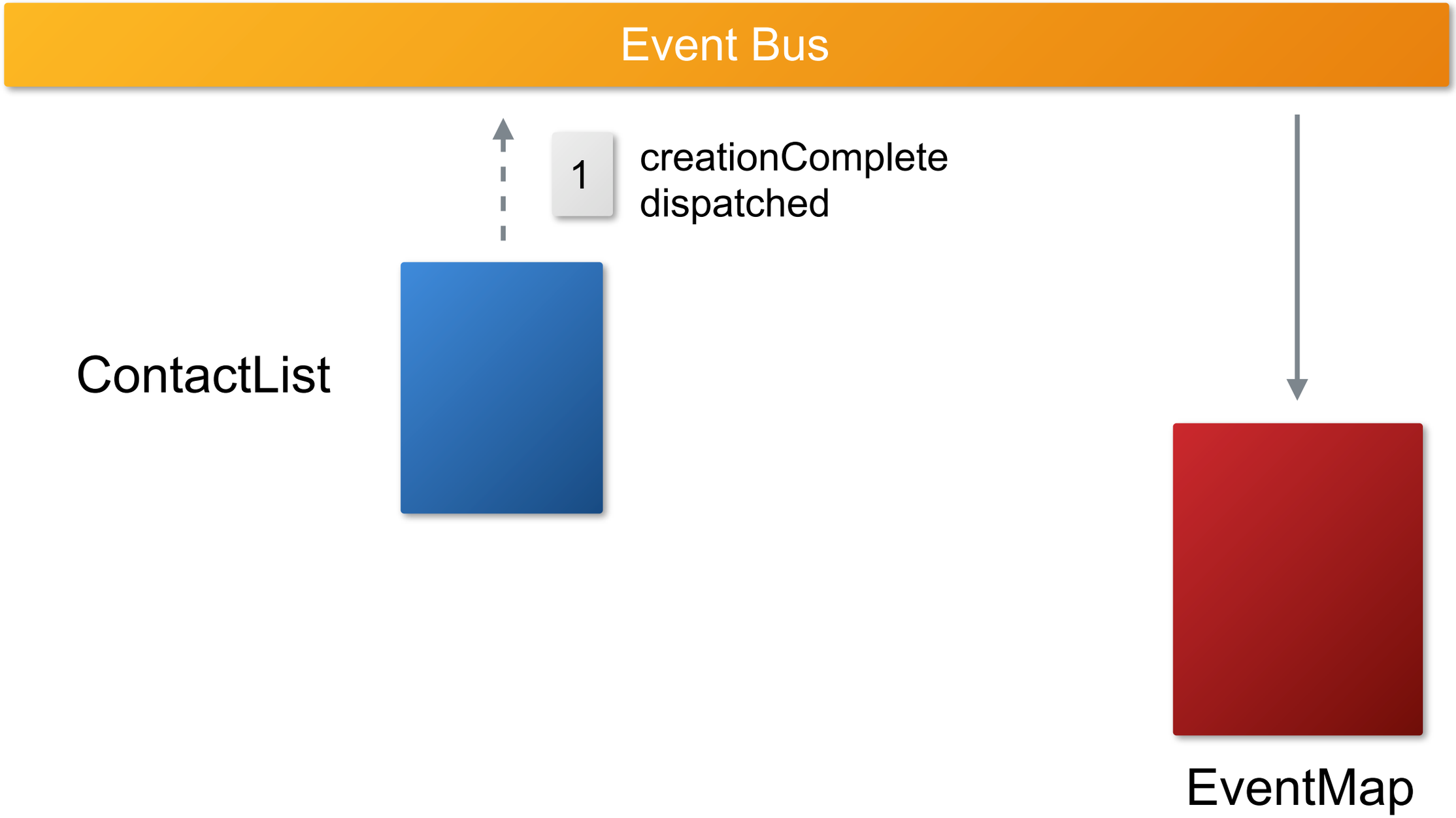
Event Bus

ContactList

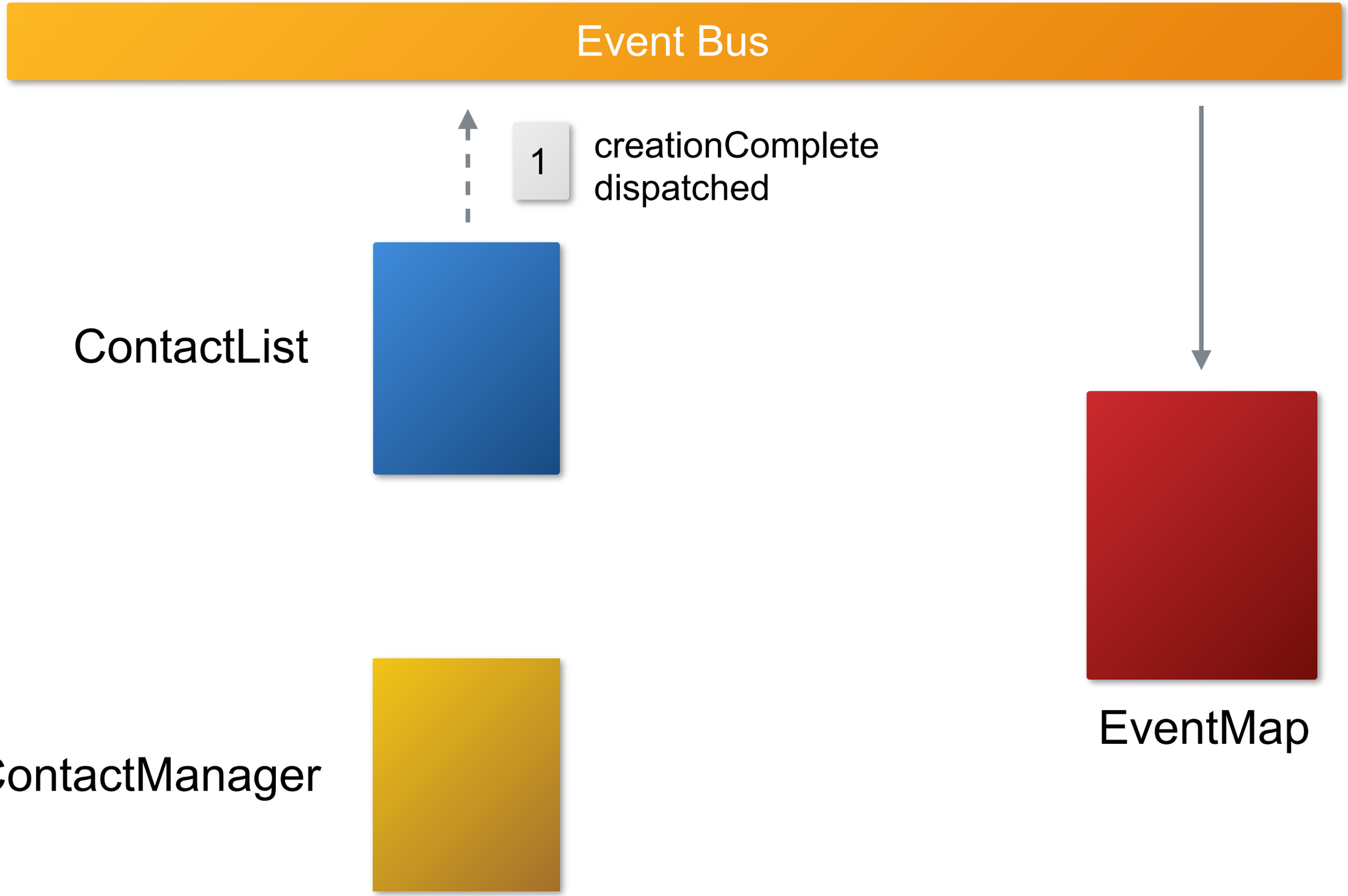
Injection



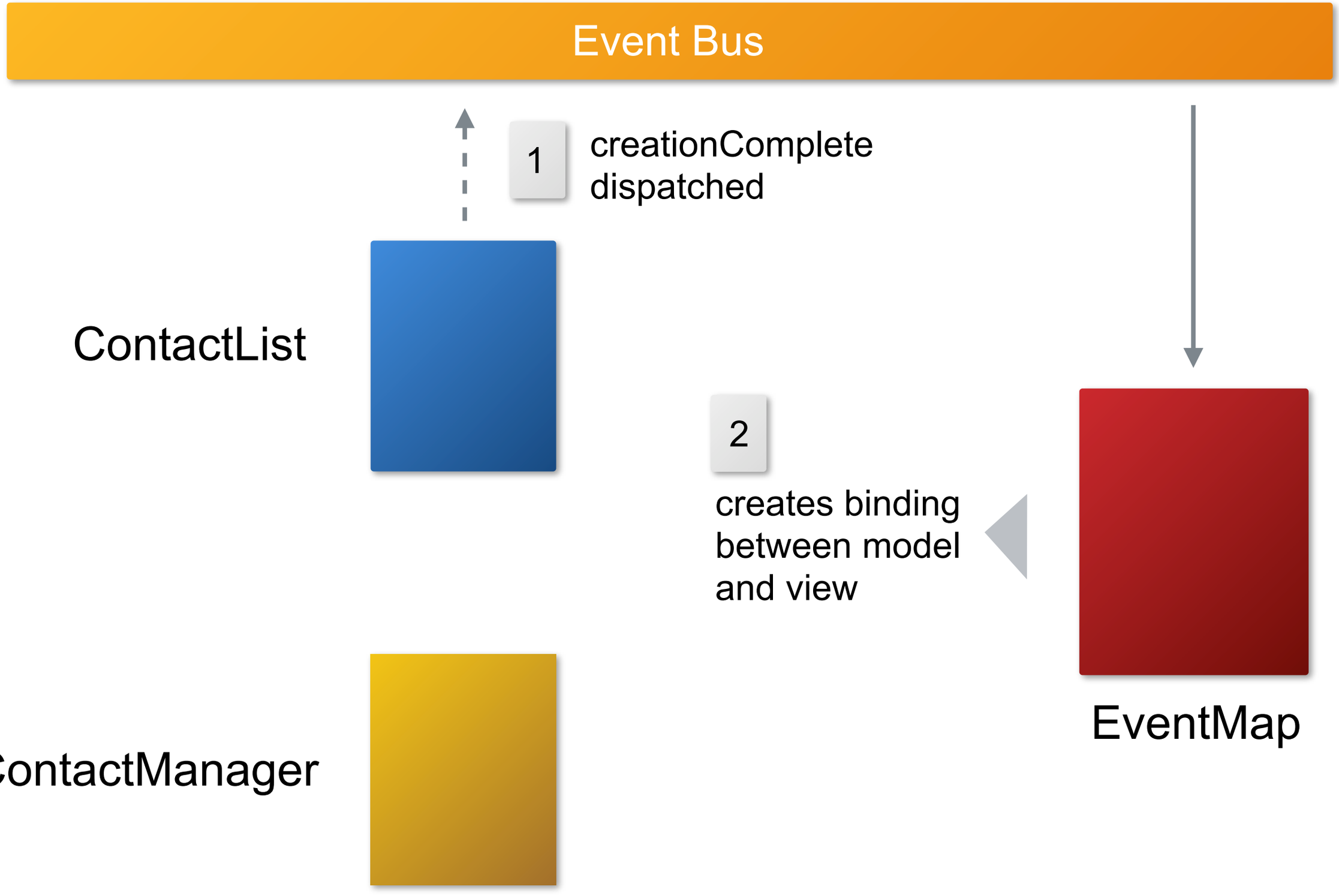
Injection



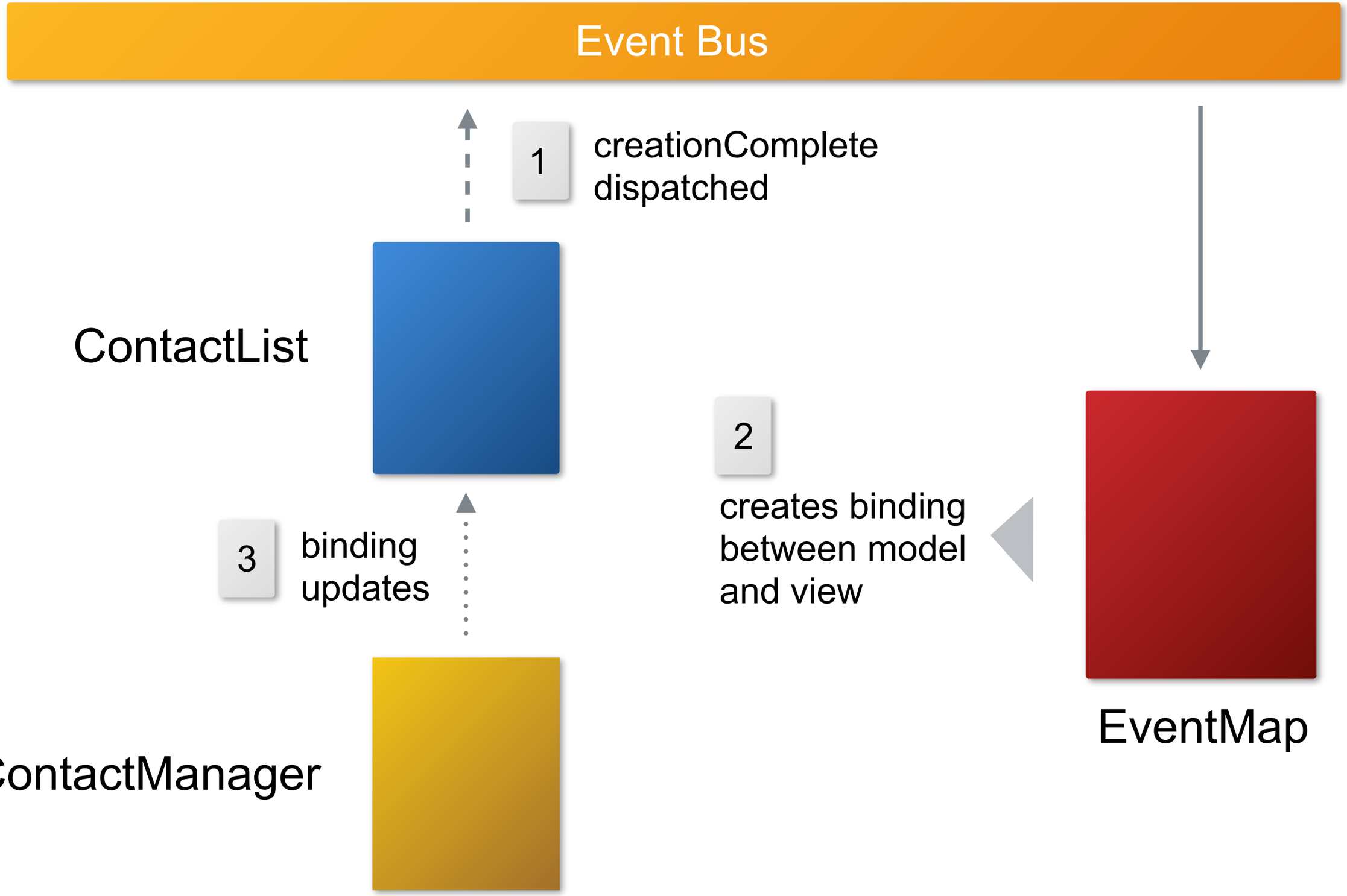
Injection



Injection



Injection



Injection

Event Bus

ContactList

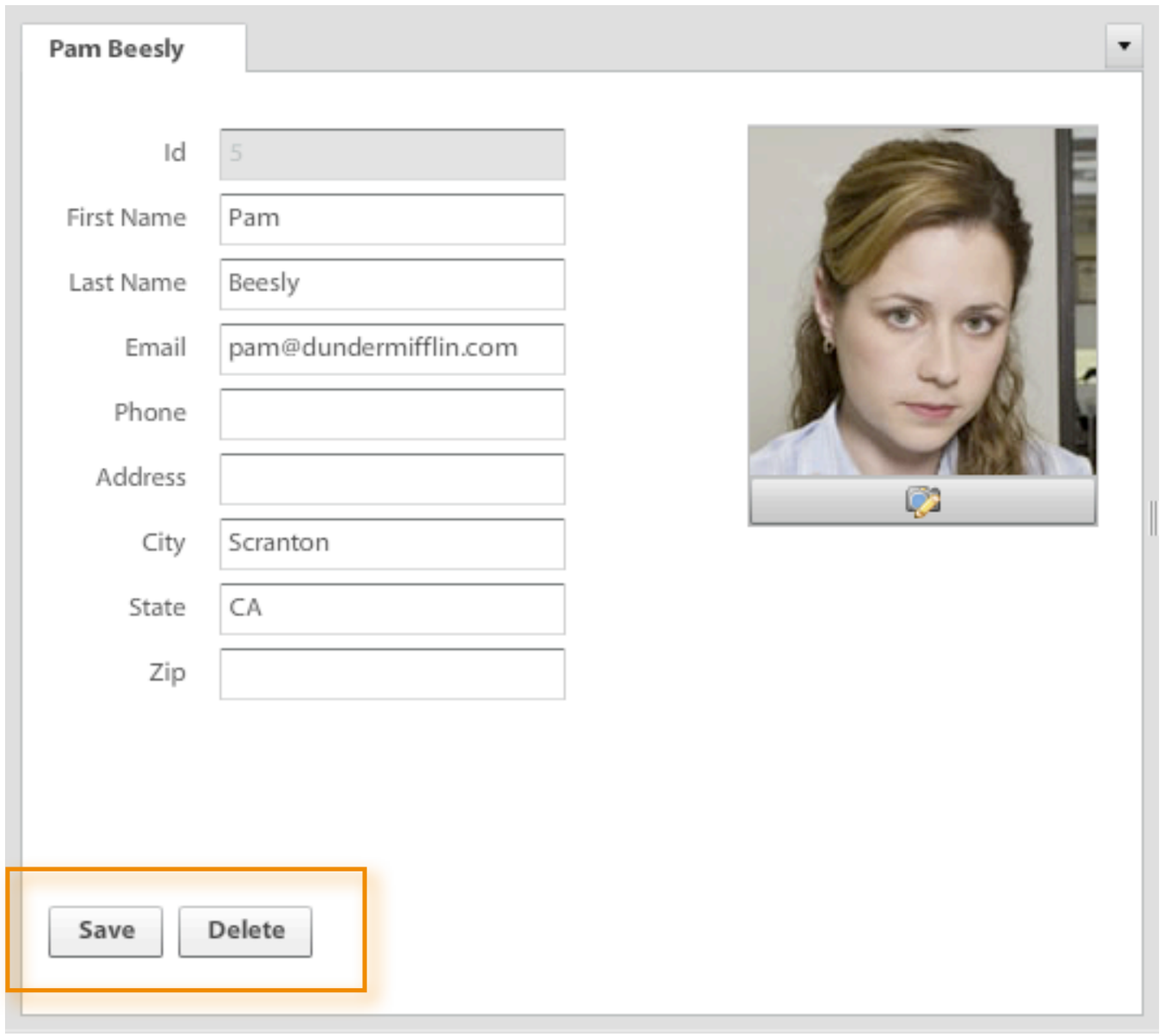
3

binding
updates

ContactManager

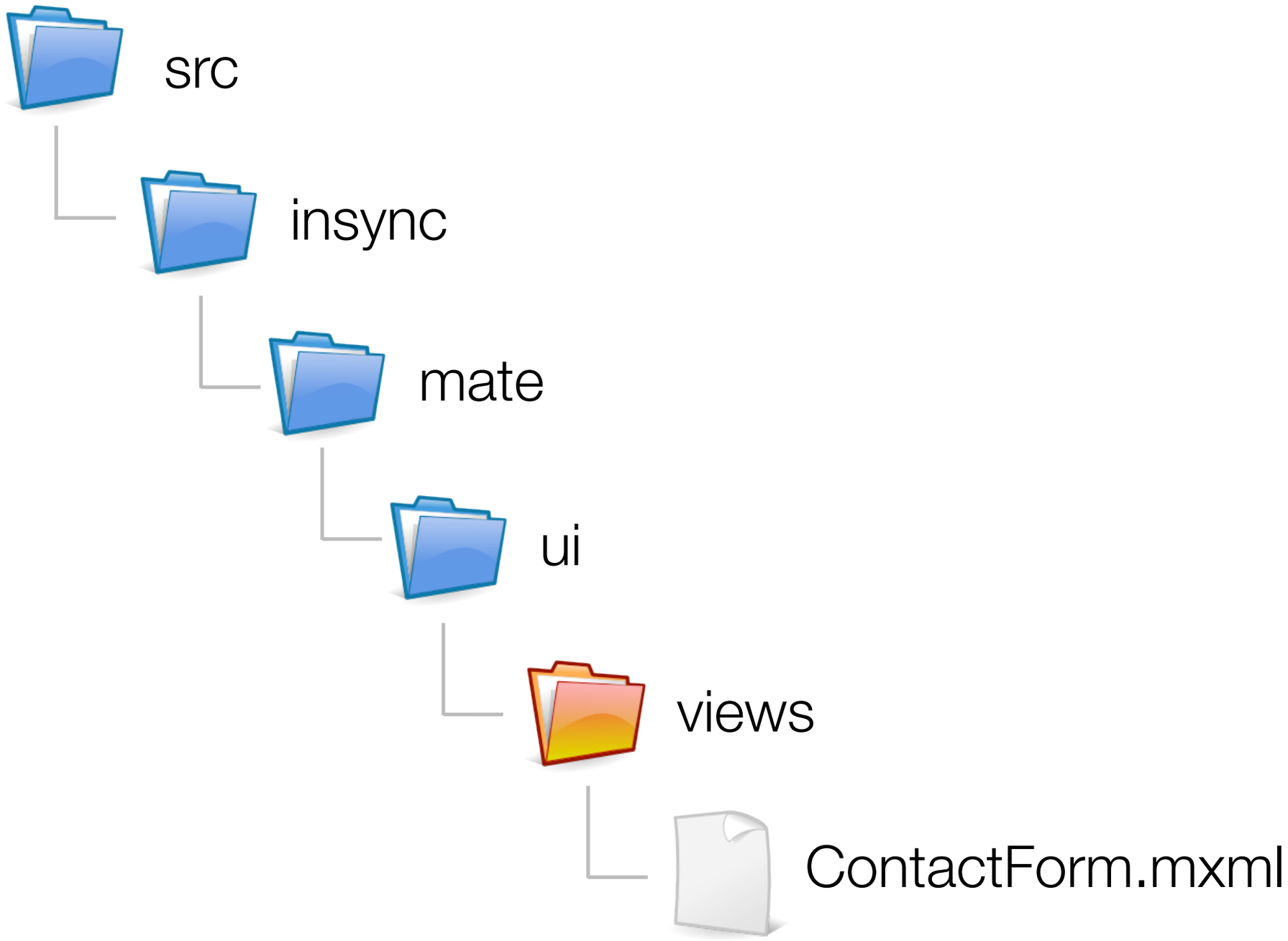
Version 1

Contact Form



`ContactEvent.SAVE` `ContactEvent.DELETE`

Open file (InsyncMate2):



```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" label="{contact.id>0?contact.fullName:'New Contact'}">

    <mx:Script>
        <![CDATA[
            [Bindable]
            public var contact:Contact;

            private function save():void
            {
                contact.firstName = firstName.text;
                contact.lastName = lastName.text;
                contact.email = email.text;
                contact.phone = phone.text;
                contact.address = address.text;
                contact.city = city.text;
                contact.state = state.text;
                contact.zip = zip.text;
                contact.pic = picture.source;

                dispatchEvent( new ContactEvent( ContactEvent.SAVE, contact ) );
            }

            private function remove():void
            {
                dispatchEvent( new ContactEvent( ContactEvent.DELETE, contact ) );
            }

            public function save_result( ):void
            {
                status.text = "Contact saved successfully";
            }

        ]]>
    </mx:Script>
```

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" label="{contact.id>0?contact.fullName:'New Contact'}">

    <mx:Script>
        <![CDATA[
            [Bindable]
            public var contact:Contact;

            private function save():void
            {
                contact.firstName = firstName.text;
                contact.lastName = lastName.text;
                contact.email = email.text;
                contact.phone = phone.text;
                contact.address = address.text;
                contact.city = city.text;
                contact.state = state.text;
                contact.zip = zip.text;
                contact.pic = picture.source;

                dispatchEvent( new ContactEvent( ContactEvent.SAVE, contact ) );
            }

            private function remove():void
            {
                dispatchEvent( new ContactEvent( ContactEvent.DELETE, contact ) );
            }

            public function save_result( ):void
            {
                status.text = "Contact saved successfully";
            }

        ]]>
    </mx:Script>
```

```
<mx:Form>
      <mx:FormItem label="Id">
          <mx:TextInput text="{contact.id}" enabled="false"/>
      </mx:FormItem>
      <mx:FormItem label="First Name">
          <mx:TextInput id="firstName" text="{contact.firstName}"/>
      </mx:FormItem>
      <mx:FormItem label="Last Name">
          <mx:TextInput id="lastName" text="{contact.lastName}"/>
      </mx:FormItem>
      <mx:FormItem label="Email">
          <mx:TextInput id="email" text="{contact.email}"/>
      </mx:FormItem>
      <-- all of the other fields here -->
  </mx:Form>

   <controls:PictureInput id="picture" source="{contact.pic}"/>

   <mx:Button label="Save" click="save()"/>
   <mx:Button label="Delete" click="remove()"/>

   <mx:Label id="status" />

</mx:Canvas>
```

```
<mx:Form>
      <mx:FormItem label="Id">
          <mx:TextInput text="{contact.id}" enabled="false"/>
      </mx:FormItem>
      <mx:FormItem label="First Name">
          <mx:TextInput id="firstName" text="{contact.firstName}"/>
      </mx:FormItem>
      <mx:FormItem label="Last Name">
          <mx:TextInput id="lastName" text="{contact.lastName}"/>
      </mx:FormItem>
      <mx:FormItem label="Email">
          <mx:TextInput id="email" text="{contact.email}"/>
      </mx:FormItem>
      <-- all of the other fields here -->
  </mx:Form>

   <controls:PictureInput id="picture" source="{contact.pic}"/>

    <mx:Button label="Save" click="save()"/>
    <mx:Button label="Delete" click="remove()"/>

    <mx:Label id="status" />

</mx:Canvas>
```

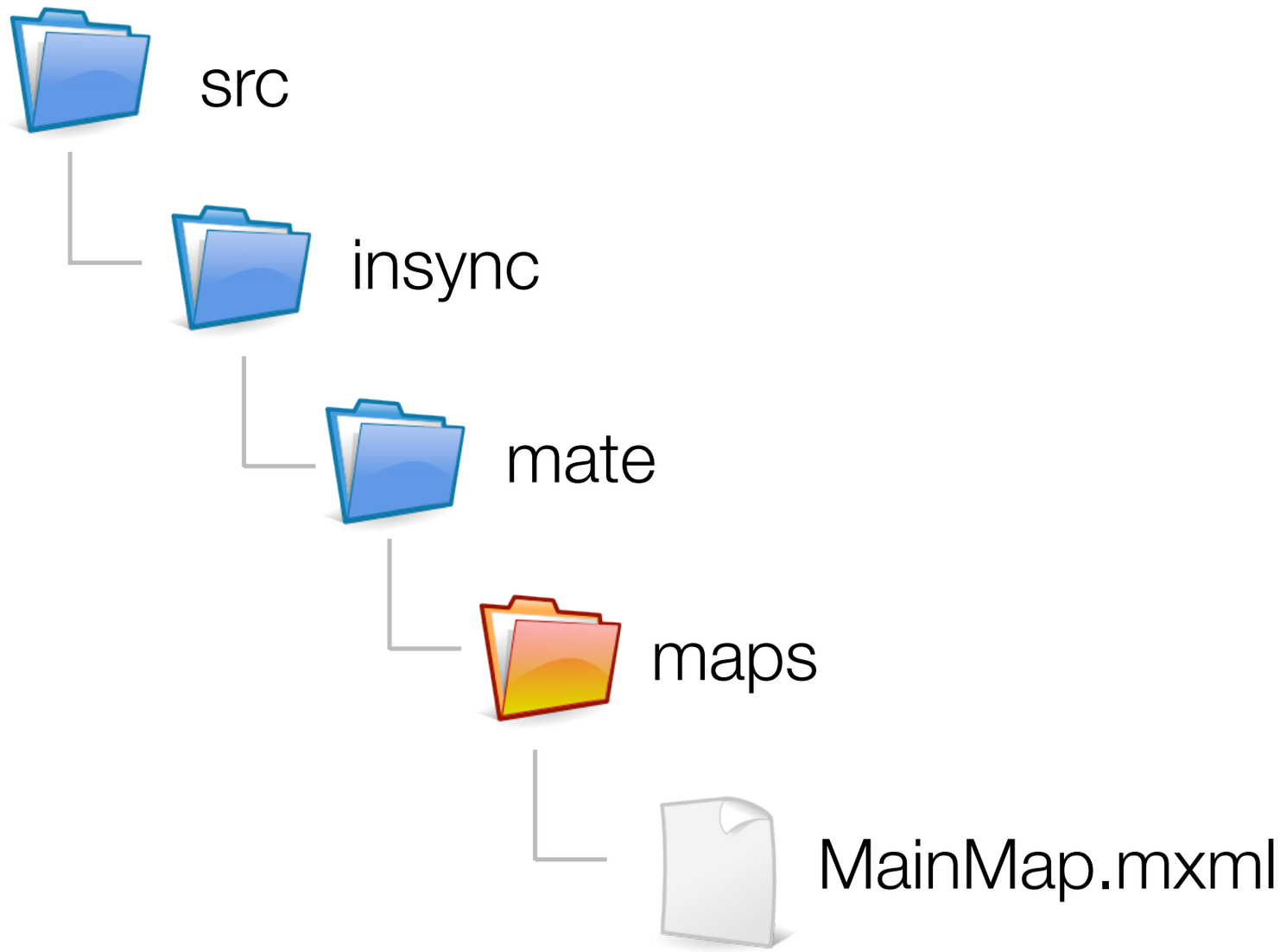
```
<mx:Form>
      <mx:FormItem label="Id">
          <mx:TextInput text="{contact.id}" enabled="false"/>
      </mx:FormItem>
      <mx:FormItem label="First Name">
          <mx:TextInput id="firstName" text="{contact.firstName}"/>
      </mx:FormItem>
      <mx:FormItem label="Last Name">
          <mx:TextInput id="lastName" text="{contact.lastName}"/>
      </mx:FormItem>
      <mx:FormItem label="Email">
          <mx:TextInput id="email" text="{contact.email}"/>
      </mx:FormItem>
      <-- all of the other fields here -->
  </mx:Form>

   <controls:PictureInput id="picture" source="{contact.pic}"/>

   <mx:Button label="Save" click="save()"/>
   <mx:Button label="Delete" click="remove()"/>

   <mx:Label id="status" />

</mx:Canvas>
```

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" label="{contact.id>0?contact.fullName:'New Contact'}">

    <mx:Script>
        <![CDATA[
            [Bindable]
            public var contact:Contact;

            private function save():void
            {
                contact.firstName = firstName.text;
                contact.lastName = lastName.text;
                contact.email = email.text;
                contact.phone = phone.text;
                contact.address = address.text;
                contact.city = city.text;
                contact.state = state.text;
                contact.zip = zip.text;
                contact.pic = picture.source;

                dispatchEvent( new ContactEvent( ContactEvent.SAVE, contact ) );
            }

            private function remove():void
            {
                dispatchEvent( new ContactEvent( ContactEvent.DELETE, contact ) );
            }

            public function save_result( ):void
            {
                status.text = "Contact saved successfully";
            }

        ]]>
    </mx:Script>
```

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" label="{contact.id>0?contact.fullName:'New Contact'}">

    <mx:Script>
        <![CDATA[
            [Bindable]
            public var contact:Contact;

            private function save():void
            {
                contact.firstName = firstName.text;
                contact.lastName = lastName.text;
                contact.email = email.text;
                contact.phone = phone.text;
                contact.address = address.text;
                contact.city = city.text;
                contact.state = state.text;
                contact.zip = zip.text;
                contact.pic = picture.source;

                dispatchEvent( new ContactEvent( ContactEvent.SAVE, contact ) );
            }

            private function remove():void
            {
                dispatchEvent( new ContactEvent( ContactEvent.DELETE, contact ) );
            }

            public function save_result( ):void
            {
                status.text = "Contact saved successfully";
            }

        ]]>
    </mx:Script>
```

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" label="{contact.id>0?contact.fullName:'New Contact'}">

    <mx:Script>
        <![CDATA[
            [Bindable]
            public var contact:Contact;

            private function save():void
            {
                contact.firstName = firstName.text;
                contact.lastName = lastName.text;
                contact.email = email.text;
                contact.phone = phone.text;
                contact.address = address.text;
                contact.city = city.text;
                contact.state = state.text;
                contact.zip = zip.text;
                contact.pic = picture.source;

                dispatchEvent( new ContactEvent( ContactEvent.SAVE, contact ) );
            }

            private function remove():void
            {
                dispatchEvent( new ContactEvent( ContactEvent.DELETE, contact ) );
            }

            public function save_result( ):void
            {
                status.text = "Contact saved successfully";
            }

        ]]>
    </mx:Script>
```

Open file (InsyncMate1):



```
<EventMap>

    <EventHandlers type="{ ContactEvent.SAVE }">

        <RemoteObjectInvoker instance="{ services.contacts }"
            method="save"
            arguments="{ event.contact }">

            <resultHandlers>

                <CallBack method="save_result"/>

                <EventAnnouncer generator="{ SearchEvent }"
                    type="{ SearchEvent.SEARCH }">
                    <Property targetKey="searchStr" source="{ ContactsManager }"
                        sourceKey="lastSearch"/>
                </EventAnnouncer>

            </resultHandlers>

        </RemoteObjectInvoker>

    </EventHandlers>



</EventMap>
```

```
<EventMap>

    <EventHandlers type="{ ContactEvent.SAVE }">

        <RemoteObjectInvoker instance="{ services.contacts }"
            method="save"
            arguments="{ event.contact }">

            <resultHandlers>

                <CallBack method="save_result"/>

                <EventAnnouncer generator="{ SearchEvent }"
                    type="{ SearchEvent.SEARCH }">
                    <Property targetKey="searchStr" source="{ ContactsManager }"
                        sourceKey="lastSearch"/>
                </EventAnnouncer>

            </resultHandlers>

        </RemoteObjectInvoker>

    </EventHandlers>



</EventMap>
```

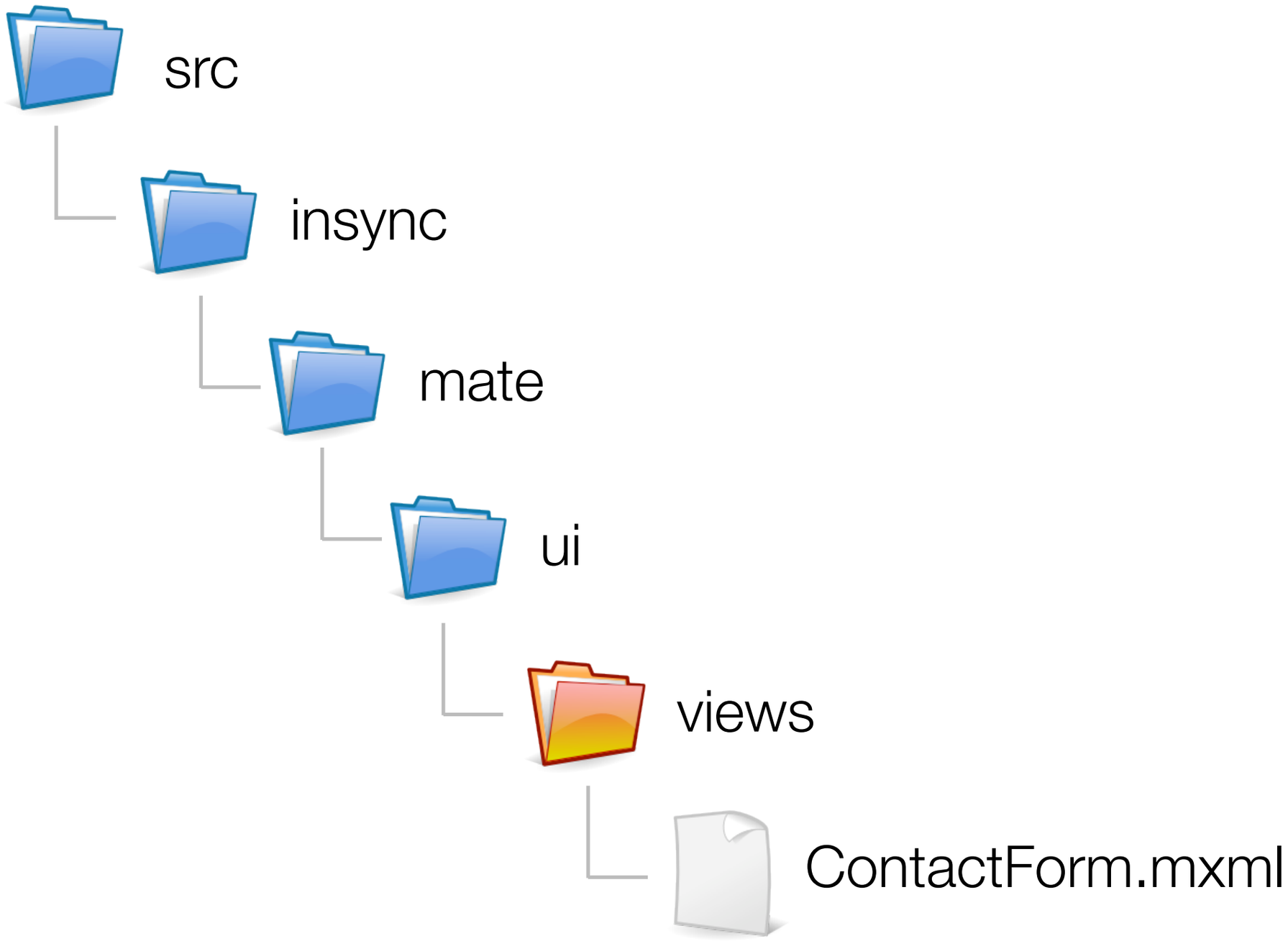
```
<EventMap>

    <EventHandlers type="{ ContactEvent.SAVE }">

        <RemoteObjectInvoker instance="{ services.contacts }"
            method="save"
            arguments="{ event.contact }">

            <resultHandlers>

                <CallBack method="save_result"/>

                <EventAnnouncer generator="{ SearchEvent }"
                    type="{ SearchEvent.SEARCH }">
                    <Property targetKey="searchStr" source="{ ContactsManager }"
                        sourceKey="lastSearch"/>
                </EventAnnouncer>

            </resultHandlers>

        </RemoteObjectInvoker>

    </EventHandlers>



</EventMap>
```

```
<EventMap>

    <EventHandlers type="{ ContactEvent.SAVE }">

        <RemoteObjectInvoker instance="{ services.contacts }"
            method="save"
            arguments="{ event.contact }">

            <resultHandlers>

                <CallBack method="save_result"/>

                <EventAnnouncer generator="{ SearchEvent }"
                    type="{ SearchEvent.SEARCH }">
                    <Property targetKey="searchStr" source="{ ContactsManager }"
                        sourceKey="lastSearch"/>
                </EventAnnouncer>

            </resultHandlers>

        </RemoteObjectInvoker>

    </EventHandlers>



</EventMap>
```

```
<EventMap>

    <EventHandlers type="{ ContactEvent.SAVE }">

        <RemoteObjectInvoker instance="{ services.contacts }"
            method="save"
            arguments="{ event.contact }">

            <resultHandlers>

                <CallBack method="save_result"/>

                <EventAnnouncer generator="{ SearchEvent }"
                    type="{ SearchEvent.SEARCH }">
                    <Property targetKey="searchStr" source="{ ContactsManager }"
                        sourceKey="lastSearch"/>
                </EventAnnouncer>

            </resultHandlers>

        </RemoteObjectInvoker>

    </EventHandlers>



</EventMap>
```

```
<EventMap>

    <EventHandlers type="{ ContactEvent.SAVE }">

        <RemoteObjectInvoker instance="{ services.contacts }"
            method="save"
            arguments="{ event.contact }">

            <resultHandlers>

                <CallBack method="save_result"/>

                <EventAnnouncer generator="{ SearchEvent }"
                    type="{ SearchEvent.SEARCH }">
                    <Property targetKey="searchStr" source="{ ContactsManager }"
                        sourceKey="lastSearch"/>
                </EventAnnouncer>

            </resultHandlers>

        </RemoteObjectInvoker>

    </EventHandlers>



</EventMap>
```

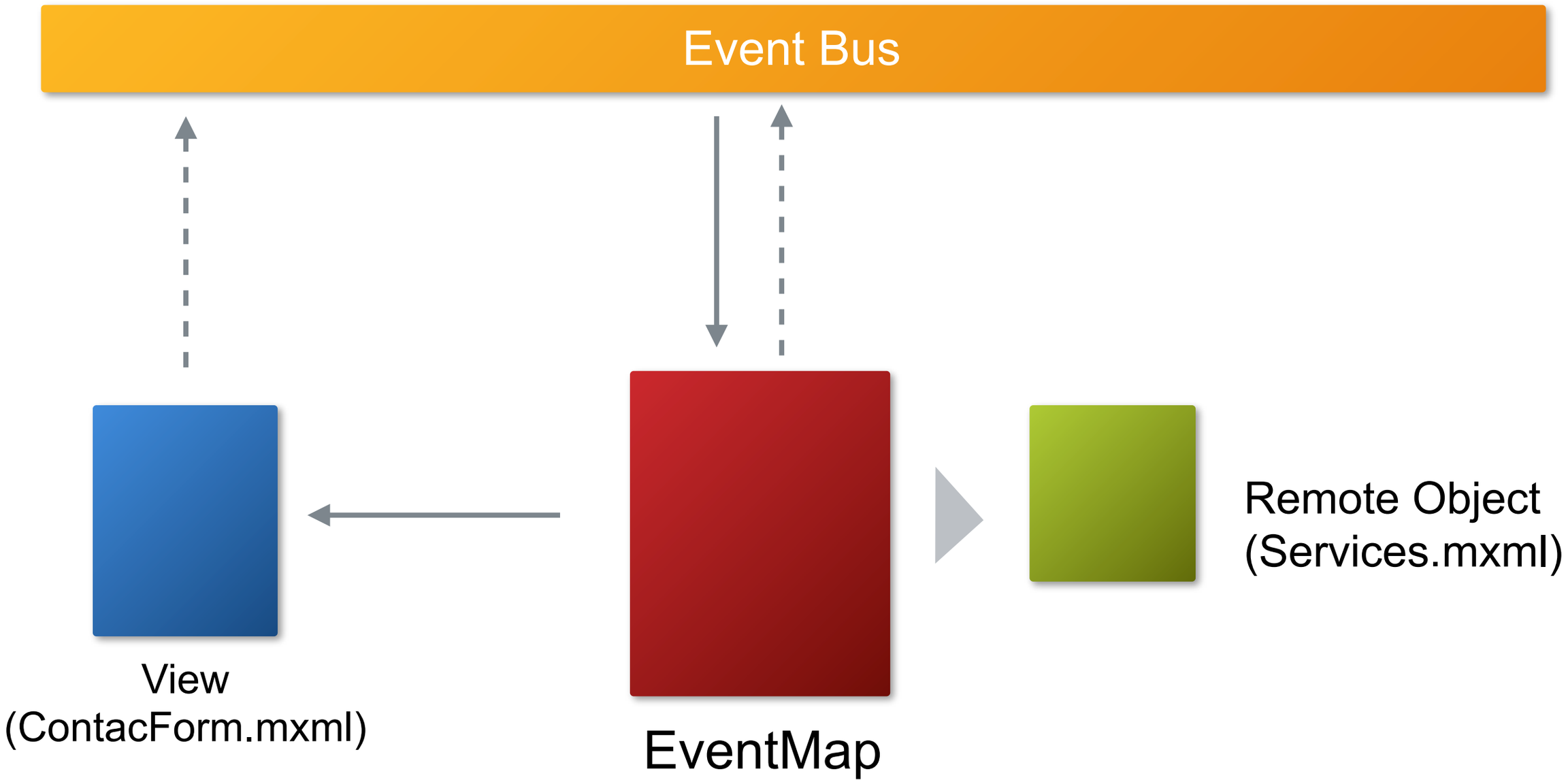
Open file (InsyncMate1):



```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" label="{contact.id>0?contact.fullName:'New Contact'}">

    <mx:Script>
        <![CDATA[
            [Bindable]
            public var contact:Contact;

            private function save():void
            {
                contact.firstName = firstName.text;
                contact.lastName = lastName.text;
                contact.email = email.text;
                contact.phone = phone.text;
                contact.address = address.text;
                contact.city = city.text;
                contact.state = state.text;
                contact.zip = zip.text;
                contact.pic = picture.source;

                dispatchEvent( new ContactEvent( ContactEvent.SAVE, contact ) );
            }

            private function remove():void
            {
                dispatchEvent( new ContactEvent( ContactEvent.DELETE, contact ) );
            }

            public function save_result( ):void
            {
                status.text = "Contact saved successfully";
            }

        ]]>
    </mx:Script>
```

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" label="{contact.id>0?contact.fullName:'New Contact'}">

    <mx:Script>
        <![CDATA[
            [Bindable]
            public var contact:Contact;

            private function save():void
            {
                contact.firstName = firstName.text;
                contact.lastName = lastName.text;
                contact.email = email.text;
                contact.phone = phone.text;
                contact.address = address.text;
                contact.city = city.text;
                contact.state = state.text;
                contact.zip = zip.text;
                contact.pic = picture.source;

                dispatchEvent( new ContactEvent( ContactEvent.SAVE, contact ) );
            }

            private function remove():void
            {
                dispatchEvent( new ContactEvent( ContactEvent.DELETE, contact ) );
            }

            public function save_result( ):void
            {
                status.text = "Contact saved successfully";
            }

        ]]>
    </mx:Script>
```

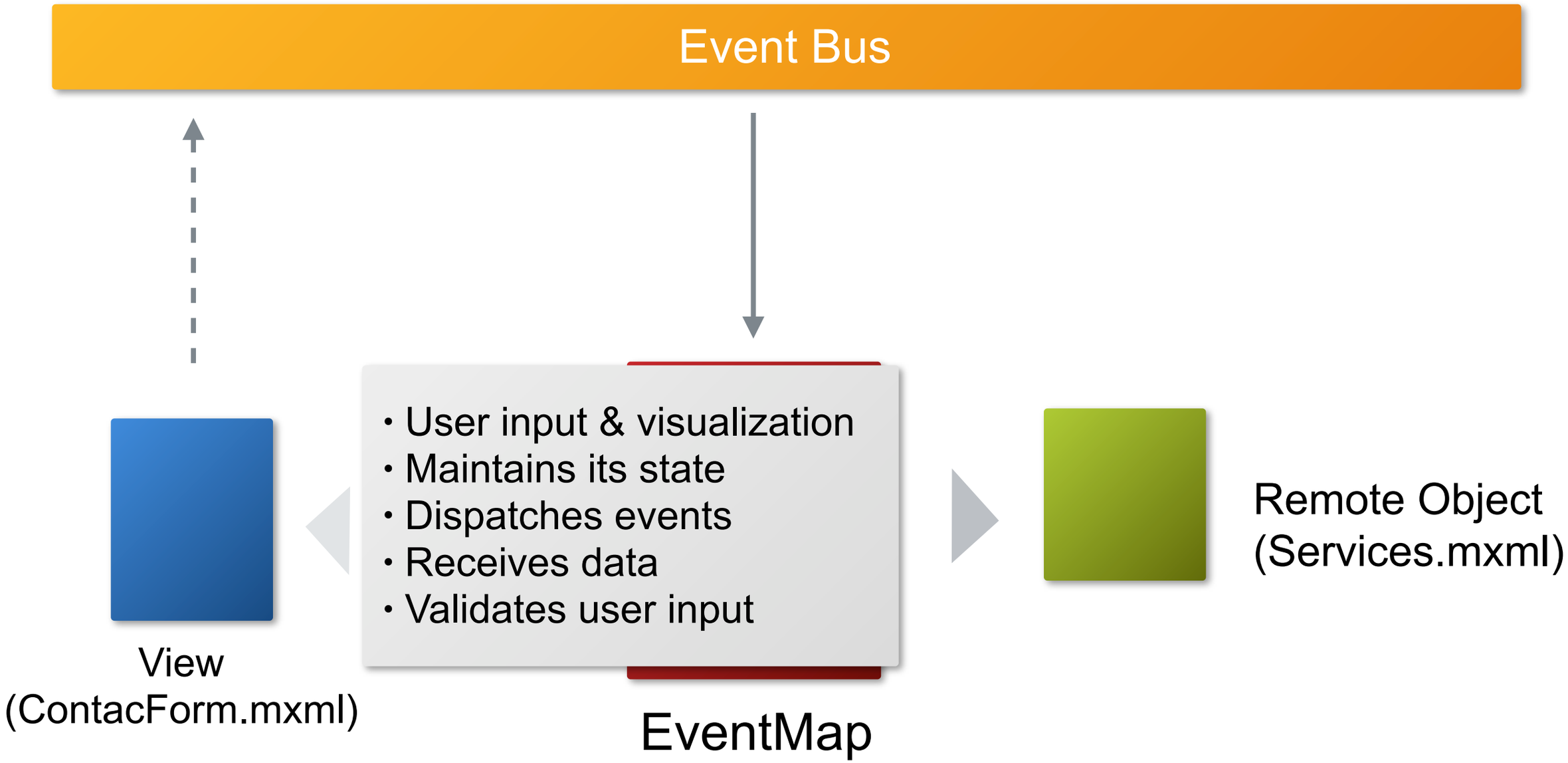
Saving a contact



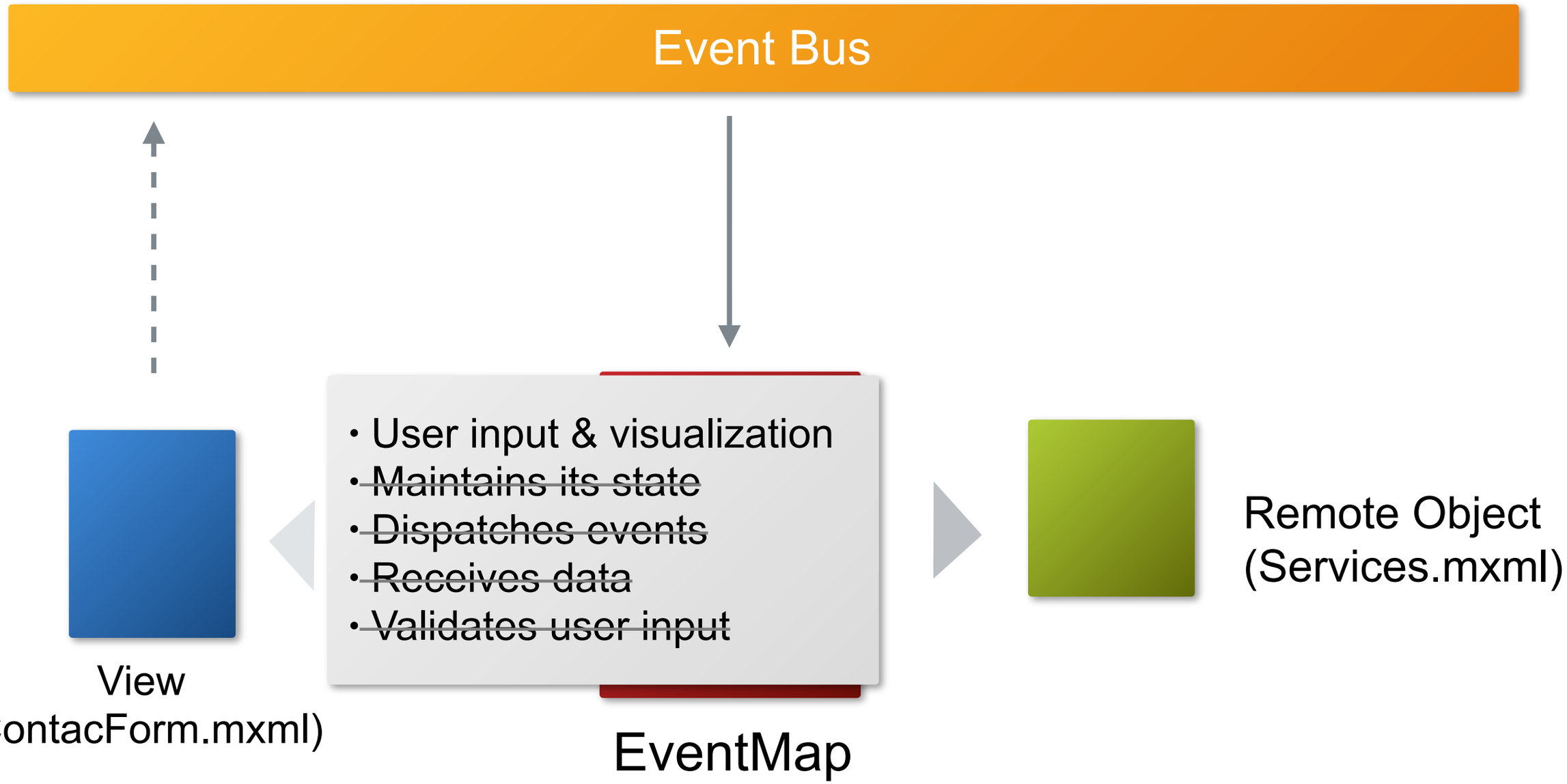
Version 3

Contact Form

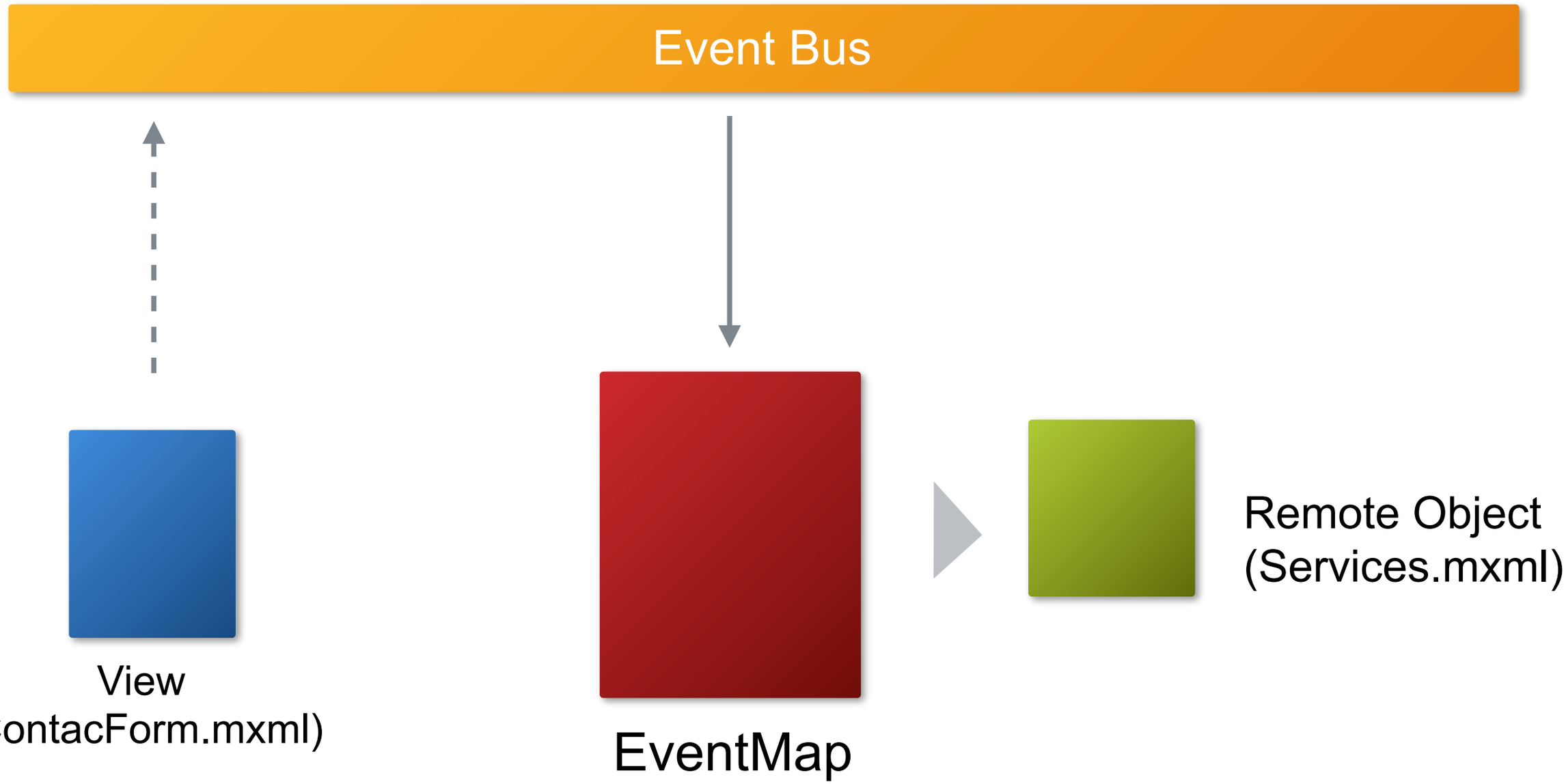
Presentation Model



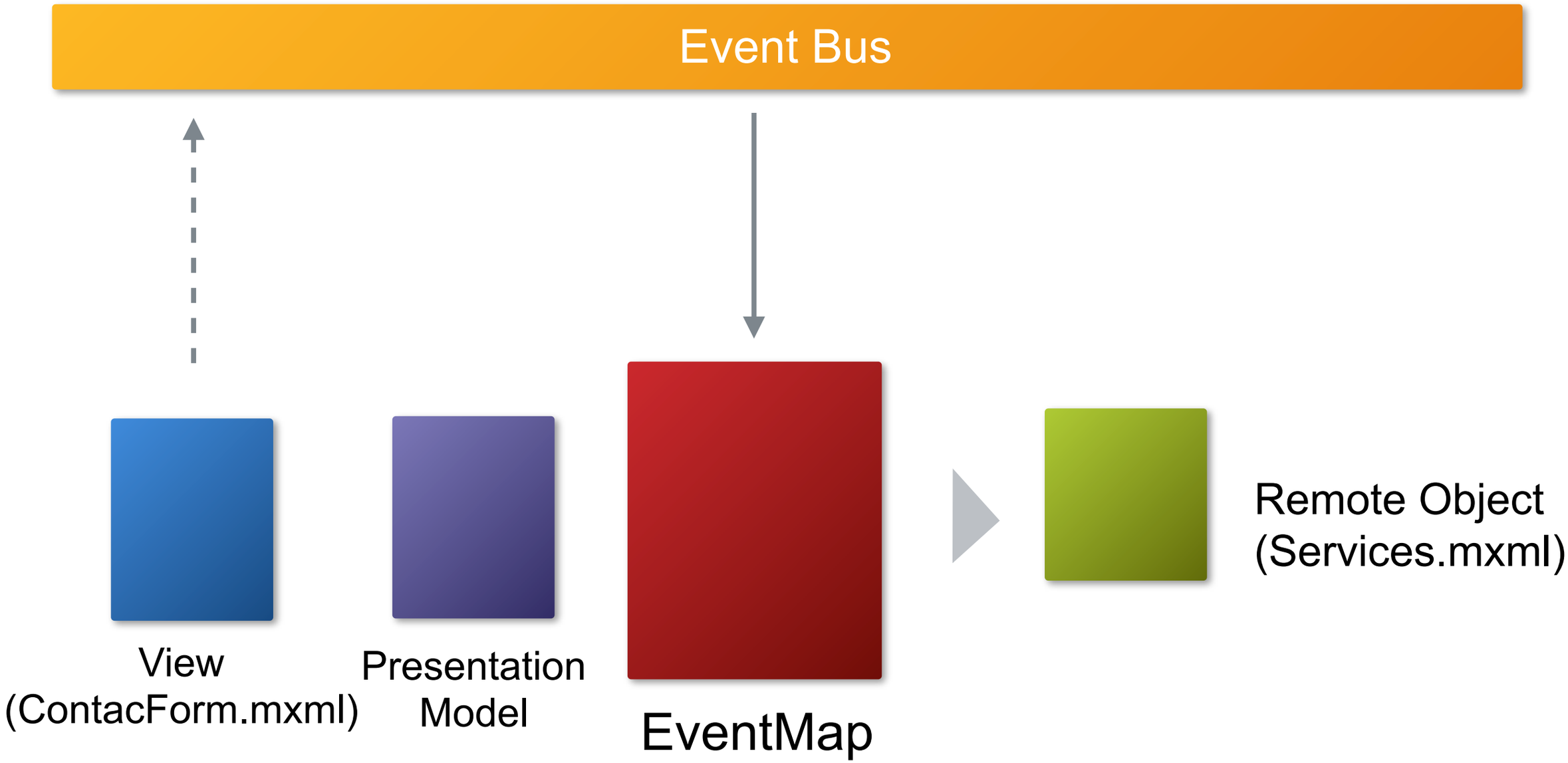
Presentation Model



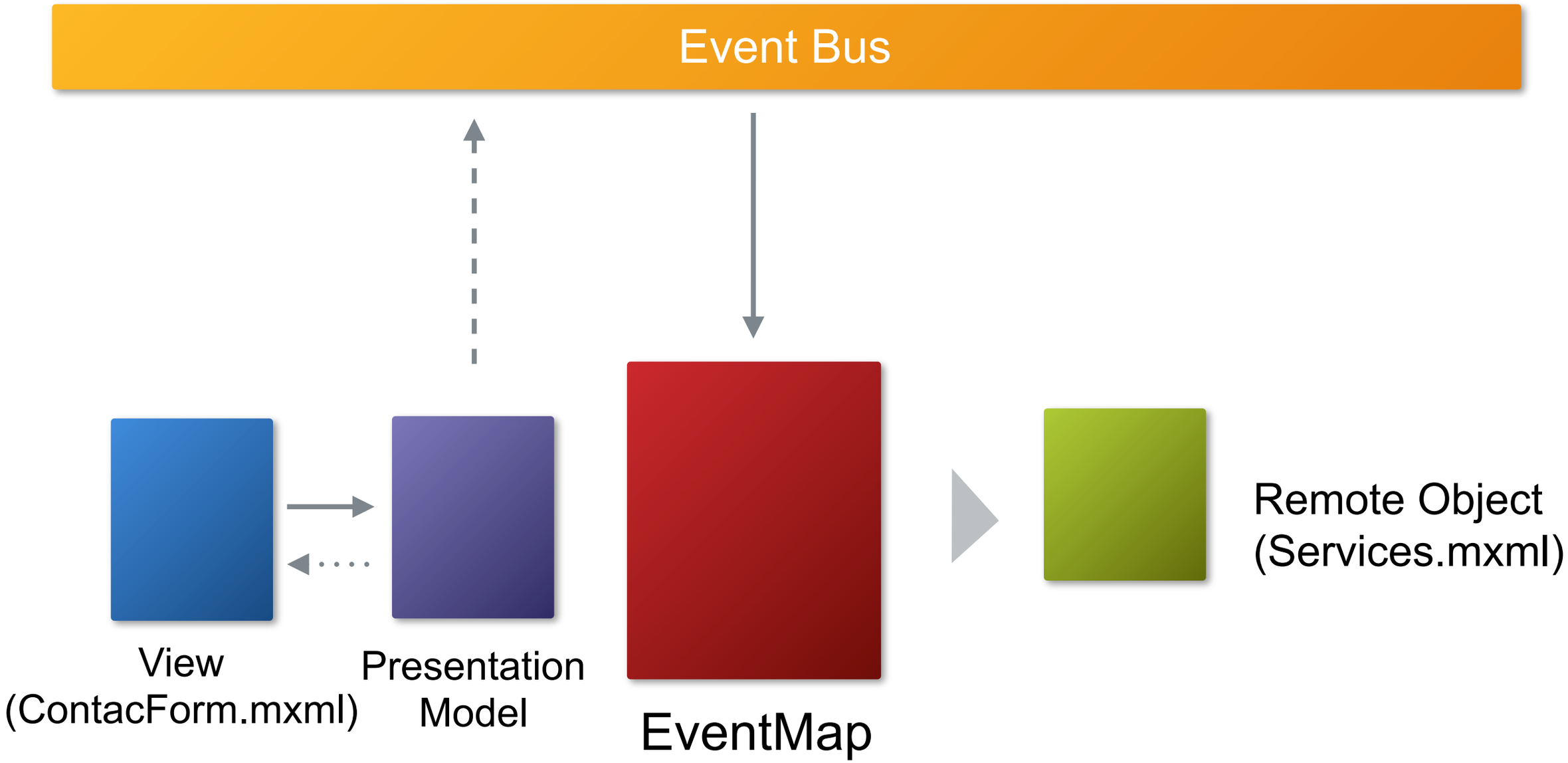
Presentation Model



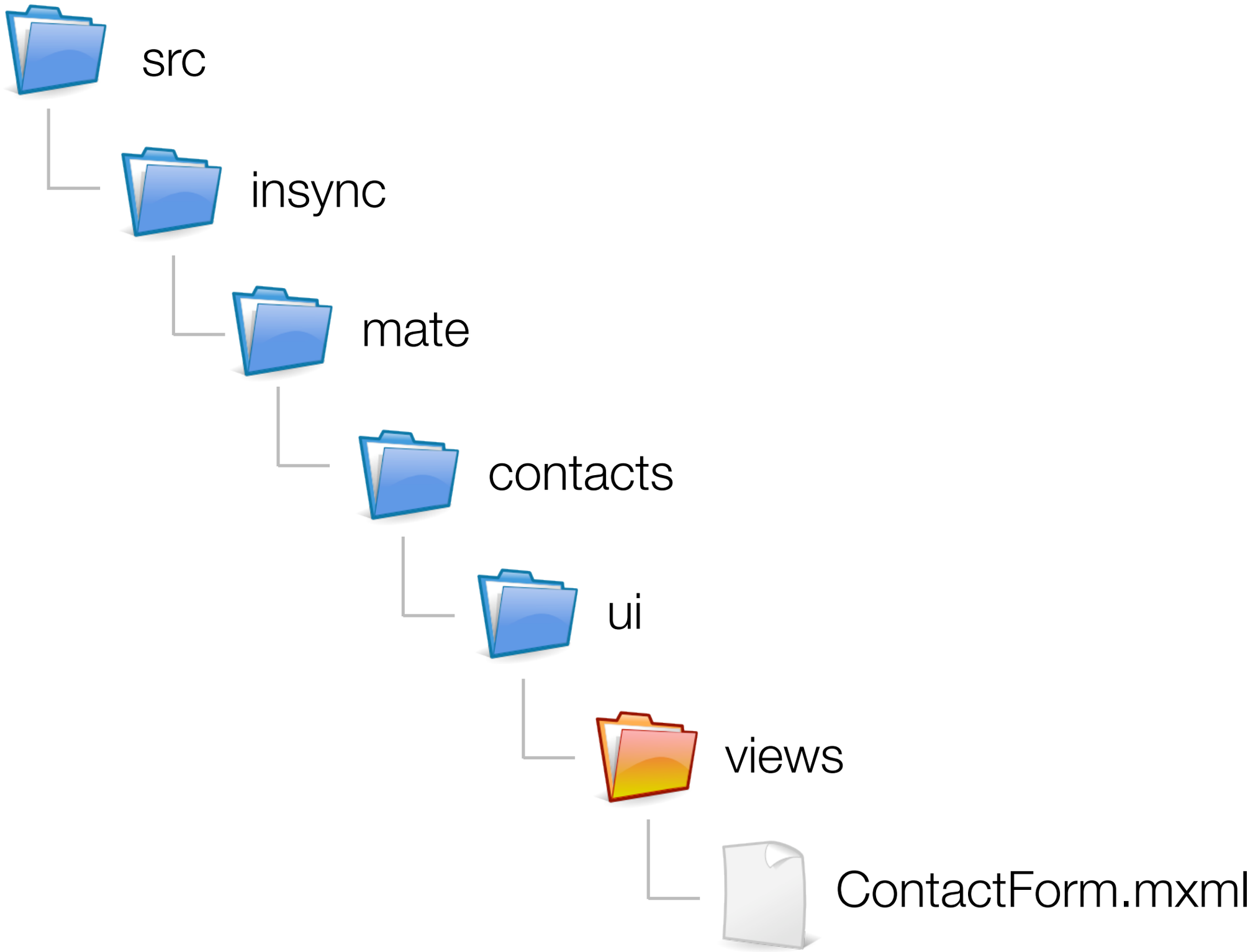
Presentation Model



Presentation Model



Open file (InsyncMate3):



```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml">

    <mx:Script>
            [Bindable]
            public var model:ContactFormPresentationModel;

    </mx:Script>

    <mx:Form>

        <mx:FormItem label="Id">
            <mx:TextInput text="{ model.contact.id }" enabled="false" />
        </mx:FormItem>

        <mx:FormItem label="First Name">
            <mx:TextInput id="firstName" text="{ model.contact.firstName }"
                change="model.updateFirstName( firstName.text )"/>
        </mx:FormItem>

        <-- all of the other fields here -->
    </mx:Form>

    <controls:PictureInput id="picture"
        source="{ model.contact.pic }"/>

    <mx:Button bottom="26" left="12" label="Save"   click="model.save()"/>
    <mx:Button bottom="26" left="72" label="Delete" click="model.remove()"/>

    <mx:Label id="status" text="{ model.status }" />

</mx:Canvas>
```

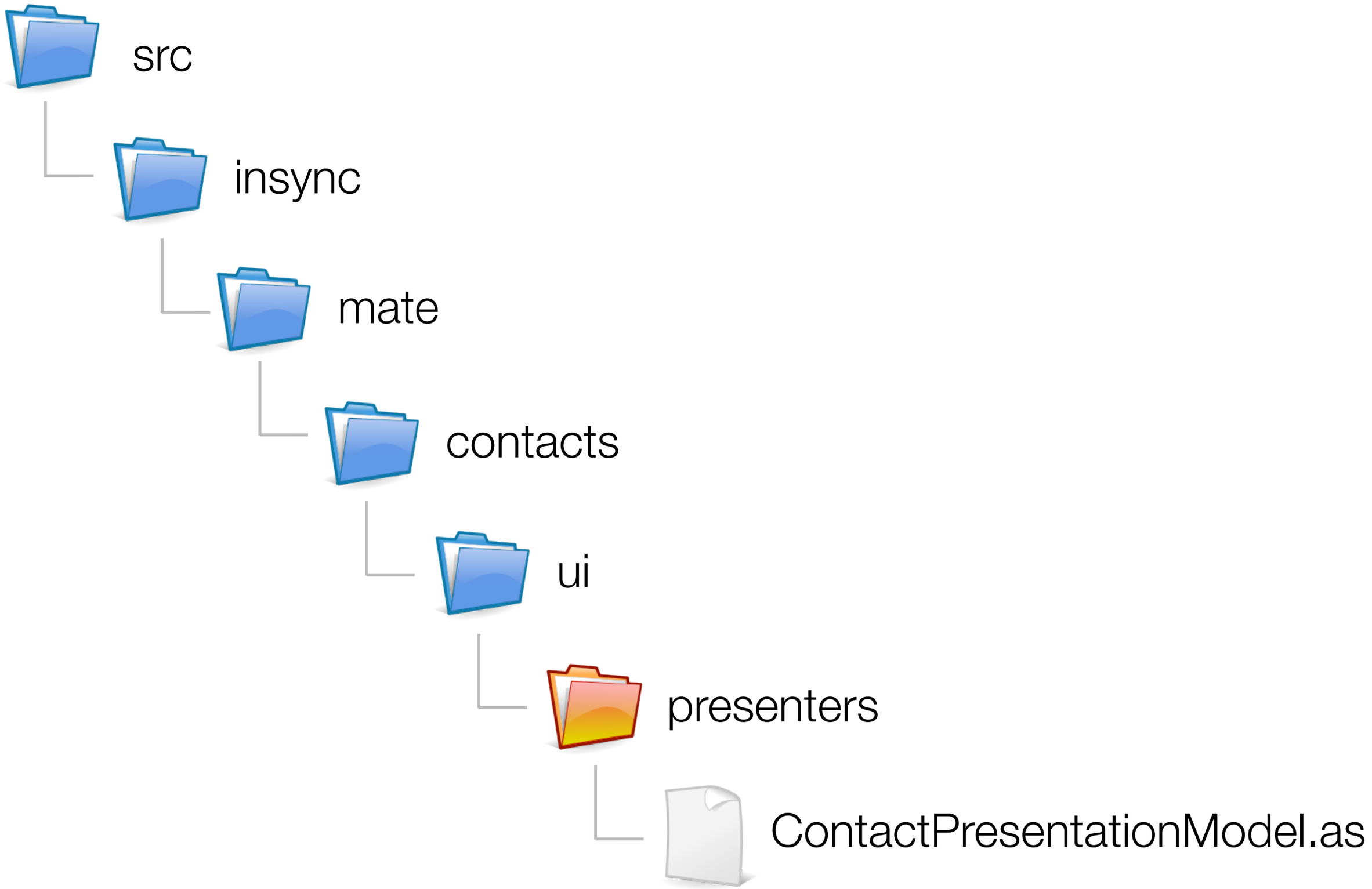
```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml">

    <mx:Script>
            [Bindable]
            public var model:ContactFormPresentationModel;

    </mx:Script>

    <mx:Form>

        <mx:FormItem label="Id">
            <mx:TextInput text="{ model.contact.id }" enabled="false" />
        </mx:FormItem>

        <mx:FormItem label="First Name">
            <mx:TextInput id="firstName" text="{ model.contact.firstName }"
                change="model.updateFirstName( firstName.text )"/>
        </mx:FormItem>

        <-- all of the other fields here -->
    </mx:Form>

    <controls:PictureInput id="picture"
        source="{ model.contact.pic }"/>

    <mx:Button bottom="26" left="12" label="Save"   click="model.save()"/>
    <mx:Button bottom="26" left="72" label="Delete" click="model.remove()"/>

    <mx:Label id="status" text="{ model.status }" />

</mx:Canvas>
```

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml">

    <mx:Script>
            [Bindable]
            public var model:ContactFormPresentationModel;

    </mx:Script>

    <mx:Form>

        <mx:FormItem label="Id">
            <mx:TextInput text="{ model.contact.id }" enabled="false" />
        </mx:FormItem>

        <mx:FormItem label="First Name">
            <mx:TextInput id="firstName" text="{ model.contact.firstName }"
                change="model.updateFirstName( firstName.text )"/>
        </mx:FormItem>

        <-- all of the other fields here -->
    </mx:Form>

    <controls:PictureInput id="picture"
        source="{ model.contact.pic }"/>

    <mx:Button bottom="26" left="12" label="Save"   click="model.save()"/>
    <mx:Button bottom="26" left="72" label="Delete" click="model.remove()"/>

    <mx:Label id="status" text="{ model.status }" />

</mx:Canvas>
```

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml">

    <mx:Script>
            [Bindable]
            public var model:ContactFormPresentationModel;

    </mx:Script>

    <mx:Form>

        <mx:FormItem label="Id">
            <mx:TextInput text="{ model.contact.id }" enabled="false" />
        </mx:FormItem>

        <mx:FormItem label="First Name">
            <mx:TextInput id="firstName" text="{ model.contact.firstName }"
                change="model.updateFirstName( firstName.text )"/>
        </mx:FormItem>

        <-- all of the other fields here -->
    </mx:Form>

    <controls:PictureInput id="picture"
        source="{ model.contact.pic }"/>

    <mx:Button bottom="26" left="12" label="Save"   click="model.save()"/>
    <mx:Button bottom="26" left="72" label="Delete" click="model.remove()"/>

    <mx:Label id="status" text="{ model.status }" />

</mx:Canvas>
```

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml">

    <mx:Script>
            [Bindable]
            public var model:ContactFormPresentationModel;

    </mx:Script>

    <mx:Form>

        <mx:FormItem label="Id">
            <mx:TextInput text="{ model.contact.id }" enabled="false" />
        </mx:FormItem>

        <mx:FormItem label="First Name">
            <mx:TextInput id="firstName" text="{ model.contact.firstName }"
                change="model.updateFirstName( firstName.text )"/>
        </mx:FormItem>

        <-- all of the other fields here -->
    </mx:Form>

    <controls:PictureInput id="picture"
        source="{ model.contact.pic }"/>

    <mx:Button bottom="26" left="12" label="Save"   click="model.save()"/>
    <mx:Button bottom="26" left="72" label="Delete" click="model.remove()"/>

    <mx:Label id="status" text="{ model.status }" />

</mx:Canvas>
```

Open file (InsyncMate3):



```
public class ContactFormPresentationModel extends EventDispatcher {

        // ------------------------------------
        private var _title:String;
        [Bindable(Event="titleChange")]
        public function get title():String
        {
            return _title;
        }

        // ------------------------------------
        private var _status:String;
        [Bindable(Event="statusChange")]
        public function get status():String
        {
            return _status;
        }

        // ------------------------------------
        private var _contact:Contact;
        [Bindable(Event="contactChange")]
        public function get contact():Contact
        {
            return _contact;
        }
```

```
public class ContactFormPresentationModel extends EventDispatcher {

        // ------------------------------------
        private var _title:String;
        [Bindable(Event="titleChange")]
        public function get title():String
        {
            return _title;
        }

        // ------------------------------------
        private var _status:String;
        [Bindable(Event="statusChange")]
        public function get status():String
        {
            return _status;
        }

        // ------------------------------------
        private var _contact:Contact;
        [Bindable(Event="contactChange")]
        public function get contact():Contact
        {
            return _contact;
        }
```

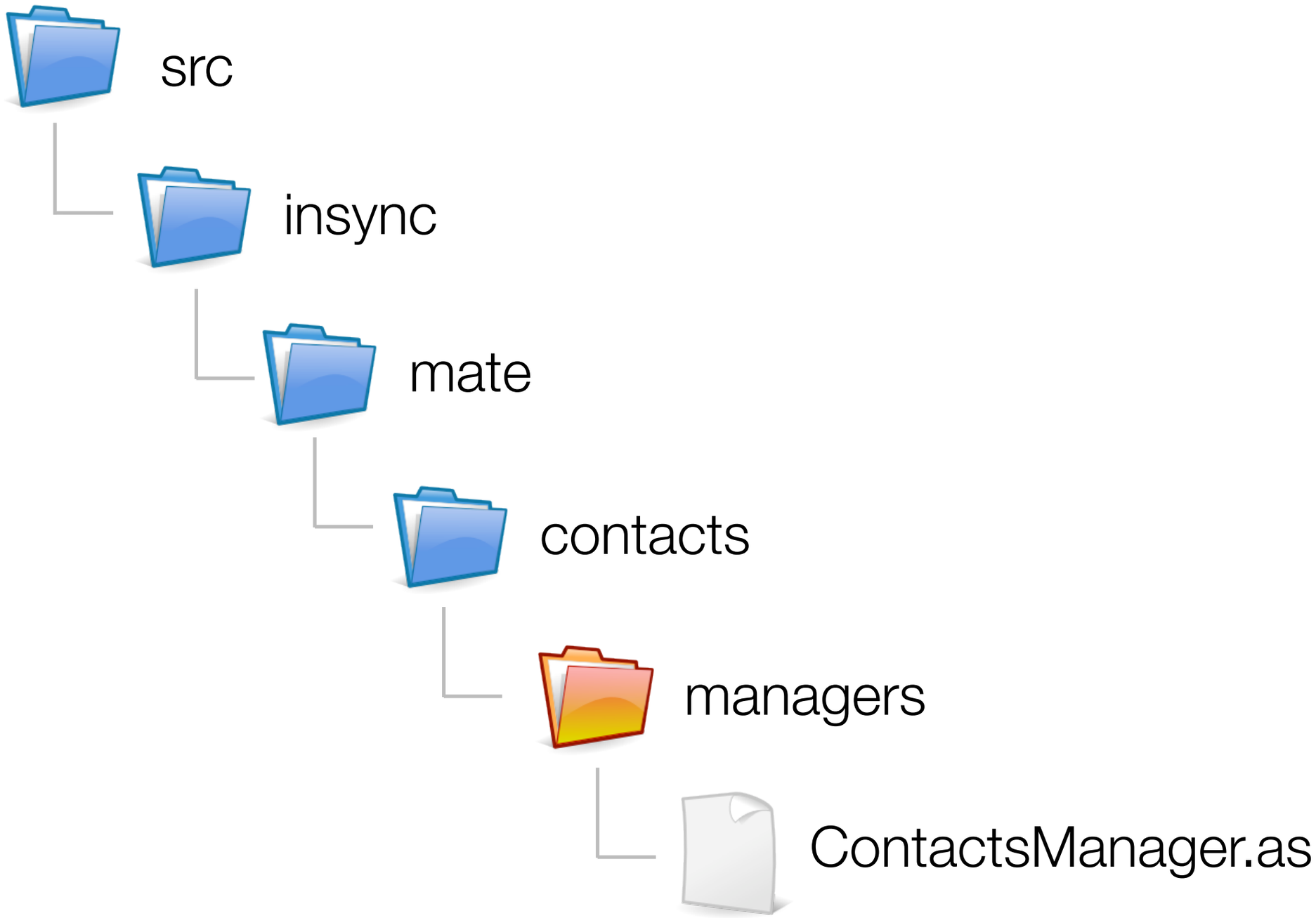
```
private var dispatcher:IEventDispatcher;
public function ContactFormPresentationModel(dispatcher:IEventDispatcher, contact:Contact)
{
    this.dispatcher = dispatcher;
    _contact = contact;
    _title = contact.id > 0 ? contact.fullName: 'New Contact';
}

// ------------------------------------
public function save():void
{
    dispatcher.dispatchEvent( new ContactEvent( ContactEvent.SAVE, contact ) );
}
// ------------------------------------
public function remove():void
{
    dispatcher.dispatchEvent( new ContactEvent( ContactEvent.DELETE, contact ) );
}
// ------------------------------------
public function contactSaved( event:ContactEvent ):void
{
    if( event.contact == contact )
    {
        _status = "Contact saved successfully";
        dispatchEvent( new Event( "statusChange" ) );
    }
```

```
//--------------------------------
  private var dispatcher:IEventDispatcher;
  public function ContactFormPresentationModel(dispatcher:IEventDispatcher, contact:Contact)
  {
      this.dispatcher = dispatcher;
      _contact = contact;
      _title = contact.id > 0 ? contact.fullName: 'New Contact';
  }

  // ------------------------------------
  public function save():void
  {
      dispatcher.dispatchEvent( new ContactEvent( ContactEvent.SAVE, contact ) );
  }
  // ------------------------------------
  public function remove():void
  {
      dispatcher.dispatchEvent( new ContactEvent( ContactEvent.DELETE, contact ) );
  }
  // ------------------------------------
  public function contactSaved( event:ContactEvent ):void
  {
      if( event.contact == contact )
      {
          _status = "Contact saved successfully";
          dispatchEvent( new Event( "statusChange" ) );
      }
```

```
//--------------------------------------
   private var dispatcher:IEventDispatcher;
   public function ContactFormPresentationModel(dispatcher:IEventDispatcher, contact:Contact)
   {
      this.dispatcher = dispatcher;
      _contact = contact;
      _title = contact.id > 0 ? contact.fullName: 'New Contact';
   }

   // --------------------------------------
   public function save():void
   {
      dispatcher.dispatchEvent( new ContactEvent( ContactEvent.SAVE, contact ) );
   }
   // --------------------------------------
   public function remove():void
   {
      dispatcher.dispatchEvent( new ContactEvent( ContactEvent.DELETE, contact ) );
   }
   // --------------------------------------
   public function contactSaved( event:ContactEvent ):void
   {
      if( event.contact == contact )
      {
         _status = "Contact saved successfully";
         dispatchEvent( new Event( "statusChange" ) );
      }
```

```
// Update functions ----------------------------------------
public function updateFirstName( firstName:String ):void
{
    contact.firstName = firstName;
}

// -------------------------------------
public function updateLastName( lastName:String  ):void
{
    contact.lastName = lastName;
}

// -------------------------------------
public function updateEmail( email:String ):void
{
    contact.email = email;
}

// -------------------------------------
public function updatePhone( phone:String ):void
{
    contact.phone = phone;
}
```

Open file (InsyncMate3):



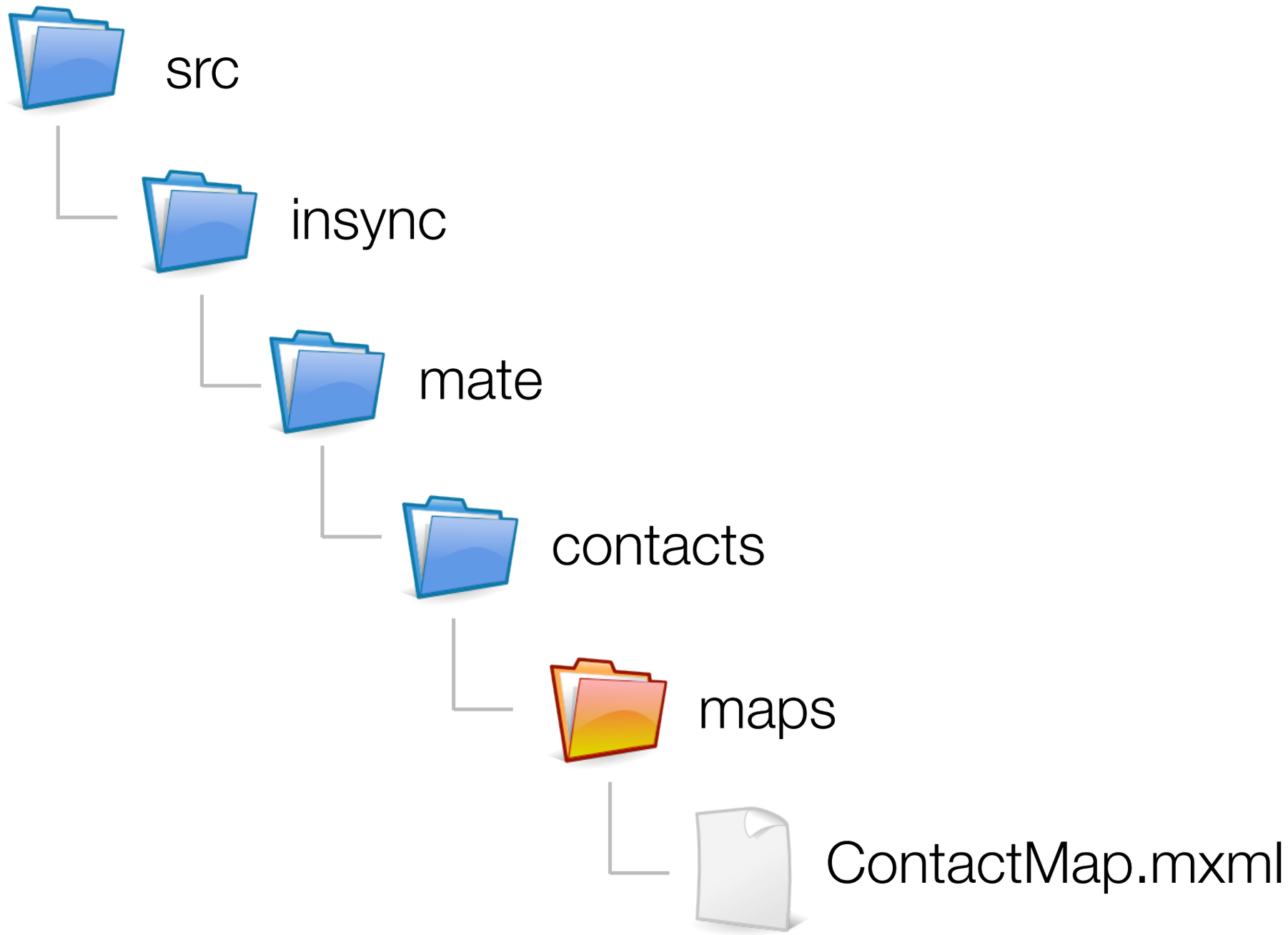
ContactsManager.as (InsyncMate3)

```
public class ContactsManager extends EventDispatcher
{
    private var editOpenContacts:Dictionary = new Dictionary( true );
    private var newOpenContacts:Dictionary = new Dictionary( true );

    // ------------------------------------------------------------------------
    // property to store the last search keyword
    public var lastSearch:String = "  ";

    // ------------------------------------------------------------------------
    // Read-only public variable used to access the current contacts to show in list
    private var _contacts:ArrayCollection;
    [Bindable(event="contactsChanged")]
    public function get contacts():ArrayCollection
    {
        return _contacts;
    }

    // ------------------------------------------------------------------------
    // Read-only public variable used to access the current contact (last selected contact)
    private var currentContact:Contact;
    public function getCurrentContact():Contact
    {
        return currentContact;
    }

    // ------------------------------------------------------------------------
    // Stores given contacts and the current search keyword
    public function storeSearch( list:ArrayCollection, searchKey:String  ):void
    {
        _contacts = list;
        dispatchEvent( new Event( "contactsChanged" ) );
```

ContactsManager.as (InsyncMate3)

```
public class ContactsManager extends EventDispatcher
{
    private var editOpenContacts:Dictionary = new Dictionary( true );
    private var newOpenContacts:Dictionary = new Dictionary( true );

    // ---------------------------------------------------------------------
    // property to store the last search keyword
    public var lastSearch:String = "  ";

    // ---------------------------------------------------------------------
    // Read-only public variable used to access the current contacts to show in list
    private var _contacts:ArrayCollection;
    [Bindable(event="contactsChanged")]
    public function get contacts():ArrayCollection
    {
        return _contacts;
    }

    // ---------------------------------------------------------------------
    // Read-only public variable used to access the current contact (last selected contact)
    private var currentContact:Contact;
    public function getCurrentContact():Contact
    {
        return currentContact;
    }

    // ---------------------------------------------------------------------
    // Stores given contacts and the current search keyword
    public function storeSearch( list:ArrayCollection, searchKey:String  ):void
    {
        _contacts = list;
        dispatchEvent( new Event( "contactsChanged" ) );
```

Open file (InsyncMate3):



ContactMap.mxml

```
<LocalEventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/">

<!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
   <Injectors target="{ ContactForm }">
      <MethodInvoker generator="{ ContactsManager }" method="getCurrentContact" />
      <ObjectBuilder generator="{ ContactFormPresentationModel }" cache="none"
         constructorArguments="{ [ scope.dispatcher, lastReturn ] }" />
      <PropertyInjector targetKey="model" source="{ lastReturn }"/>
   </Injectors>

   <Injectors target="{ ContactFormPresentationModel }">
      <ListenerInjector eventType="{ ContactEvent.SAVED }" method="contactSaved" />
   </Injectors>
</LocalEventMap>
```

ContactMap.mxml

```
<LocalEventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/">

<!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
   <Injectors target="{ ContactForm }">
      <MethodInvoker generator="{ ContactsManager }" method="getCurrentContact" />
      <ObjectBuilder generator="{ ContactFormPresentationModel }" cache="none"
         constructorArguments="{ [ scope.dispatcher, lastReturn ] }" />
      <PropertyInjector targetKey="model" source="{ lastReturn }"/>
   </Injectors>


   <Injectors target="{ ContactFormPresentationModel }">
      <ListenerInjector eventType="{ ContactEvent.SAVED }" method="contactSaved" />
   </Injectors>
</LocalEventMap>
```

ContactMap.mxml (InsyncMate3)

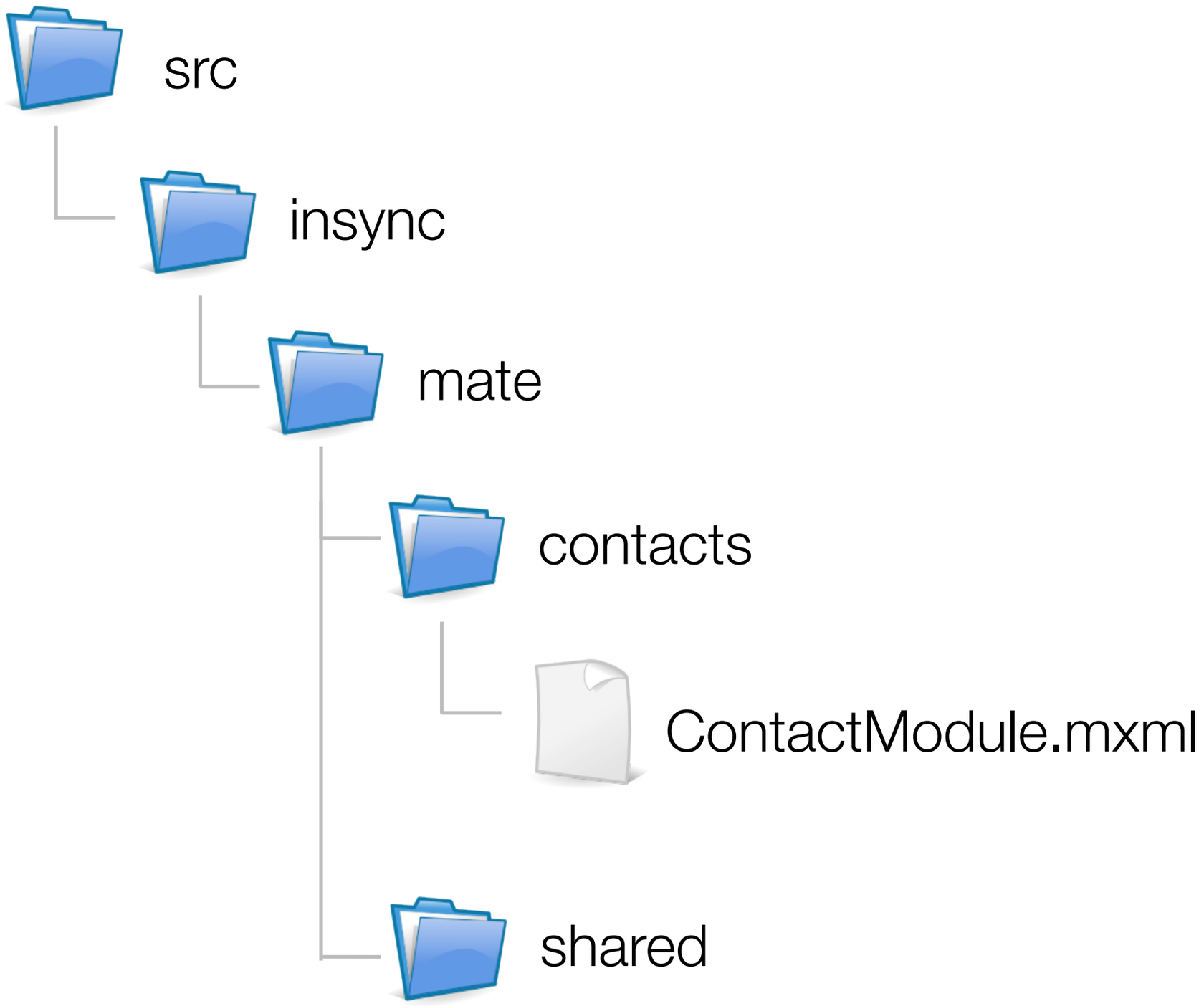
```
<!-- ContactEvent.SAVE  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <EventHandlers type="{ ContactEvent.SAVE }">

        <RemoteObjectInvoker instance="{ services.contacts }"  method="save"
            arguments="{ event.contact }">

            <resultHandlers>
                <MethodInvoker generator="{ ContactsManager }" method="save" arguments="{ [resultObject,
event.contact] }"/>
                <EventAnnouncer generator="{ SearchEvent }" dispatcherType="global"
                    type="{ ContactEvent.REFRESH }"/>

                <EventAnnouncer generator="{ ContactEvent }" constructorArguments="{ [ContactEvent.SAVED,
event.contact] }"/>

            </resultHandlers>

        </RemoteObjectInvoker>
    </EventHandlers>
```

MAX

Version 3

Modules

Open folder (InsyncMate3):



ContactModule.mxml

```
<mx:Module xmlns:mx="http://www.adobe.com/2006/mxml">

    <mx:Script>
            public function addTab( event:NavigatorEvent ):void
            {
                try
                {
                    var childIndex:int = tabNavigator.getChildIndex( event.content );
                    tabNavigator.selectedIndex = childIndex;
                }
                catch( e:Error)
                {
                    tabNavigator.addChild( event.content );
                    tabNavigator.selectedChild = event.content;
                }
            }

            public function removeTab( event:NavigatorEvent ):void
            {
                tabNavigator.removeChild( event.content );
            }
    </mx:Script>

    <!-- EventMaps  -->
    <maps:ContactMap  dispatcher="{ this }"/>

    <!-- UI  -->
    <mx:HDividedBox width="100%" height="100%">
        <flexLib:SuperTabNavigator id="tabNavigator" width="100%" height="100%"/>
        <views:ContactList id="list" width="300"/>
    </mx:HDividedBox>
```

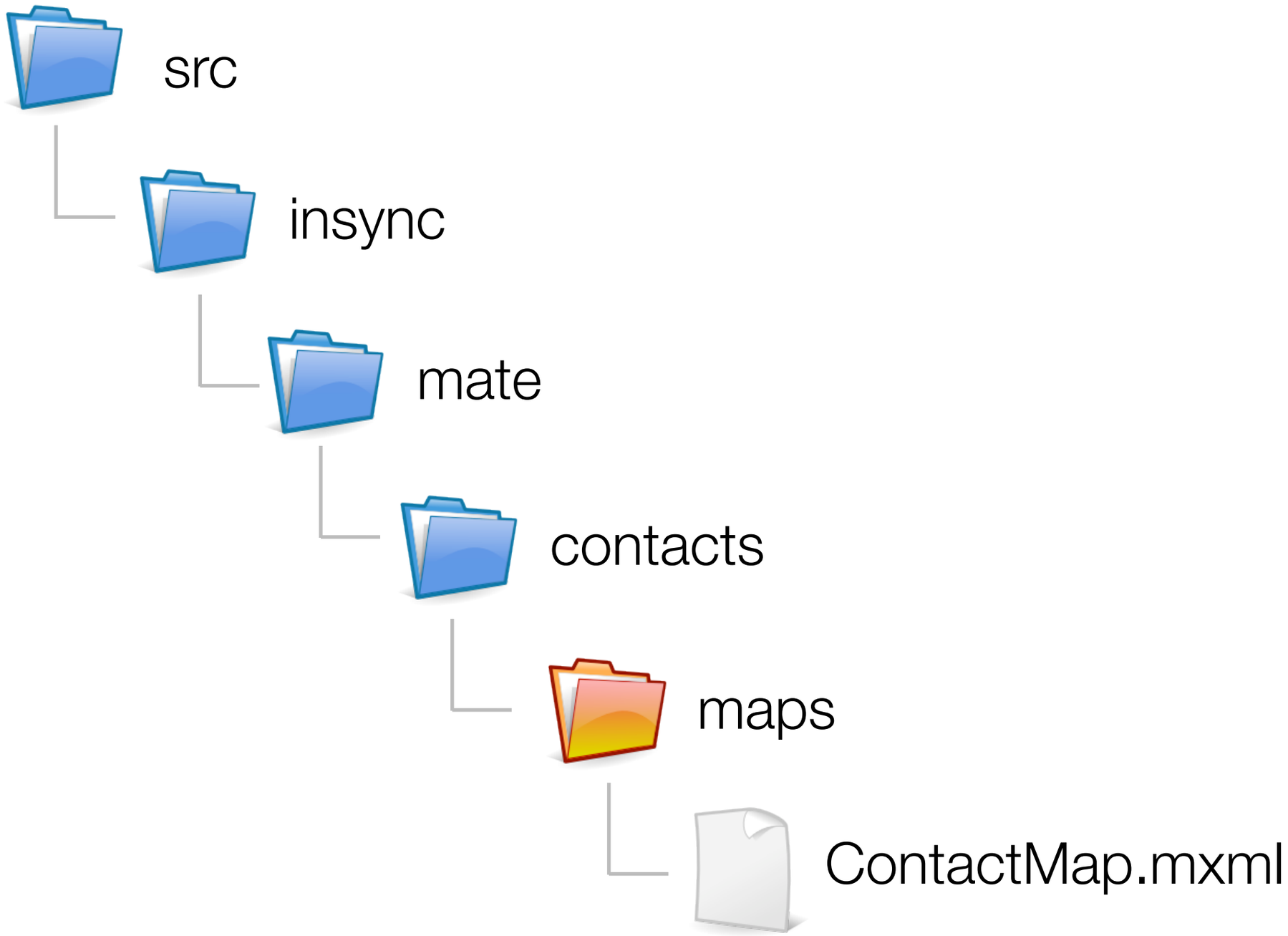
ContactModule.mxml

```
<mx:Module xmlns:mx="http://www.adobe.com/2006/mxml">

    <mx:Script>
            public function addTab( event:NavigatorEvent ):void
            {
                try
                {
                    var childIndex:int = tabNavigator.getChildIndex( event.content );
                    tabNavigator.selectedIndex = childIndex;
                }
                catch( e:Error)
                {
                    tabNavigator.addChild( event.content );
                    tabNavigator.selectedChild = event.content;
                }
            }

            public function removeTab( event:NavigatorEvent ):void
            {
                tabNavigator.removeChild( event.content );
            }
    </mx:Script>

    <!-- EventMaps  -->
    <maps:ContactMap  dispatcher="{ this }"/>

    <!-- UI  -->
    <mx:HDividedBox width="100%" height="100%">
        <flexLib:SuperTabNavigator id="tabNavigator" width="100%" height="100%"/>
        <views:ContactList id="list" width="300"/>
    </mx:HDividedBox>
```

ContactModule.mxml

```
<mx:Module xmlns:mx="http://www.adobe.com/2006/mxml">

    <mx:Script>
            public function addTab( event:NavigatorEvent ):void
            {
                try
                {
                    var childIndex:int = tabNavigator.getChildIndex( event.content );
                    tabNavigator.selectedIndex = childIndex;
                }
                catch( e:Error)
                {
                    tabNavigator.addChild( event.content );
                    tabNavigator.selectedChild = event.content;
                }
            }

            public function removeTab( event:NavigatorEvent ):void
            {
                tabNavigator.removeChild( event.content );
            }
    </mx:Script>

    <!-- EventMaps  -->
    <maps:ContactMap  dispatcher="{ this }"/>

    <!-- UI  -->
    <mx:HDividedBox width="100%" height="100%">
        <flexLib:SuperTabNavigator id="tabNavigator" width="100%" height="100%"/>
        <views:ContactList id="list" width="300"/>
    </mx:HDividedBox>
```

Open file (InsyncMate3):



ContactMap.mxml

```
<LocalEventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/">

<!-- SearchEvent.SEARCH  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <EventHandlers type="{ SearchEvent.SEARCH }" dispatcherType="global">
        <RemoteObjectInvoker instance="{ services.contacts }"
            method="getContactsByName"
            arguments="{ event.searchStr }">

            <resultHandlers>
                <MethodInvoker generator="{ ContactsManager }"
                    method="storeSearch"
                    arguments="{ [resultObject, event.searchStr] }"/>
            </resultHandlers>

        </RemoteObjectInvoker>
    </EventHandlers>

    <!-- ContactEvent.EDIT  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <EventHandlers type="{ ContactEvent.EDIT }">
        <MethodInvoker generator="{ ContactsManager }" method="openEditContact"
arguments="{[ event.contact,  ContactForm ]}" />
        <EventAnnouncer generator="{ NavigatorEvent }" type="{ NavigatorEvent.ADD_CONTENT }" >
            <Property targetKey="content" source="{ lastReturn }"/>
        </EventAnnouncer>
    </EventHandlers>


    <!-- ContactEvent.ADD  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <EventHandlers type="{ ContactEvent.ADD }" dispatcherType="global">
        <MethodInvoker generator="{ ContactsManager }" method="openNewContact"
arguments="{ ContactForm }"/>
        <EventAnnouncer generator="{ NavigatorEvent }" type="{ NavigatorEvent.ADD_CONTENT }" >
```

ContactMap.mxml

```
<LocalEventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/">

<!-- SearchEvent.SEARCH  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <EventHandlers type="{ SearchEvent.SEARCH }" dispatcherType="global">
        <RemoteObjectInvoker instance="{ services.contacts }"
            method="getContactsByName"
            arguments="{ event.searchStr }">

            <resultHandlers>
                <MethodInvoker generator="{ ContactsManager }"
                    method="storeSearch"
                    arguments="{ [resultObject, event.searchStr] }"/>
            </resultHandlers>

        </RemoteObjectInvoker>
    </EventHandlers>

    <!-- ContactEvent.EDIT  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <EventHandlers type="{ ContactEvent.EDIT }">
        <MethodInvoker generator="{ ContactsManager }" method="openEditContact"
arguments="{[ event.contact,  ContactForm ]}" />
        <EventAnnouncer generator="{ NavigatorEvent }" type="{ NavigatorEvent.ADD_CONTENT }" >
            <Property targetKey="content" source="{ lastReturn }"/>
        </EventAnnouncer>
    </EventHandlers>


    <!-- ContactEvent.ADD  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <EventHandlers type="{ ContactEvent.ADD }" dispatcherType="global">
        <MethodInvoker generator="{ ContactsManager }" method="openNewContact"
arguments="{ ContactForm }"/>
        <EventAnnouncer generator="{ NavigatorEvent }" type="{ NavigatorEvent.ADD_CONTENT }" >
```

ContactMap.mxml

```
<LocalEventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/">

<!-- SearchEvent.SEARCH ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
    <EventHandlers type="{ SearchEvent.SEARCH }" dispatcherType="global">
        <RemoteObjectInvoker instance="{ services.contacts }"
            method="getContactsByName"
            arguments="{ event.searchStr }">

            <resultHandlers>
                <MethodInvoker generator="{ ContactsManager }"
                    method="storeSearch"
                    arguments="{ [resultObject, event.searchStr] }"/>
            </resultHandlers>

        </RemoteObjectInvoker>
    </EventHandlers>

    <!-- ContactEvent.EDIT ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
    <EventHandlers type="{ ContactEvent.EDIT }">
        <MethodInvoker generator="{ ContactsManager }" method="openEditContact"
arguments="{[ event.contact,  ContactForm ]}" />
        <EventAnnouncer generator="{ NavigatorEvent }" type="{ NavigatorEvent.ADD_CONTENT }" >
            <Property targetKey="content" source="{ lastReturn }"/>
        </EventAnnouncer>
    </EventHandlers>


    <!-- ContactEvent.ADD ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
    <EventHandlers type="{ ContactEvent.ADD }" dispatcherType="global">
        <MethodInvoker generator="{ ContactsManager }" method="openNewContact"
arguments="{ ContactForm }"/>
        <EventAnnouncer generator="{ NavigatorEvent }" type="{ NavigatorEvent.ADD_CONTENT }" >
```

ContactMap.mxml

```
<LocalEventMap xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="http://mate.asfusion.com/">

<!-- SearchEvent.SEARCH  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <EventHandlers type="{ SearchEvent.SEARCH }" dispatcherType="global">
        <RemoteObjectInvoker instance="{ services.contacts }"
            method="getContactsByName"
            arguments="{ event.searchStr }">

            <resultHandlers>
                <MethodInvoker generator="{ ContactsManager }"
                    method="storeSearch"
                    arguments="{ [resultObject, event.searchStr] }"/>
            </resultHandlers>

        </RemoteObjectInvoker>
    </EventHandlers>

    <!-- ContactEvent.EDIT  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <EventHandlers type="{ ContactEvent.EDIT }">
        <MethodInvoker generator="{ ContactsManager }" method="openEditContact"
arguments="{[ event.contact,  ContactForm ]}" />
        <EventAnnouncer generator="{ NavigatorEvent }" type="{ NavigatorEvent.ADD_CONTENT }" >
            <Property targetKey="content" source="{ lastReturn }"/>
        </EventAnnouncer>
    </EventHandlers>

    <!-- ContactEvent.ADD  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~  -->
    <EventHandlers type="{ ContactEvent.ADD }" dispatcherType="global">
        <MethodInvoker generator="{ ContactsManager }" method="openNewContact"
arguments="{ ContactForm }"/>
        <EventAnnouncer generator="{ NavigatorEvent }" type="{ NavigatorEvent.ADD_CONTENT }" >
```

Learn more

http://mate.asfusion.com