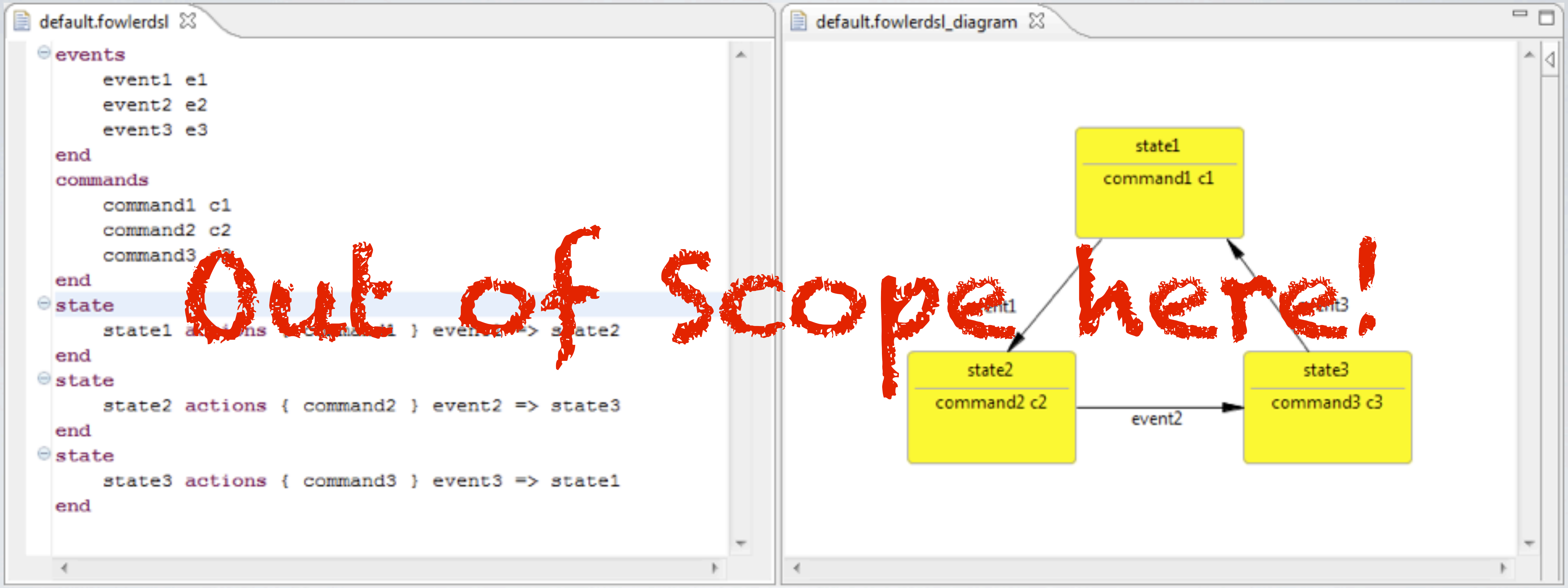
TMF meets GMF
Combining Graphical & Textual Modeling

Alexander Nyßen
itemis AG

Simultaneous/Parallel Editing

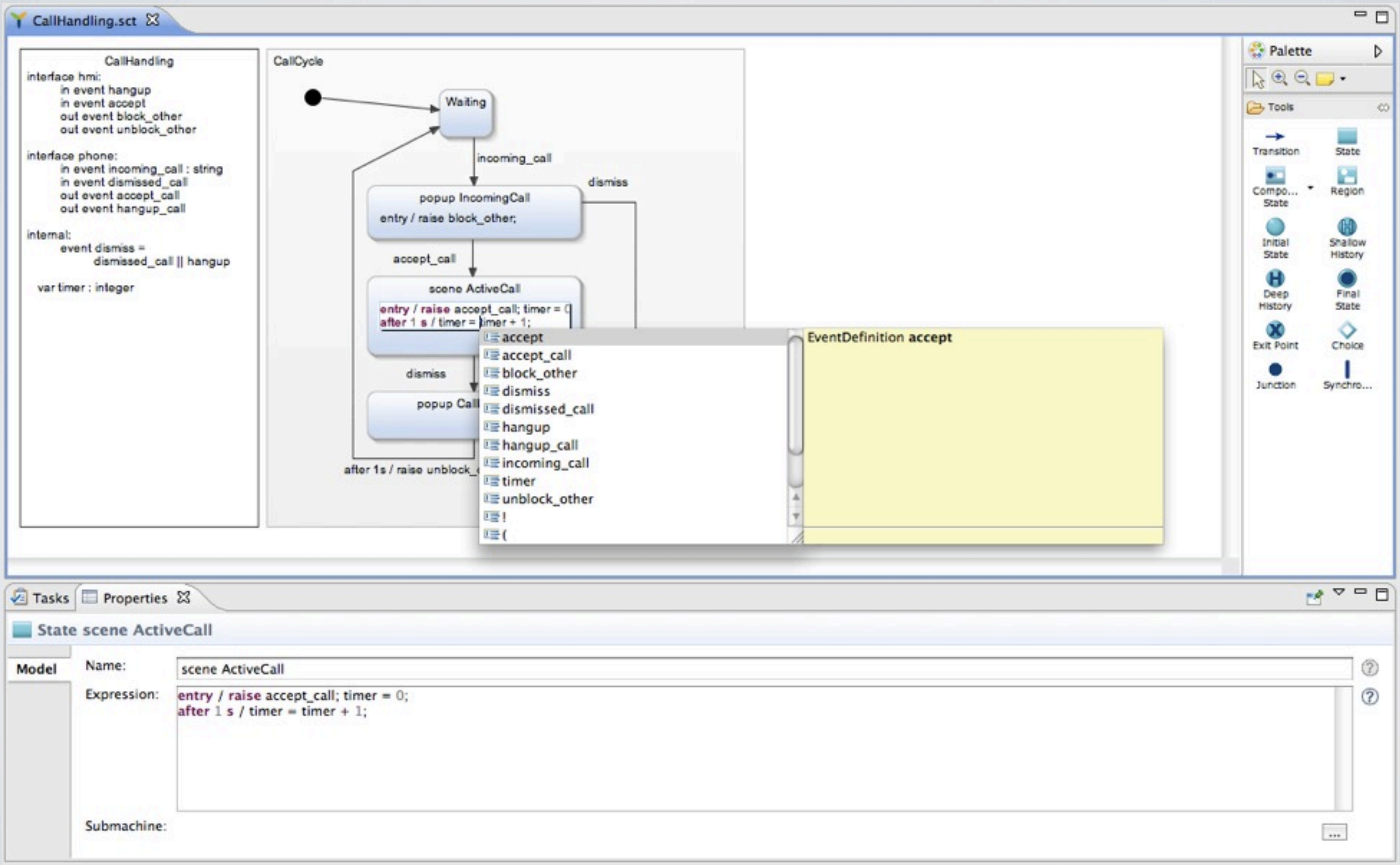


```
events
    event1 e1
    event2 e2
    event3 e3
end
commands
    command1 c1
    command2 c2
    command3 c3
end
state
    state1 actions { command1 } event1 => state2
end
state
    state2 actions { command2 } event2 => state3
end
state
    state3 actions { command3 } event3 => state1
end
```

Out of Scope here!

- http://www.eclipse.org/Xtext/documentation/2_0_0/210-emf-integration.php
- A. Mülder, A. Nyßen: TMF meets GMF. Eclipse Magazin 03/2011 (German)

Editing Embedded-Xtext



CallHandling

```
interface hmi:
        in event hangup
        in event accept
        out event block_other
        out event unblock_other

interface phone:
        in event incoming_call : string
        in event dismissed_call
        out event accept_call
        out event hangup_call

internal:
        event dismiss =
                dismissed_call || hangup

  var timer : integer
```

```
entry / raise accept_call; timer = 0
after 1 s / timer = timer + 1;
```

- accept
- accept_call
- block_other
- dismiss
- dismissed_call
- hangup
- hangup_call
- incoming_call
- timer
- unblock_other
- !
- (

EventDefinition **accept**

State scene ActiveCall

Model

Name: scene ActiveCall

Expression:

```
entry / raise accept_call; timer = 0;
after 1 s / timer = timer + 1;
```

Submachine:

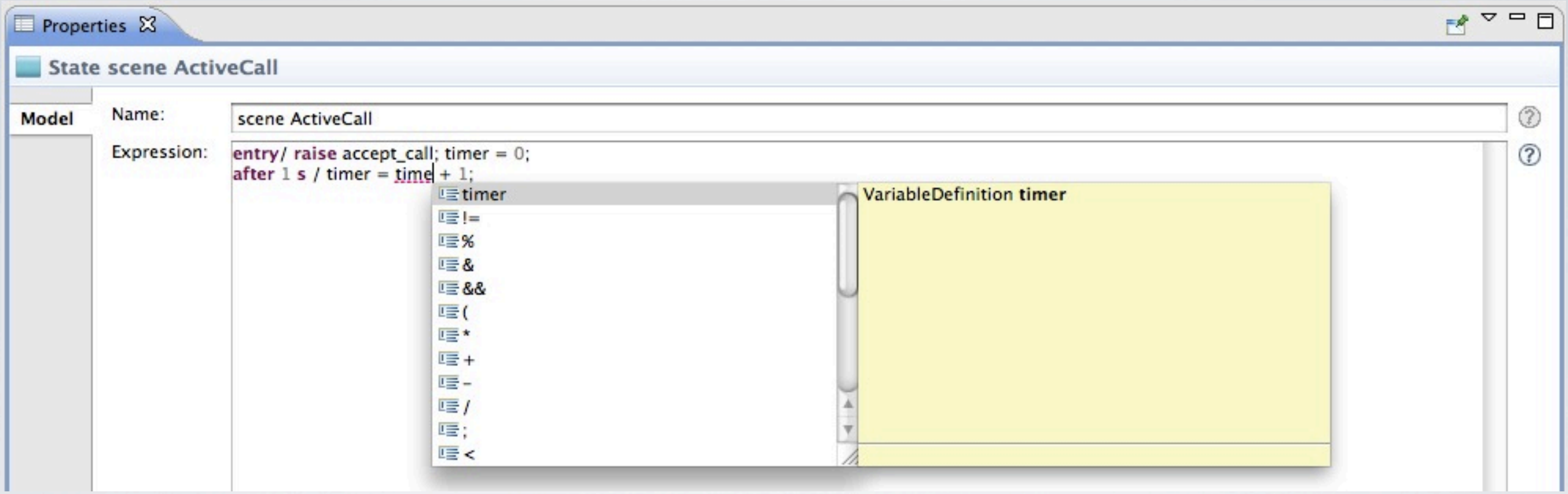
Editing Embedded-Xtext

Enable editing of Xtext-strings, providing all the nice Xtext-features like syntax coloring, content assist, and validation, outside an Xtext-editor within the following contexts:

- SWT/JFace: enable editing of Xtext-strings e.g. within WizardPages or PropertySheets.
- GEF/GMF (and potentially Graphiti): enable „direct-editing“ of Xtext-strings from within graphical editors.

So, let‘s start with looking at SWT/JFace!

Xtext-JFace-Integration



scene ActiveCall

```
entry/ raise accept_call; timer = 0;
after 1 s / timer = time + 1;
```

- `StyledTextXtextAdapter` to adapt Xtext-editing functionality to any StyledText (SWT control)
- `XtextStyledTextCellEditor` to enable Xtext-editing within arbitrary JFace Viewers (using StyledText and XtextAdapter)

Adapter & CellEditor Usage

- `StyledTextXtextAdapter` can easily be „hooked" to any StyledText

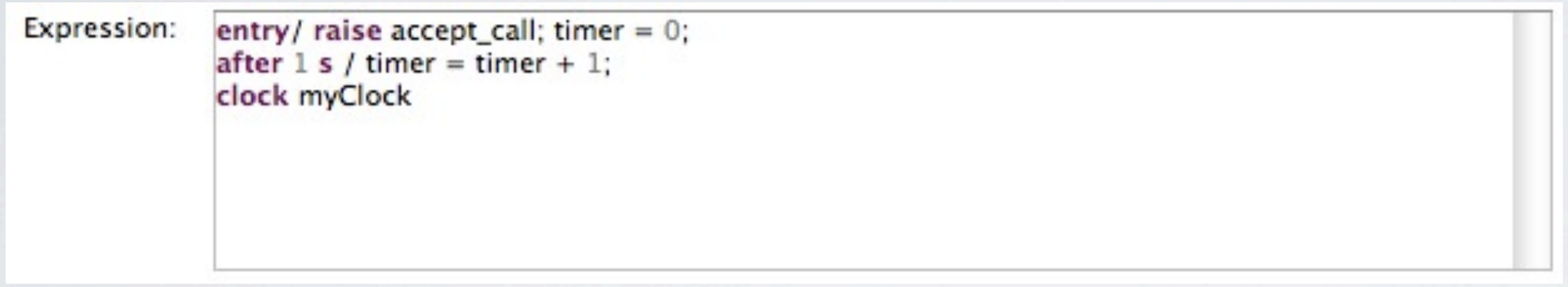
```
StyledText styledText = new StyledText(parent, style);
xtextAdapter = new StyledTextXtextAdapter(getInjector());
xtextAdapter.adapt(styledText);
```

- `XtextStyledTextCellEditor` can be used transparently as any JFace CellEditor (it adapts internally)

```
xtextCellEditor = new XtextStyledTextCellEditor(style,
                  getInjector());
xtextCellEditor.create((Composite)viewer.getControl());
```

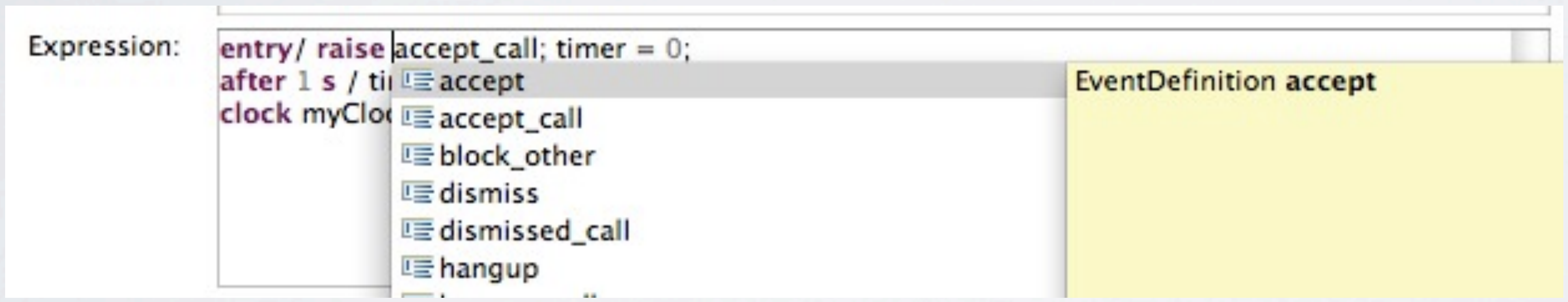
Xtext-JFace-Integration

- Syntax Highlighting: ✓



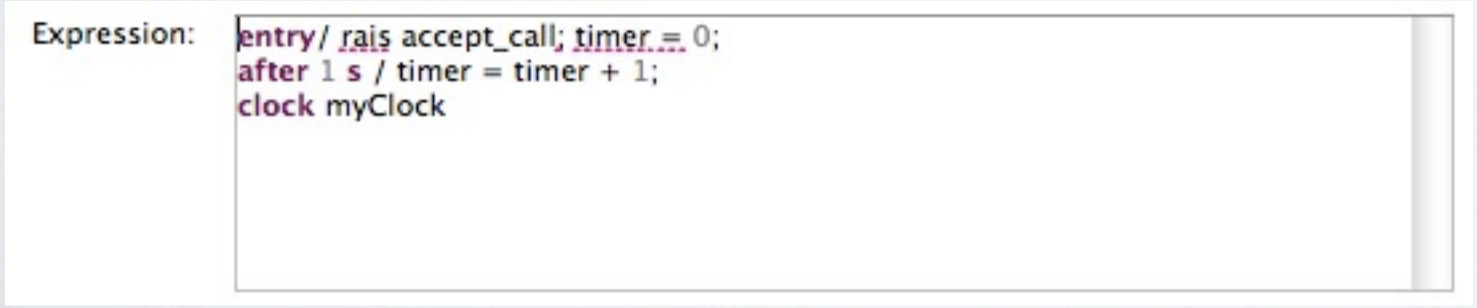
```
Expression:   entry/ raise accept_call; timer = 0;
              after 1 s / timer = timer + 1;
              clock myClock
```

- Auto Completion: ✓



```
Expression:   entry/ raise accept_call; timer = 0;
              after 1 s / ti
              clock myCloc
```

- Validation: ✓

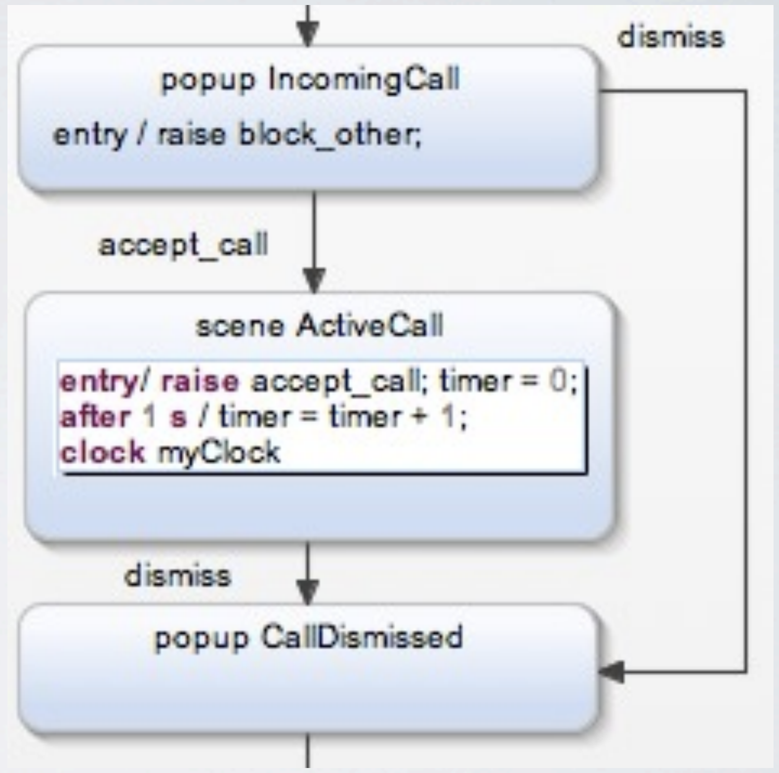


```
Expression:   entry/ rais accept_call; timer = 0;
              after 1 s / timer = timer + 1;
              clock myClock
```

And how to integrate Xtext with GEF/GMF?

Direct-Editing Embedded-Xtext

- GEF-integration:
  - `XtextDirectEditManager`, internally making use of the `Xtext-StyledTextCellEditor`
- GMF-integration:
  - `XtextLabelEditPart` (CompartmentEditPart)
  - `ExternalXtextLabelEditPart` (LabelEditPart)

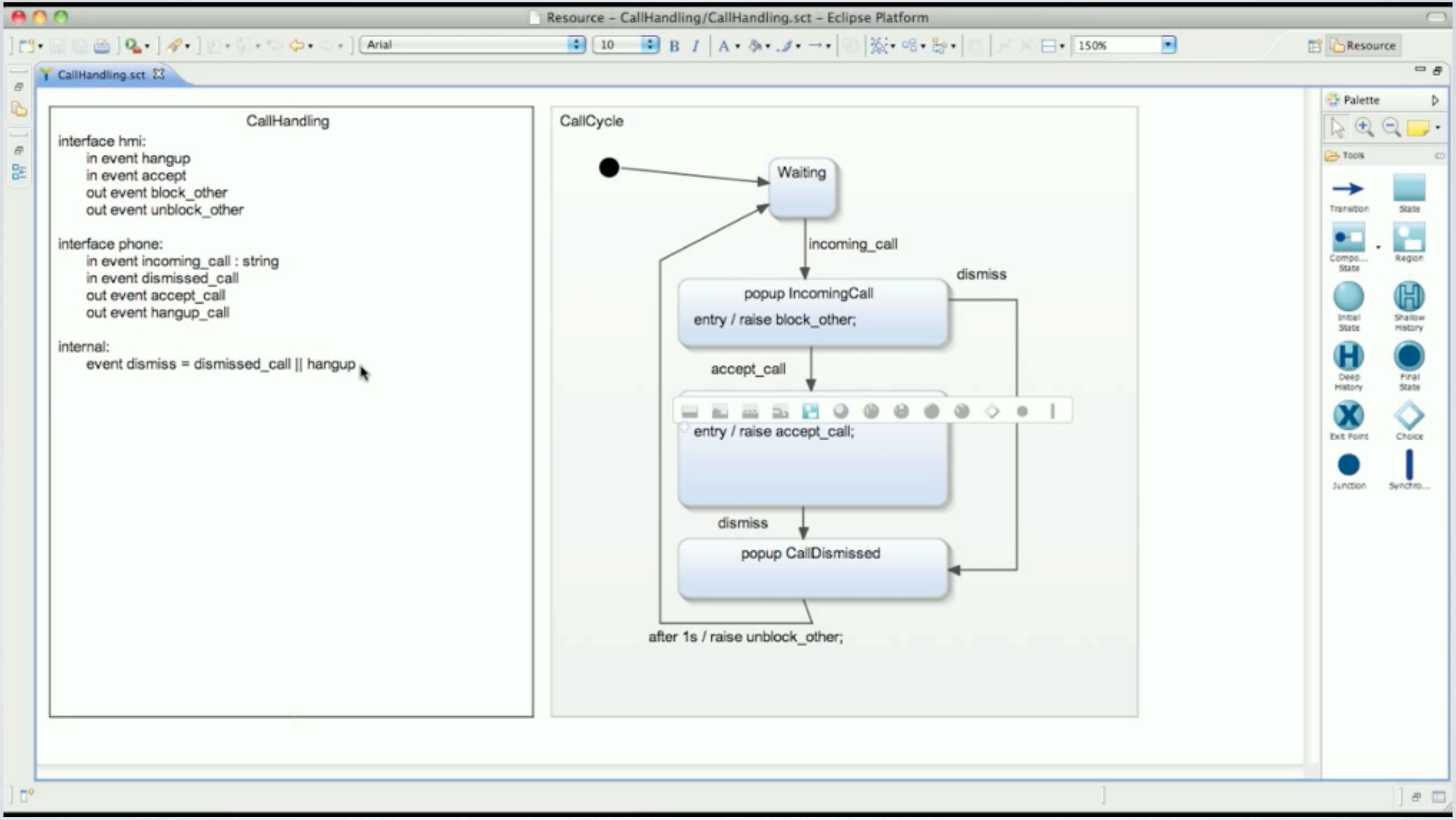

dismiss
popup IncomingCall
entry / raise block_other;
accept_call
scene ActiveCall
entry/ raise accept_call; timer = 0;
after 1 s / timer = timer + 1;
clock myClock
dismiss
popup CallDismissed

DirectEditManager Usage

- `XtextDirectEditManager` can be transparently used as any GEF DirectEditManager:

```
protected void performDirectEditRequest(final Request request) {
  final XtextDirectEditManager manager =
    new XtextDirectEditManager(this, getInjector(), getEditorStyles());
  try {
    getEditingDomain().runExclusive(new Runnable() {
      public void run() {
        ...
        manager.show();
        ...
      }
    });
  } catch (final InterruptedException e) {...}
}
```

Editing Embedded-Xtext

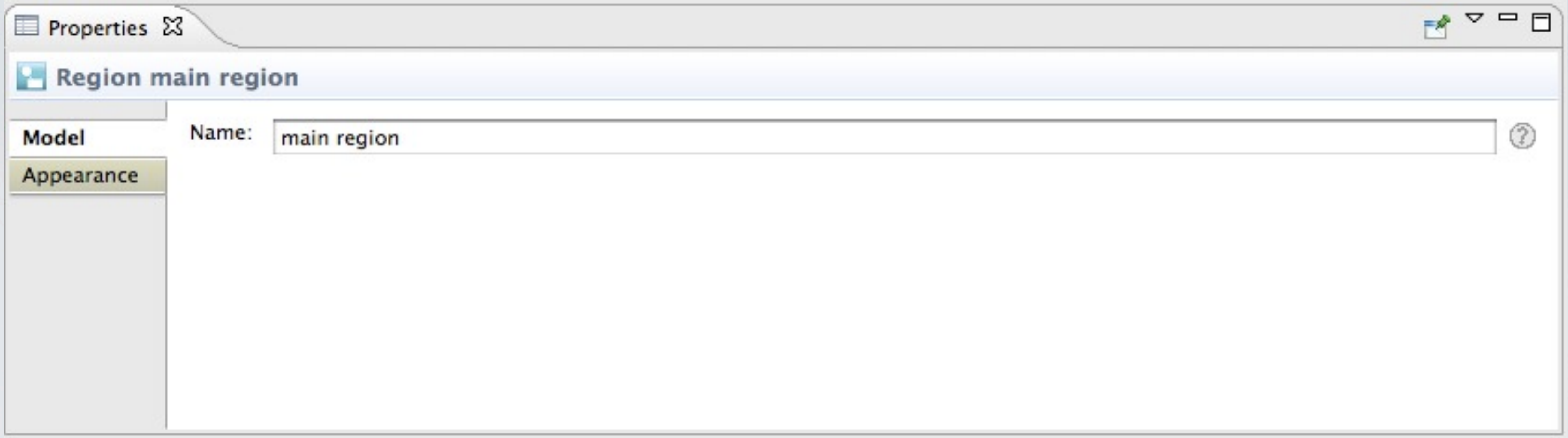


Resource – CallHandling/CallHandling.sct – Eclipse Platform

```
CallHandling
interface hmi:
    in event hangup
    in event accept
    out event block_other
    out event unblock_other

interface phone:
    in event incoming_call : string
    in event dismissed_call
    out event accept_call
    out event hangup_call

internal:
    event dismiss = dismissed_call || hangup
```

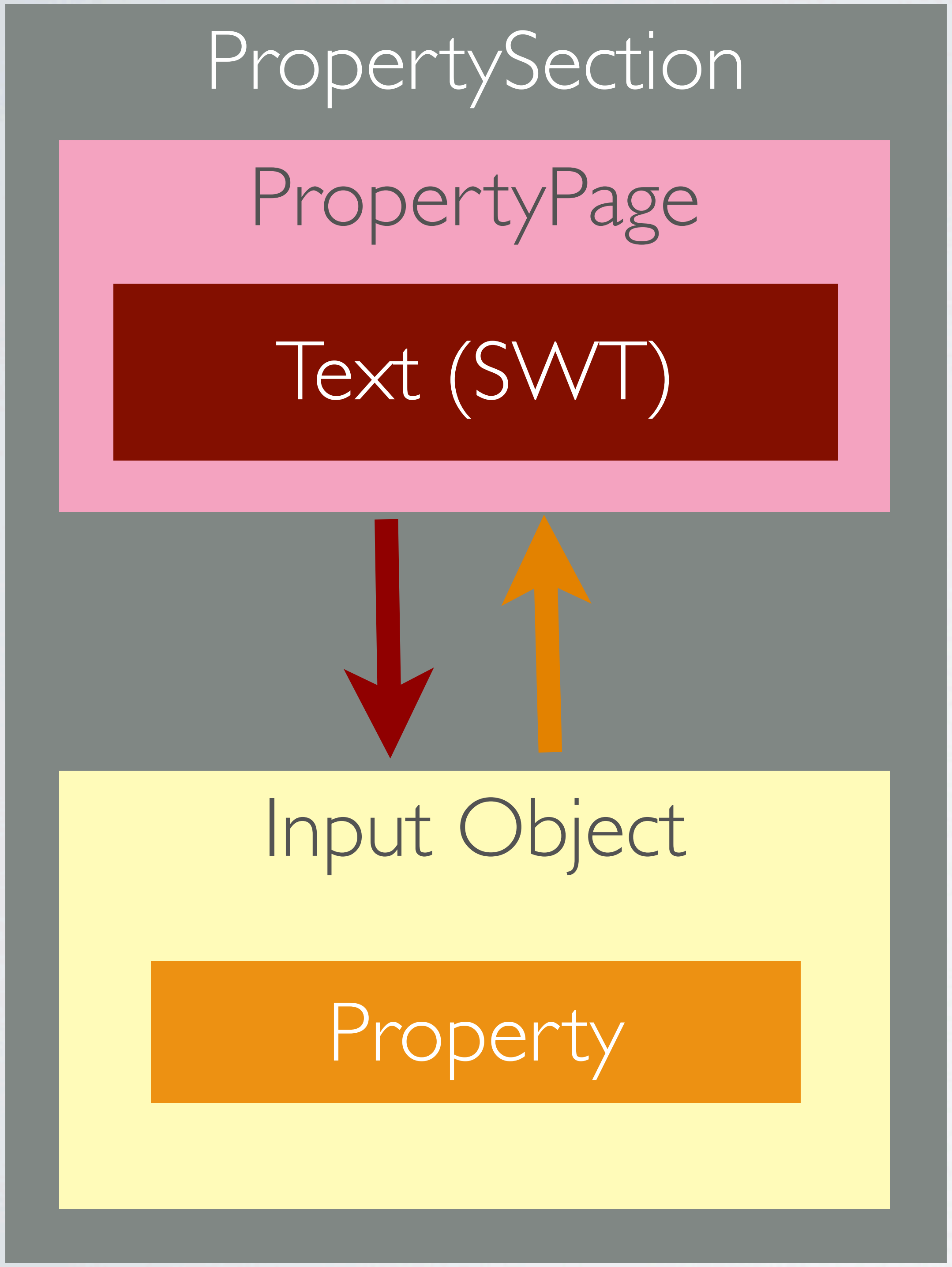
And how does it work?

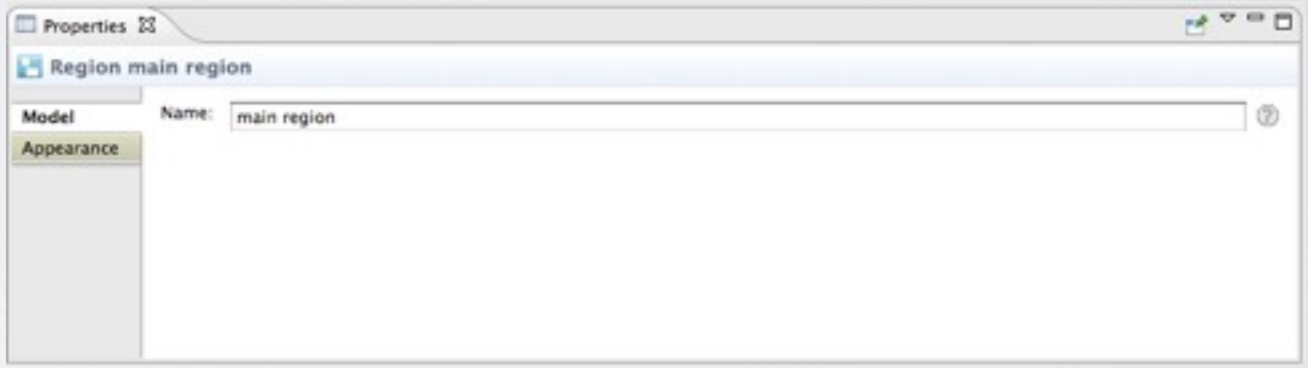
PropertySection



Name: main region

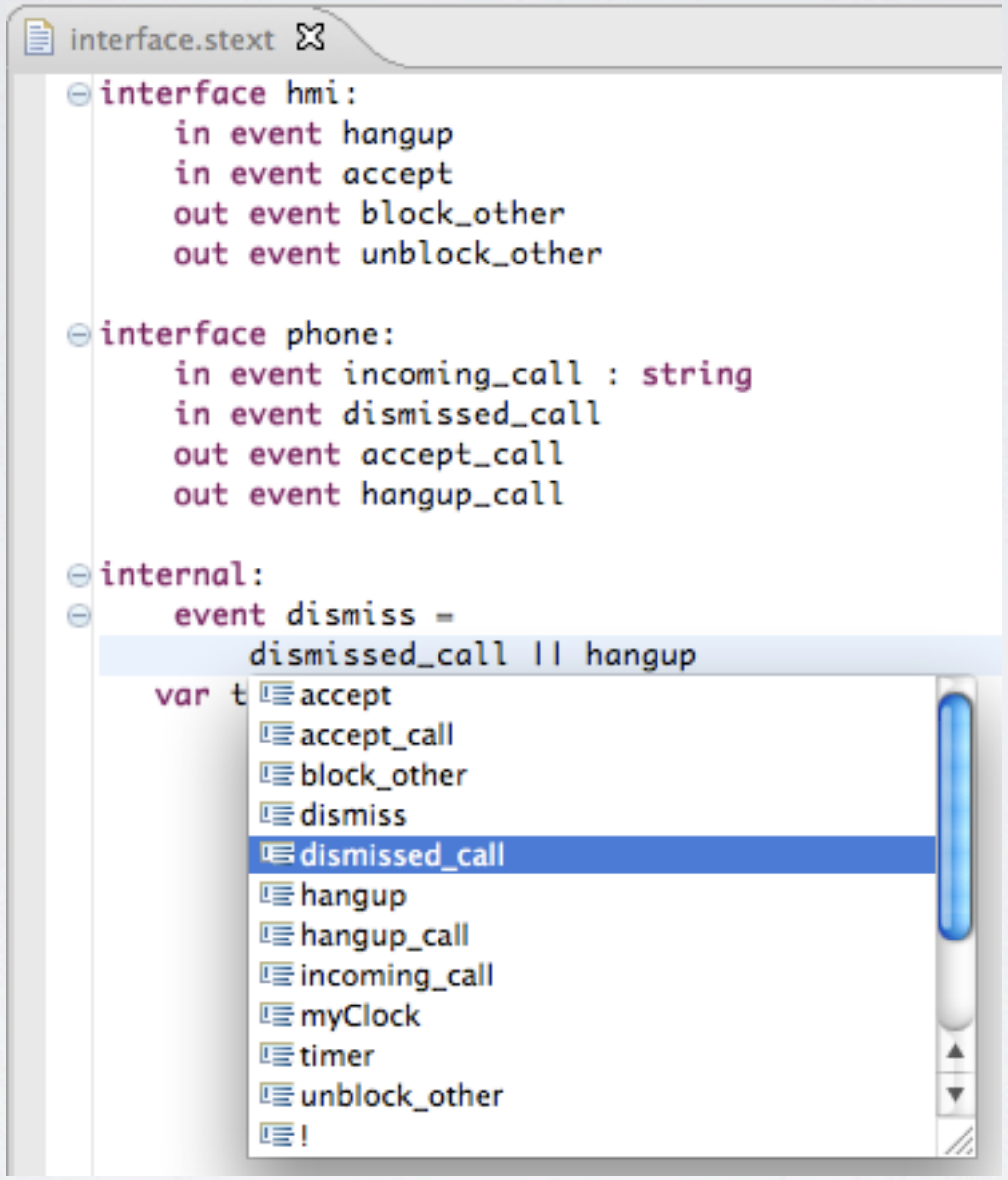
PropertySection





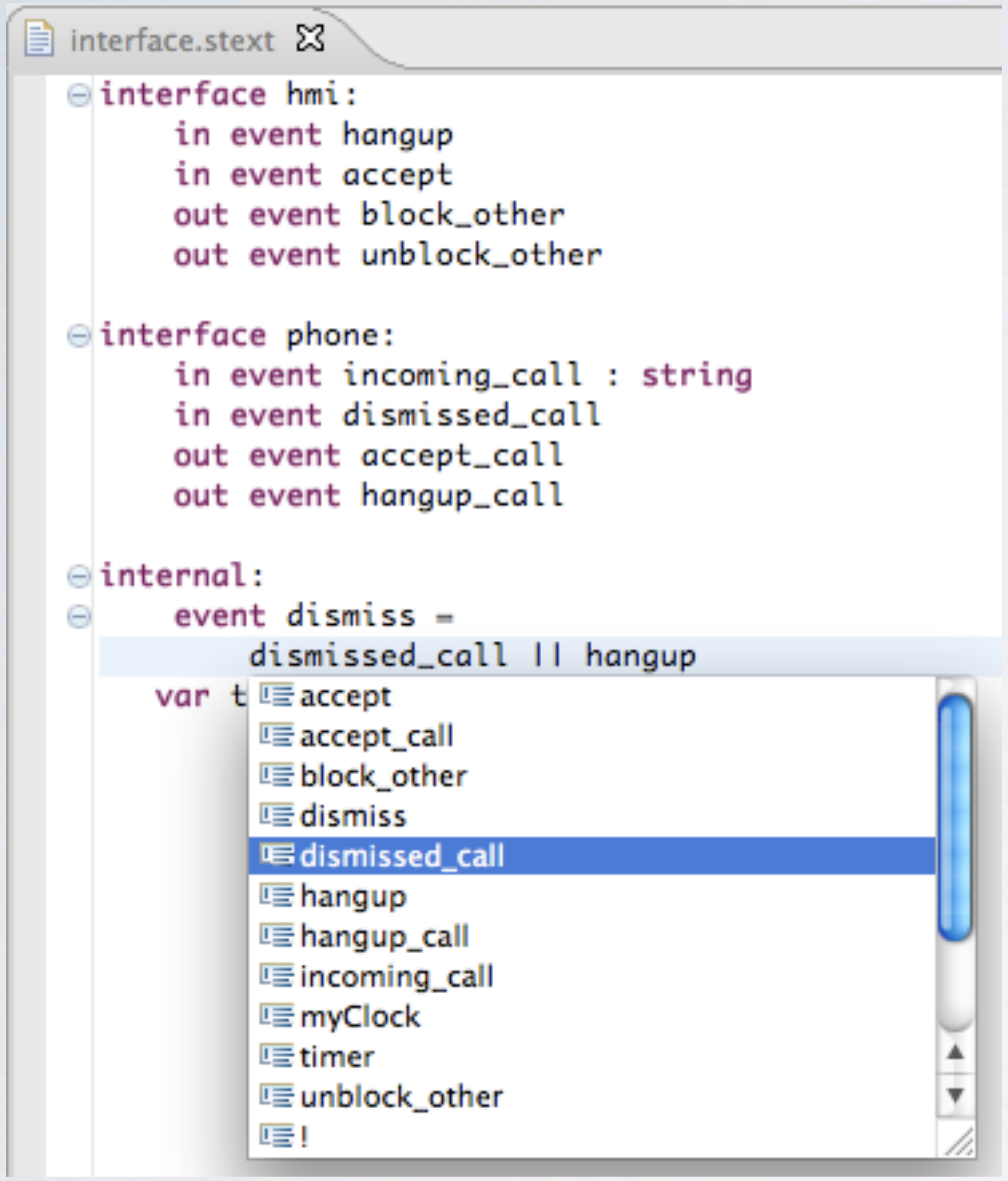
Name: main region

Xtext-Editor

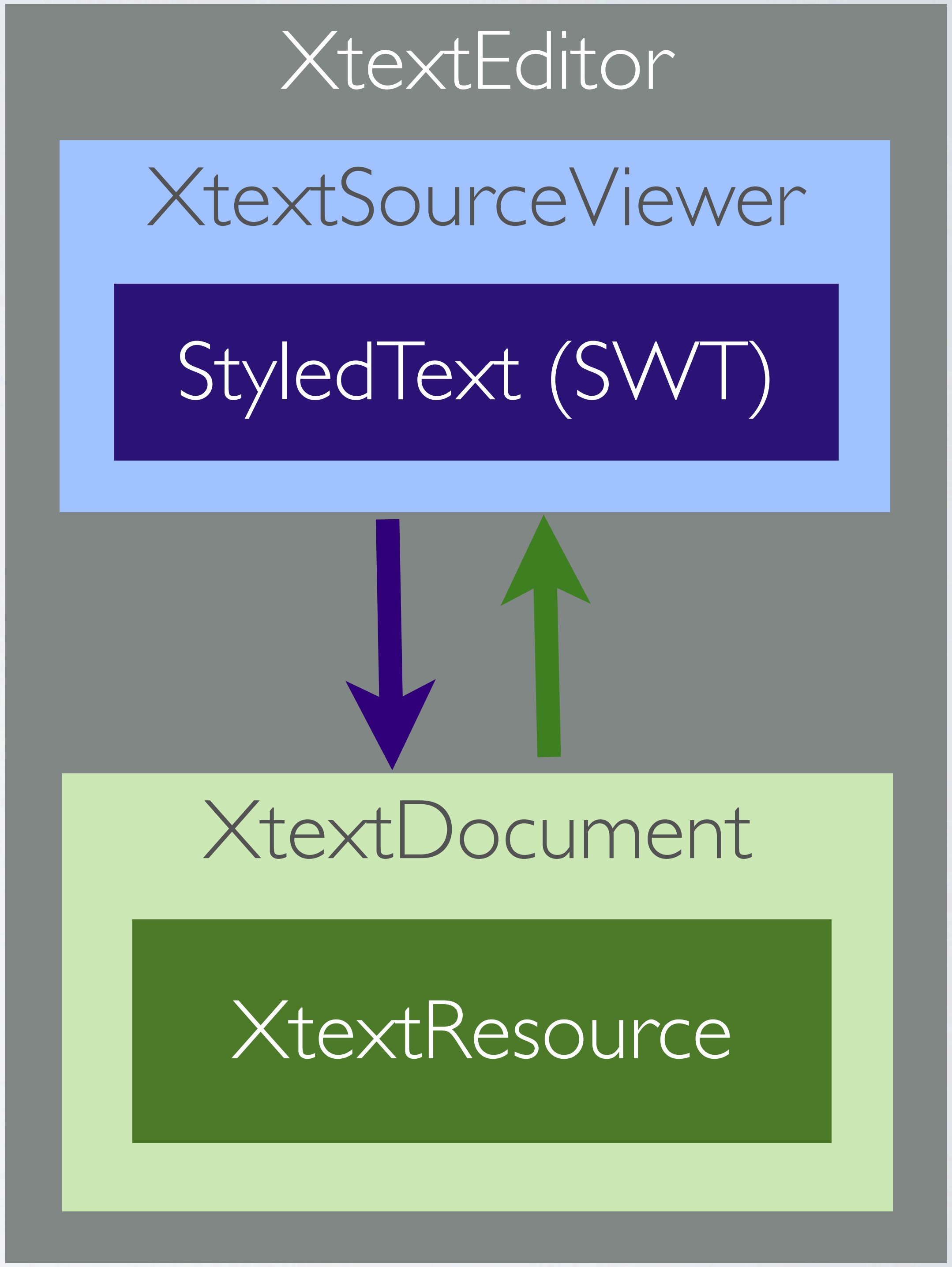


```
interface hmi:
    in event hangup
    in event accept
    out event block_other
    out event unblock_other

interface phone:
    in event incoming_call : string
    in event dismissed_call
    out event accept_call
    out event hangup_call

internal:
    event dismiss =
        dismissed_call || hangup
   var t
```

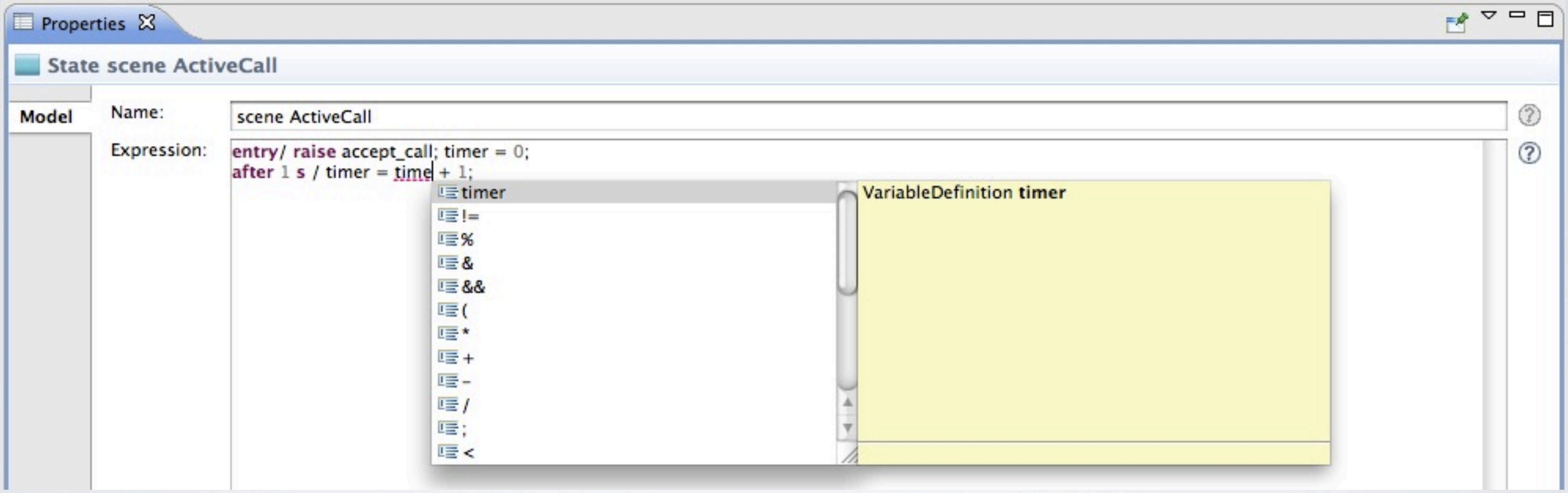
Xtext-Editor



```
interface.stext

interface hmi:
    in event hangup
    in event accept
    out event block_other
    out event unblock_other

interface phone:
    in event incoming_call : string
    in event dismissed_call
    out event accept_call
    out event hangup_call

internal:
    event dismiss =
        dismissed_call || hangup
  var t
```



Xtext-JFace-Integration - The Principle



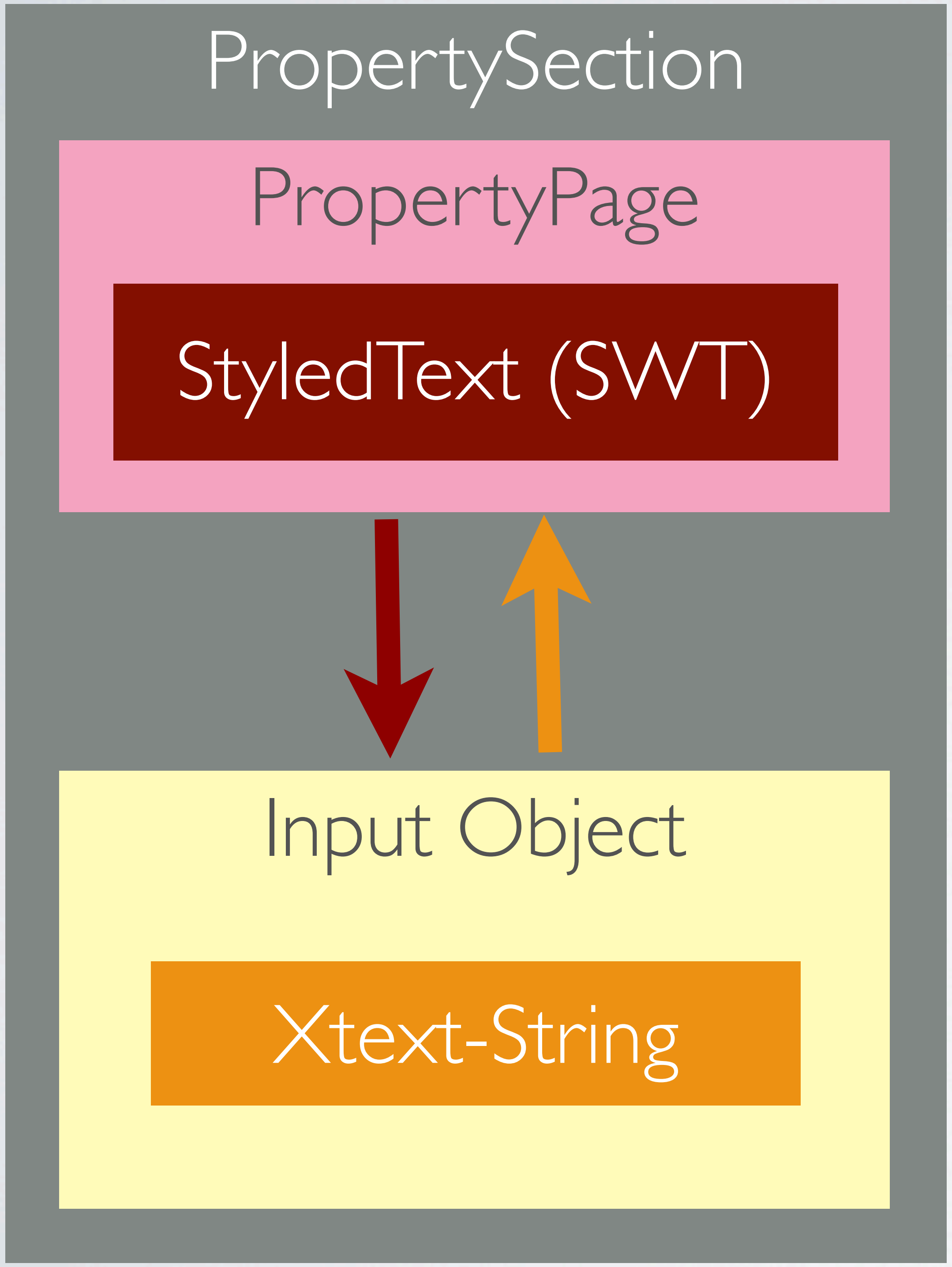
scene ActiveCall

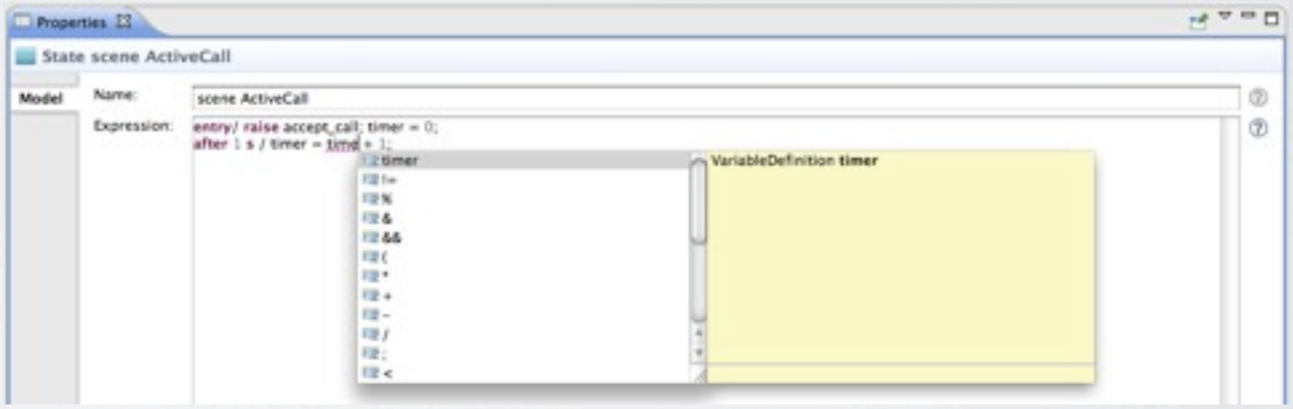
```
entry/ raise accept_call; timer = 0;
after 1 s / timer = time + 1;
```

```
timer
!=
%
&
&&
(
*
+
-
/
;
<
```

VariableDefinition **timer**

Xtext-JFace-Integration - The Principle



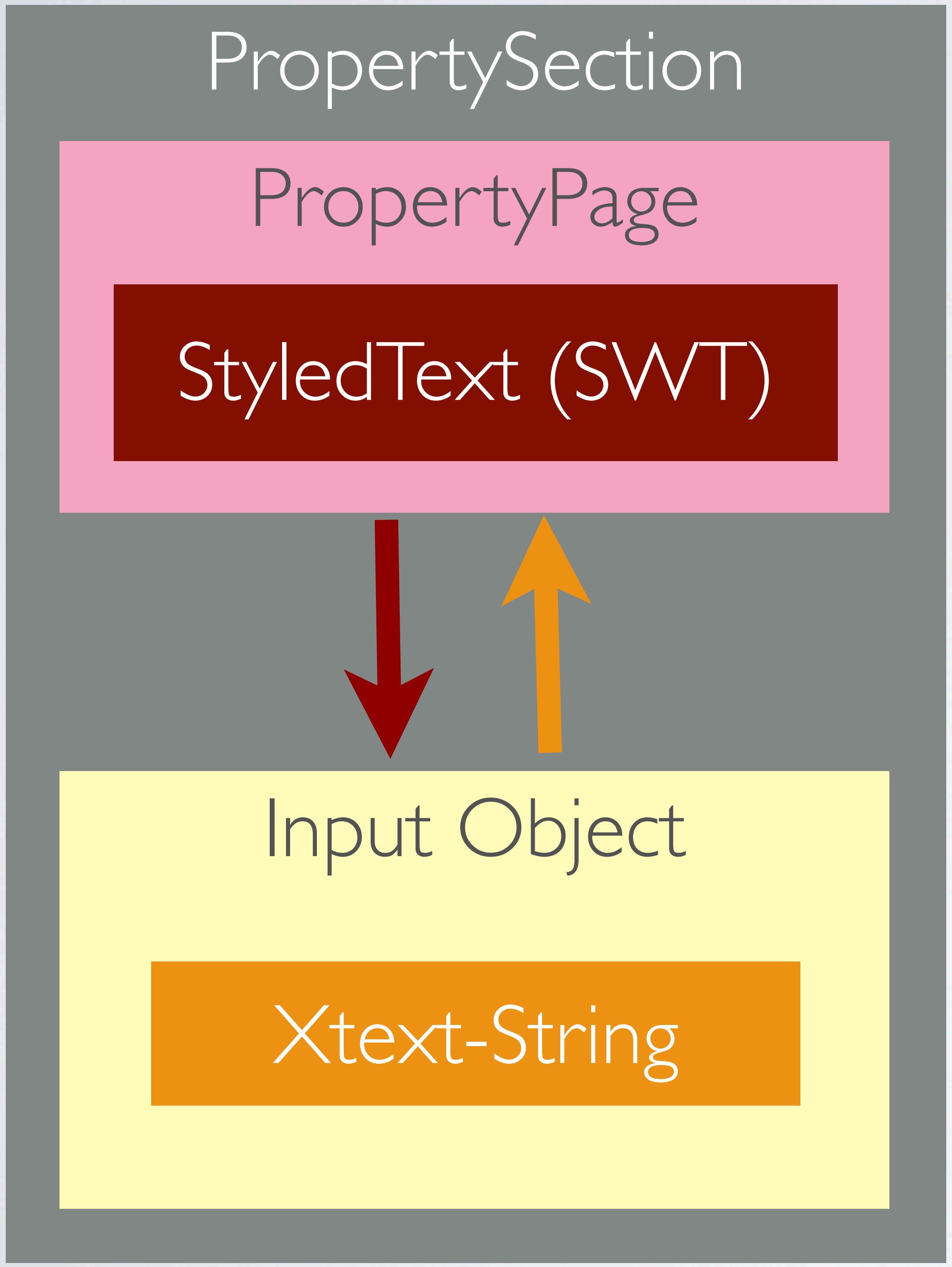


Name: scene ActiveCall

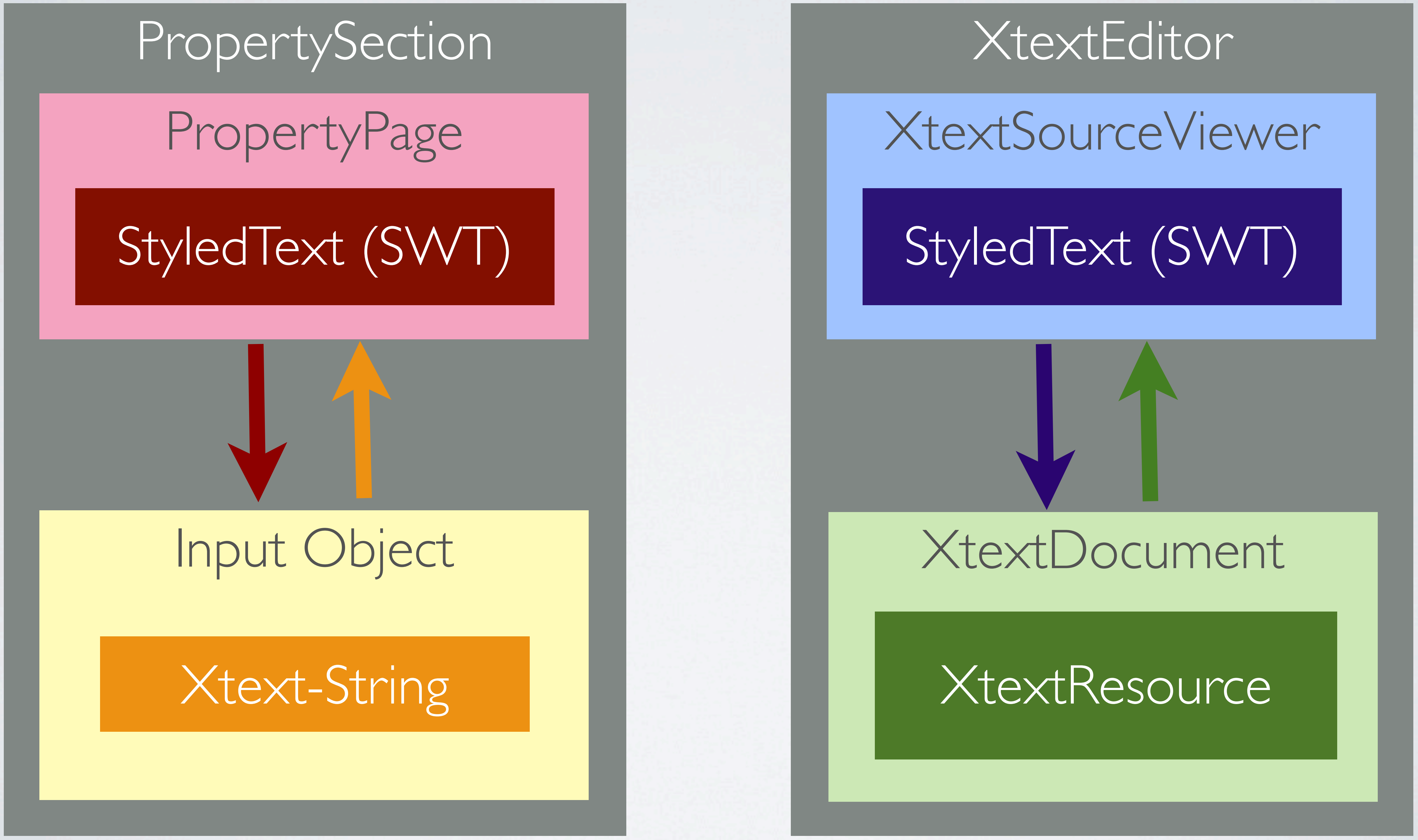
Expression:

```
entry/ raise accept_call; timer = 0;
after 1 s / timer = timd + 1;
```

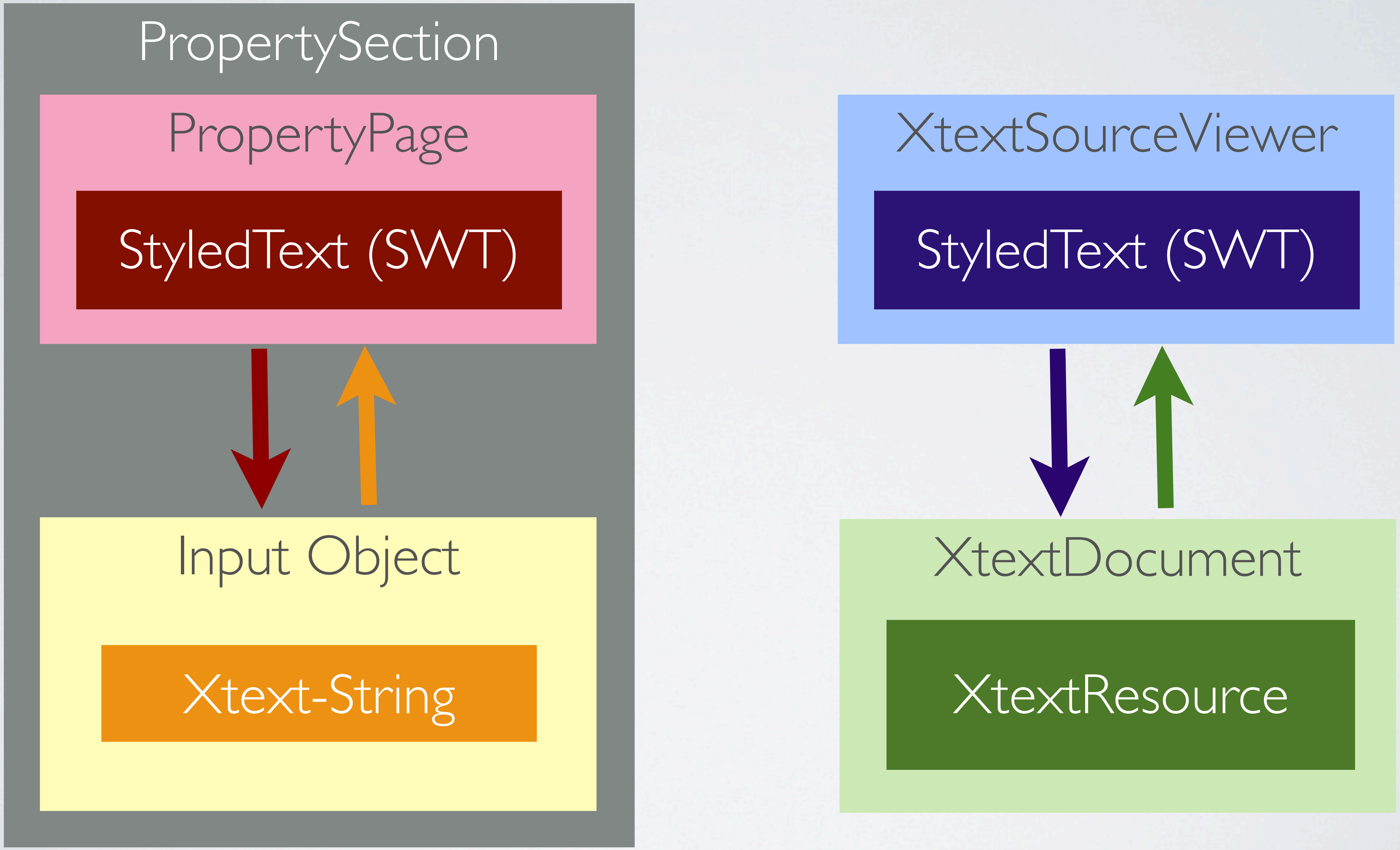
Xtext-JFace-Integration - The Principle



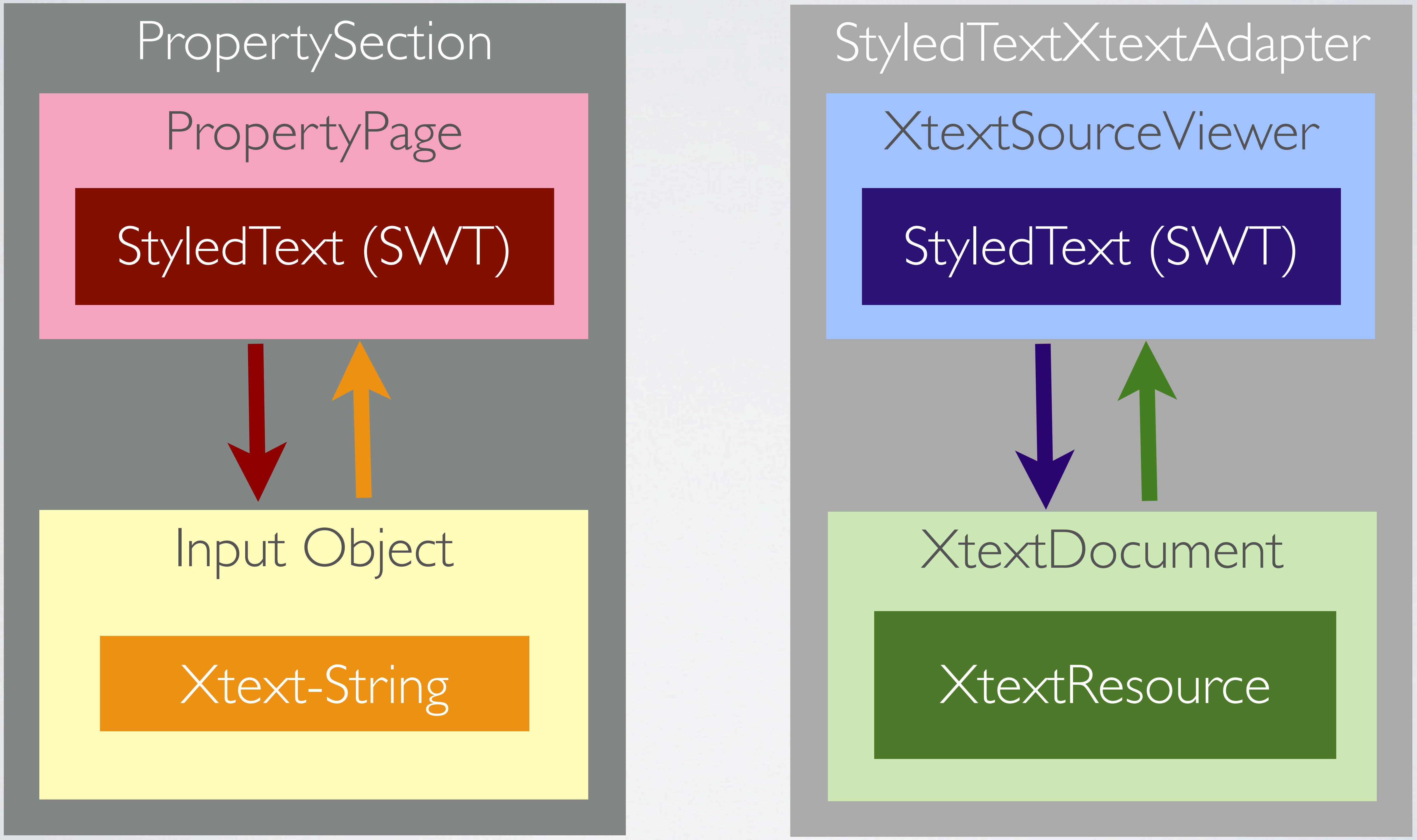
Xtext-JFace-Integration - The Principle



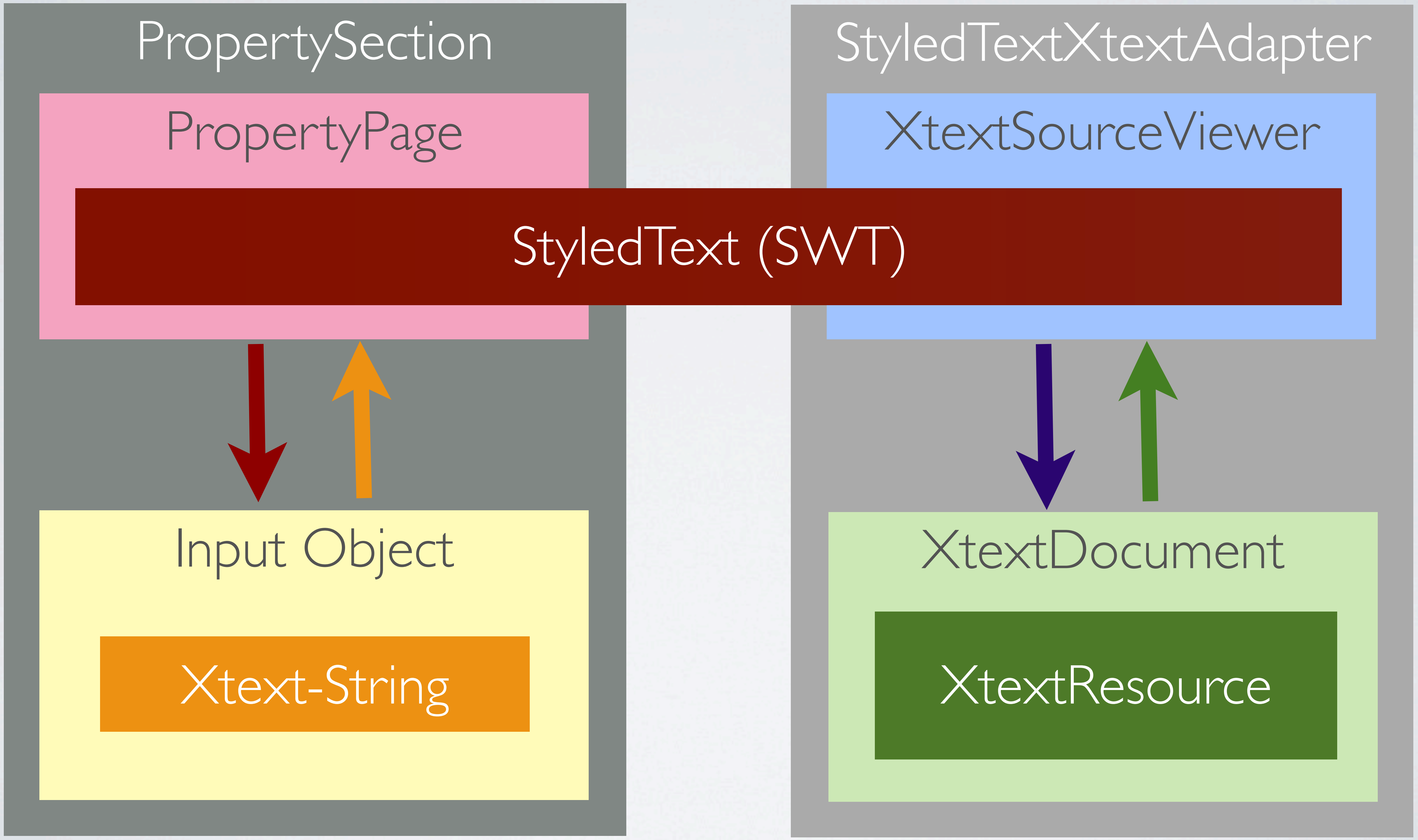
Xtext-JFace-Integration - The Principle



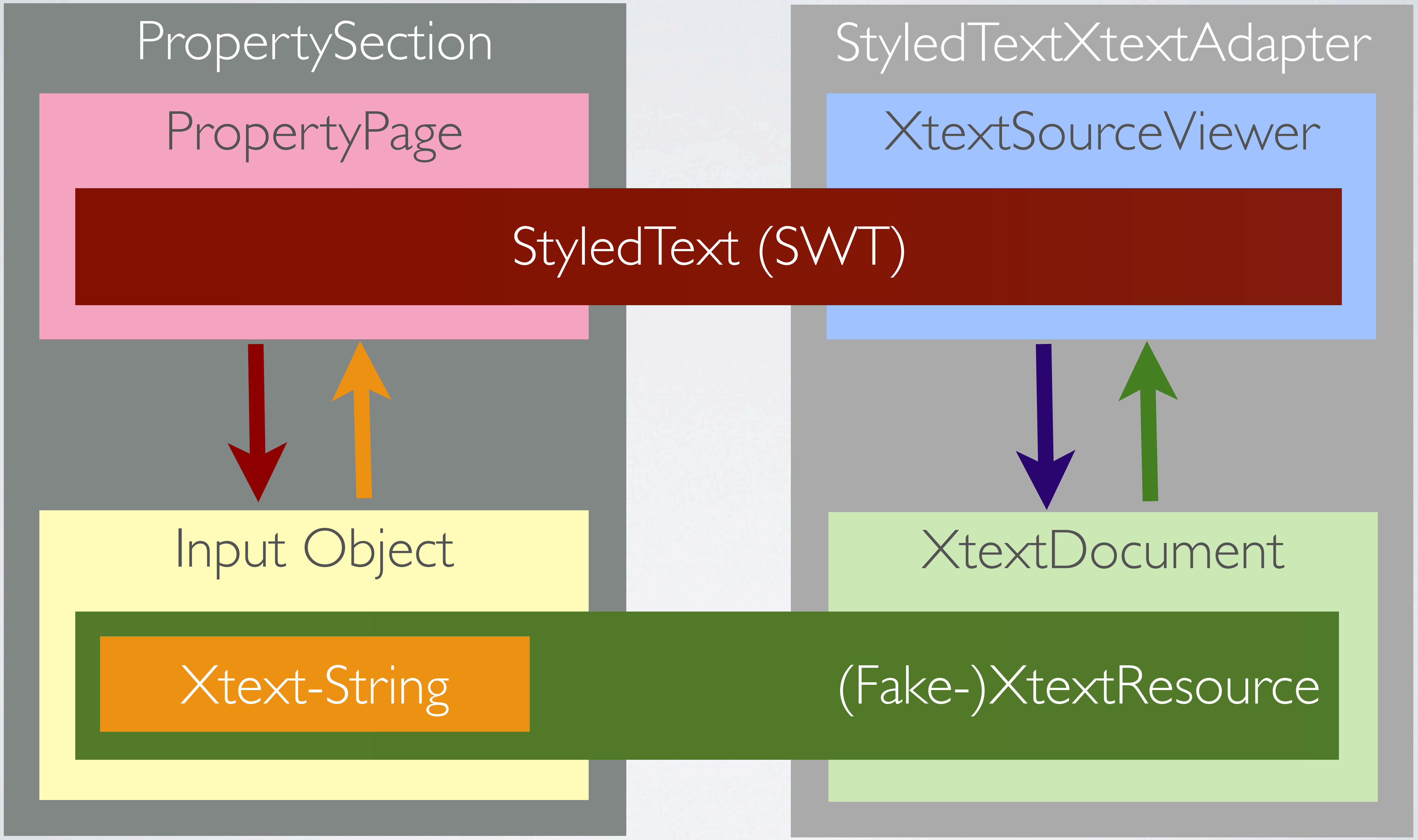
Xtext-JFace-Integration - The Principle



Xtext-JFace-Integration - The Principle



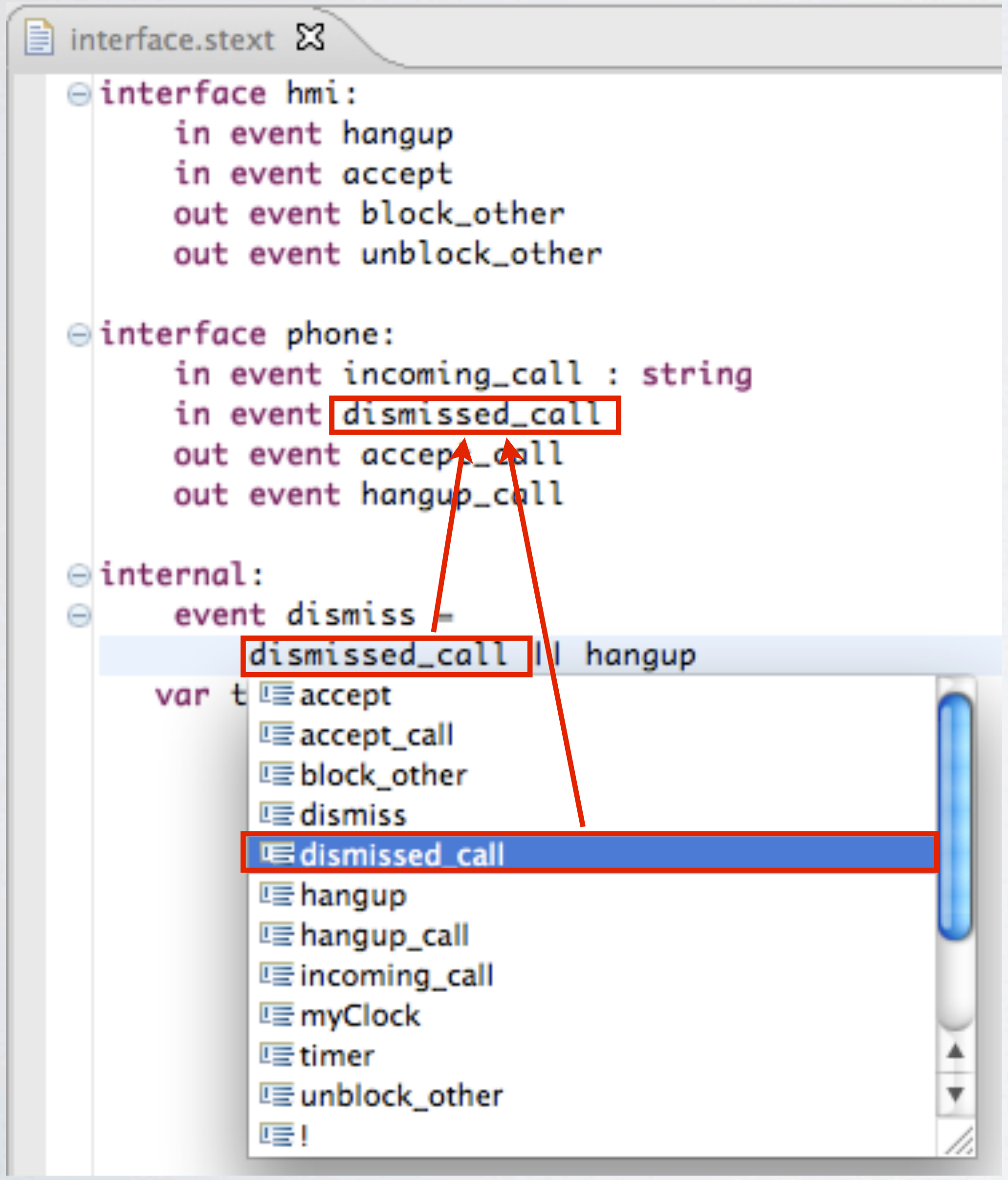
Xtext-JFace-Integration - The Principle



And what about scoping?

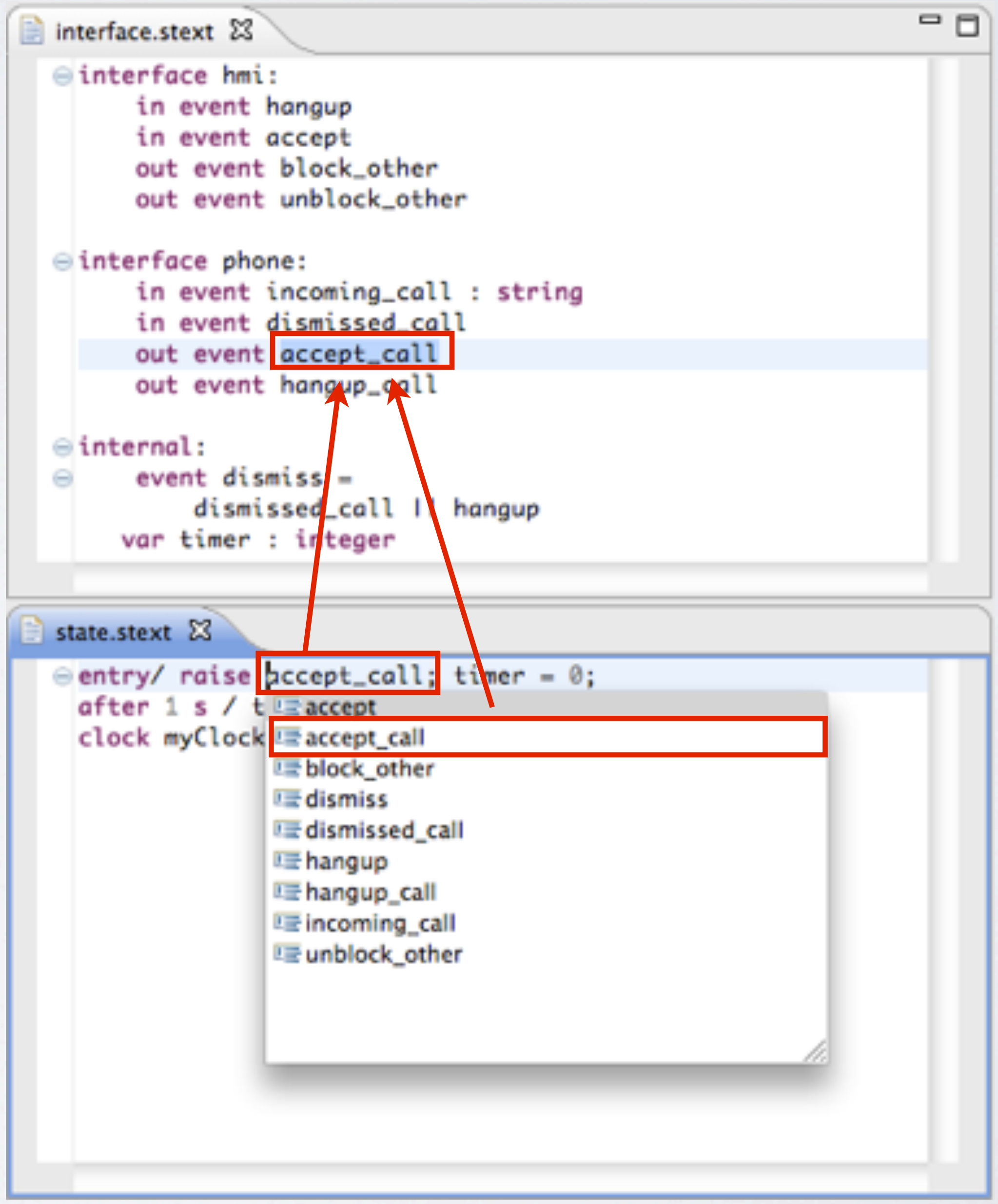
Scoping in Xtext

- When resolving cross-references scoping decides which elements (of potentially different resources) are referable



```
interface hmi:
    in event hangup
    in event accept
    out event block_other
    out event unblock_other

interface phone:
    in event incoming_call : string
    in event dismissed_call
    out event accept_call
    out event hangup_call

internal:
    event dismiss =
        dismissed_call || hangup
  var t
```

```
accept
accept_call
block_other
dismiss
dismissed_call
hangup
hangup_call
incoming_call
myClock
timer
unblock_other
!
```



```
interface hmi:
    in event hangup
    in event accept
    out event block_other
    out event unblock_other

interface phone:
    in event incoming_call : string
    in event dismissed_call
    out event accept_call
    out event hangup_call

internal:
    event dismiss =
        dismissed_call || hangup
   var timer : integer
```

```
entry/ raise accept_call; timer = 0;
after 1 s / t
clock myClock
```

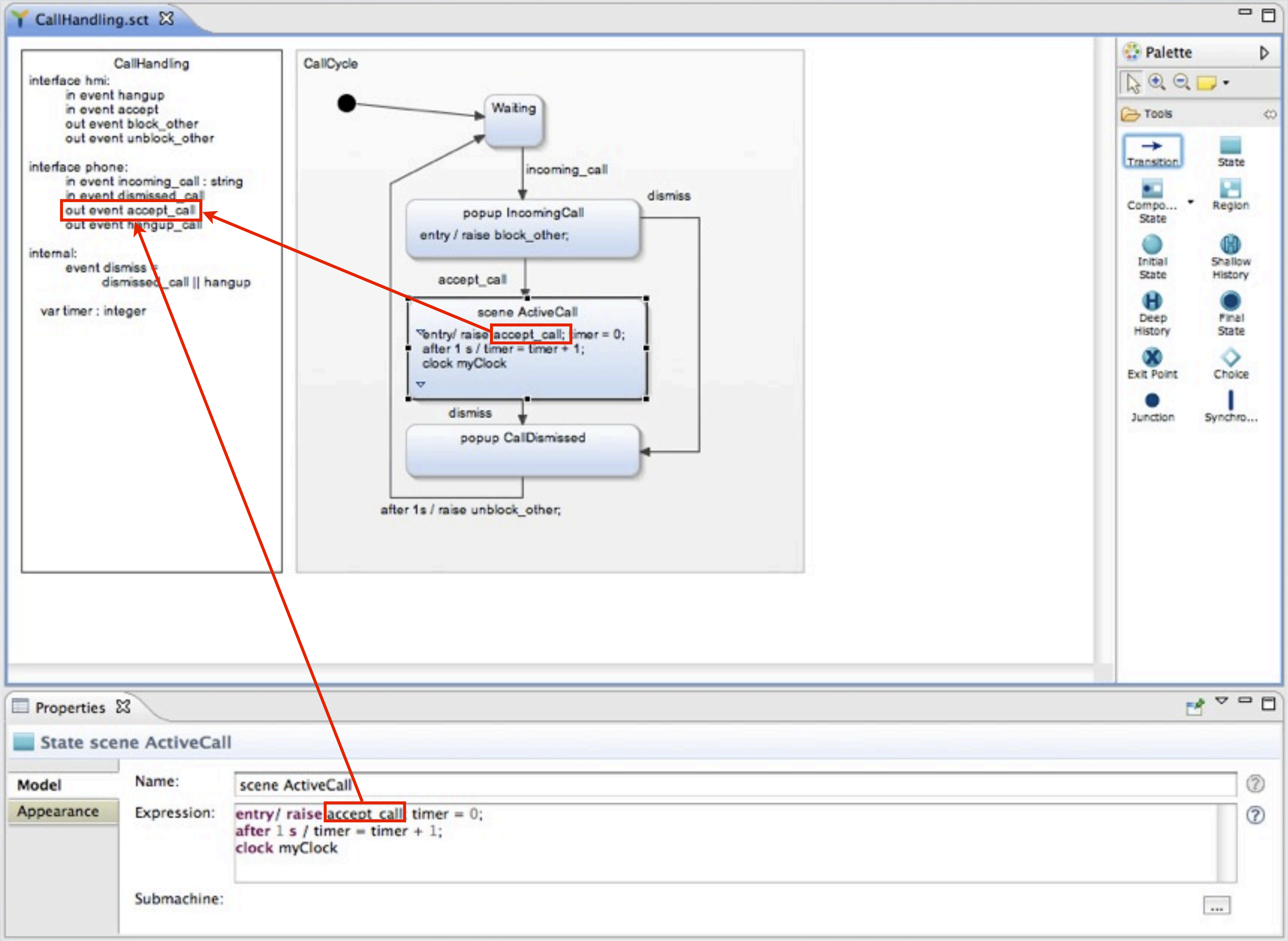
```
accept
accept_call
block_other
dismiss
dismissed_call
hangup
hangup_call
incoming_call
unblock_other
```

Scoping in Xtext (continued)

- Xtext distincs two notions of scope:
  - Local Scope (internal to the context resource)
  - Global Scope (external to the context resource)

- Global Scope is based on ResourceDescriptions which are provided by an indexing mechanism (Xtext builder)
  - XtextEditors are dirty-aware, i.e. their current editing state is proclamated to the ResourceDescriptions

Scoping in Embedded Xtext?



```
CallHandling
interface hmi:
        in event hangup
        in event accept
        out event block_other
        out event unblock_other

interface phone:
        in event incoming_call : string
        in event dismissed_call
        out event accept_call
        out event hangup_call

internal:
        event dismiss =
            dismissed_call || hangup

    var timer : integer
```

State scene ActiveCall

Name: scene ActiveCall

Expression:

```
entry/ raise accept_call; timer = 0;
after 1 s / timer = timer + 1;
clock myClock
```

Submachine:

Fake-XtextResource

The Fake-XtextResource used by the XtextAdapter does only contain the currently edited Xtext-String, not any other contents of the context resource

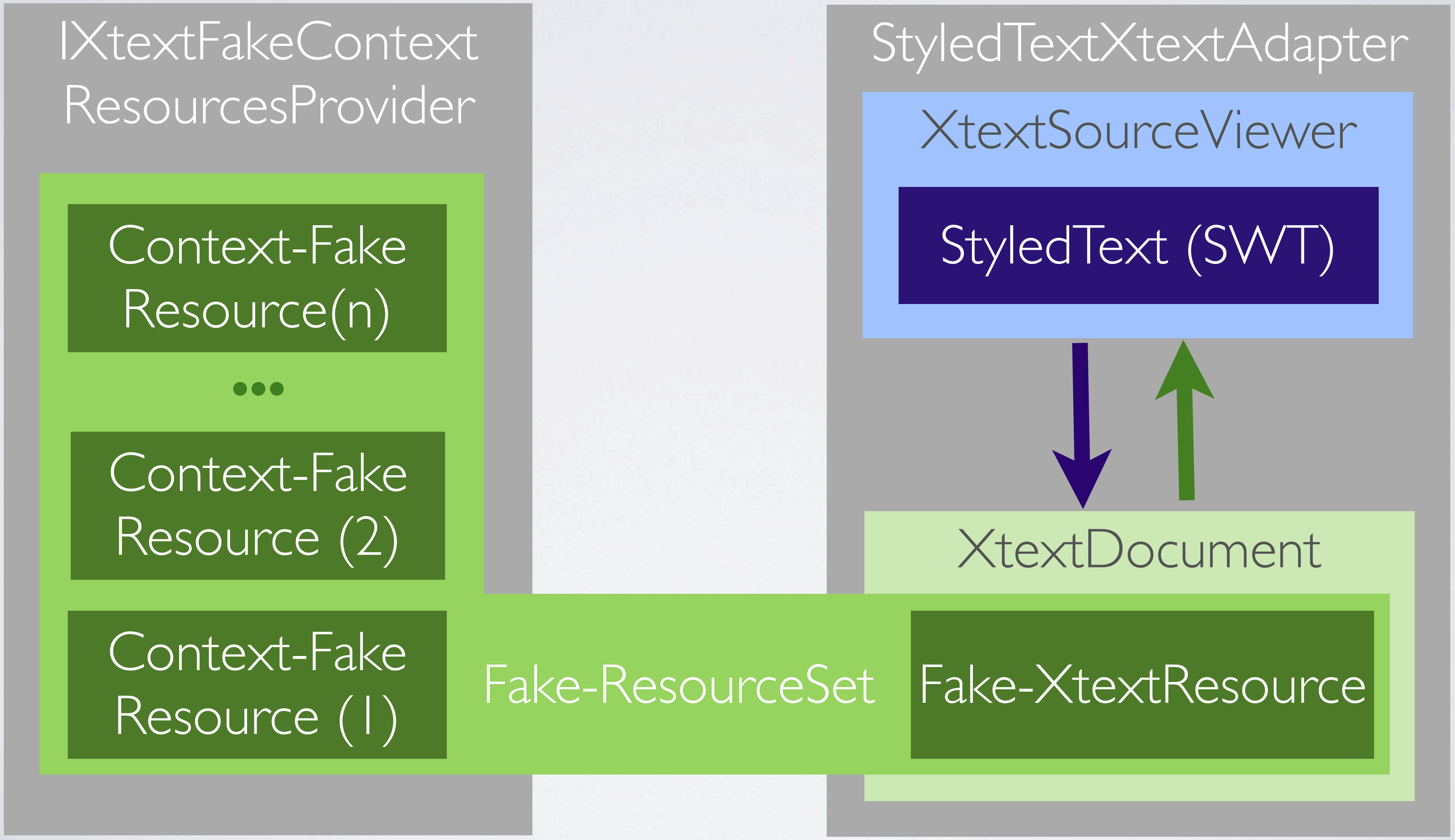
➡ Local scope will allow us to refer to elements in the edited Xtext-String, but not outside

➡ Global scope will allow to reference external elements, but by default context resource contents is not proclamated to the global scope

Scoping based on Fake Resources

- Sophisticated Solution:
  - Expose all Xtext-Strings contained within context resource to ResourceDescriptions „dirty-state aware“
- Simple Solution:
  - Populate the Fake-Xtext-Resource‘s ResourceSet with other Fake-Context-Resources (e.g. one for each Xtext-String)

IFakeContextResourcesProvider



Populating Fake Resource Set

- `IXtextContextFakeResourcesProvider` allows to populate the fake ResourceSet:

```
IXtextFakeContextResourcesProvider provider =
                   new IXtextFakeContextResourcesProvider(){
  public void populateFakeResourceSet(
                             ResourceSet fakeResourceSet,
                             XtextResource fakeResource){
    // create context fake resources via given resource set
    ...
   }
};

xtextAdapter = new StyledTextXtextAdapter(getInjector(), provider);
xtextAdapter.adapt(styledText);
```

And where can I get it?

Xtext-Integration @Yakindu

- Xtext-JFace-Integration and Xtext-GMF-Integration is made available by the YAKINDU project
  - Open Source / EPL
  - Project Site: http://yakindu.org
  - Eclipse Labs Site: http://code.google.com/a/eclipselabs.org/p/yakindu/
  - Update Site: http://updates.yakindu.com/indigo/milestones/
  - Source Code: http://svn.codespot.com/a/eclipselabs.org/yakindu/BASE/trunk/de.itemis.xtext.utils

Thank You! Questions?