

# Model-Driven Software Development with *SuperCollider* and the *UML*

Jens Gulden, [jgulden@cs.tu-berlin.de](mailto:jgulden@cs.tu-berlin.de)

<http://swiki.hfbk-hamburg.de/MusicTechnology/751>



# Overview

- **SuperCollider** development
- The **Unified Modeling Language (UML)**
- **Adopting the UML to SuperCollider**
- **Examples, Plans & Conclusion**



# Typical SuperCollider Development Environment

The screenshot displays a typical SuperCollider development environment on a Linux system. The main window is Emacs, showing the SCLang code for a server and a client. The code includes comments and function definitions for starting the server and testing the client.

```
// --- Usage examples for the prototype implementation of
// --- XML Specification of Time, Positions and Parts of a Waveform
// Jens Gulden 08/2005, jgulden@cs.tu-berlin.de

// --- start server ---
Server.local.boot;

// -----

// TEST 1: short form
// (SFSliceInstrument plays first Cuelist by default.)

SFSliceInstrument.new("examples/slice.xml").verbose_(true).play;

// or even shorter, without console output:
-- sfslicetest.sc (SCLang)--L13--Top-----
//
SFSlice {
  // --- attributes
  var <>name; // type String
  var <>childrenByName; // type Dictionary
  var <>anchorsByName; // type Dictionary
  var padCount = 0; // type int
  var lastSlice; // type SFSlice
-- SFSlice.sc<2> (SCLang)--L15-- 2%-----
JackDriver: couldn't connect alsa_pcm:capture_2 to SuperCollider
JackDriver: max output latency 0.046440
JackDriver: connected SuperCollider:out_1 to alsa_pcm:capture_2
JackDriver: connected SuperCollider:out_2 to alsa_pcm:capture_2
SuperCollider 3 server ready..
notification is on
[]
localhost (127.0.0.1) 4.5|6.4 % u: 0 s: 0 q: 2 d

jgulden@synthie:~$
jgulden@synthie:~$ emacs -sclang sc/sfslicetest.sc
```

The JACK Audio Connection Kit is running, showing the status of the audio connection. The Connections window shows the SuperCollider client connected to the ALSA PCM output.



# Typical SuperCollider Development Environment

Application or  
Test Code

The screenshot displays a typical SuperCollider development environment. The main window is Emacs, showing a SuperCollider patch file named `sfslicetest.sc`. The code includes comments and a test section:

```
// --- Usage examples for the prototype implementation of  
// --- XML Specification of Time, Positions and Parts of a Waveform  
// Jens Gulden 08/2005, jgulden@cs.tu-berlin.de  
  
// --- start server ---  
Server.local.boot;  
  
// -----  
  
// TEST 1: short form  
// (SFSliceInstrument plays first Cuelist by default.)  
  
SFSliceInstrument.new("examples/slice.xml").verbose_(true).play;  
  
// or even shorter, without console output:  
-- sfslicetest.sc (SCLang)--L13--Top-----  
//  
SFSlice {  
  // --- attributes  
  var <name; // type String  
  var <childrenByName; // type Dictionary  
  var <anchorsByName; // type Dictionary  
  var padCount = 0; // type int  
  var lastSlice; // type SFSlice  
-- SFSlice.sc<2> (SCLang)--L15-- 2%-----  
JackDriver: couldn't connect alsa_pcm:capture_2 to SuperCollider  
JackDriver: max output latency 0.046440  
JackDriver: connected SuperCollider:out_1 to alsa_pcm:capture_2  
JackDriver: connected SuperCollider:out_2 to alsa_pcm:capture_2  
SuperCollider 3 server ready..  
notification is on  
[]  
localhost (127.0.0.1) 4.5|6.4 % u: 0 s: 0 q: 2 d  
jgulden@synthie:~$  
jgulden@synthie:~$ emacs -sclang sc/sfslicetest.sc
```

The JACK Audio Connection Kit (JACK) interface is also visible, showing the "Connections - JACK Audio Connection Kit" window. It displays the "Audio" tab with "Readable Clients / Output Port" and "Writable Clients / Input Ports". The "SuperCollider" client is connected to the "alsa\_pcm" client.



# Typical SuperCollider Development Environment

**Class-Library Code (OOP)**

```
emac@synthie.flp.cs.tu-berlin.de
File Edit Options Buffers Tools SCLang Help

// --- Usage examples for the prototype implementation of
// --- XML Specification of Time, Positions and Parts of a Waveform
// Jens Gulden 08/2005, jgulden@cs.tu-berlin.de

// --- start server ---
Server.local.boot;

// -----

// TEST 1: short form
// (SFSliceInstrument plays first Cuelist by default.)
SFSliceInstrument.new("examples/slice.xml").verbose(true).play

// or even shorter, without console output:
--- sfslicetest.sc (SCLang)--L13--Top-----

SFSlice {
  // --- attributes
  var <name; // type String
  var <childrenByName; // type Dictionary
  var <anchorsByName; // type Dictionary
  var padCount = 0; // type int
  var lastSlice; // type SFSlice
}
--- sfslicetest.sc<2> (SCLang)--L15-- 2%-----
JackDriver: couldn't connect alsa_pcm:capture_2 to SuperCollider
JackDriver: max output latency 0.046440
JackDriver: connected SuperCollider:out_1 to alsa_pcm:capture_1
JackDriver: connected SuperCollider:out_2 to alsa_pcm:capture_2
SuperCollider 3 server ready..
notification is on
[]

localhost (127.0.0.1) 4.5|6.4 % u: 0 s: 0 q: 2 d

jgulden@synthie:~$
jgulden@synthie:~$ emacs -sclang sc/sfslicetest.sc
```

JACK Audio Connection Kit [(default)] Started.

Started -- 3.9% 44100 Hz  
Stopped -- 00:00:00

Connections - JACK Audio Connection Kit

Audio	MIDI
Readable Clients / Output Port	Writable Clients / Input Ports
SuperCollider	alsa_pcm
	SuperCollider



# Model-Driven Development

The screenshot displays the Poseidon for UML Professional Edition IDE. The main window shows a UML class diagram with classes SFSlice, SFAnchor, SFCueList, and SF3. SFSlice has associations with SFAnchor (via +start, +end, +current) and SFCueList (via +parent). SFAnchor has associations with SFSlice (via +reAnchor) and SFCueList (via +time). SFCueList has associations with SFAnchor (via +anchors) and SFSlice (via +children). SF3 has an association with SFSlice (via +master, 0..1).

An Emacs editor window titled 'emacs@synthie.fip.cs.tu-berlin.de' is open, showing SCLang code:

```
// TEST 1: short form
// (SFSliceInstrument plays first Cuelist.)
SFSliceInstrument.new("examples/slice.xml").verbose_(true).play;
// or even shorter, without console output:
:** sfslicetest.sc (SCLang)--L13--16%-----
JackDriver: connected SuperCollider:out_2 to also_pcm:playback_2
SuperCollider 3 server ready..
notification is on
localhost (127.0.0.1) 3|4.4 % u: 0 s: 0 q: 2 d: 109
```

A terminal window titled 'Befehlsfenster 40' shows the command 'emacs -sclang sc/sfslicetest.sc' being executed.

The bottom right shows a SuperCollider window with a 'Connect' button.



# Model-Driven Development

The screenshot displays the Poseidon for UML Professional Edition IDE. The main window shows a UML class diagram with classes SFSlice, SFAnchor, and SF3. SFSlice has attributes name, childrenByName, and ankersByName. SFAnchor has attributes relativeTo and time. SF3 has attributes master and slice. The diagram also shows relationships like +start, +end, +current, +ownerSlice, +anchors, +relativeTo, +reAnchor, and +time.

A terminal window (emacs@synthie.fip.cs.tu-berlin.de) is open, showing the execution of a test script (sfslicetest.sc) using SCLang. The output indicates that the SuperCollider server is ready and the test is running on localhost (127.0.0.1).

The source code editor shows the following code for SFSlice:

```
// @poseidon-object-id [I1e00761m103d7a76e9dmm67d5]
SFSlice {
    // --- attributes
    var <>name; // type String
    var <>childrenByName; // type Dictionary
    var <>ankersByName; // type Dictionary
    var padCount = 0; // type int
    var lastSlice; // type SFSlice
```

Application or  
Test Code



# Model-Driven Development

The screenshot displays the Poseidon for UML Professional Edition environment. The main window shows a UML class diagram with the following classes and relationships:

- SFSlice**: Has attributes `+ start`, `+ end`, `+ current`, `+ ownerSlice`, `+ masterslice`, `+ cues (ordered)`, and `@ children`. It has a `+ parent` association.
- SFAnchor**: Has attributes `+ reAnchor` and `+ time`. It has a `+ relativeTo` association.
- SFCueList**: Has an association with `SFSlice`.

Overlaid on the diagram is an Emacs editor window titled `emacs@synthie.fip.cs.tu-berlin.de` showing the following SCLang code:

```
// TEST 1: short form
// (SFSliceInstrument plays first CueList.)
SFSliceInstrument.new("examples/slice.xml").verbose_(true).play;

// or even shorter, without console output:
:** sfslicetest.sc (SCLang)--L13--16%-----
JackDriver: connected SuperCollider:out_2 to als_a_pcm:playback_2
SuperCollider 3 server ready..
notification is on
localhost (127.0.0.1) 3|4.4 % u: 0 s: 0 q: 2 d: 109
```

Below the diagram is a console window titled `Befehlsfenster 40` showing the following shell commands and output:

```
jgulden@synthie:~$
jgulden@synthie:~$
jgulden@synthie:~$
jgulden@synthie:~$
jgulden@synthie:~$
jgulden@synthie:~$ emacs -sclang sc/sfslicetest.sc
```

The bottom of the screenshot shows a terminal window with the following text:

```
@poseidon-object-id [I1e00761m103d7a76e9dmm67d5]
SFSlice {
// --- attributes
var <start> : type SFSlice
var <end> : type SFSlice
var <current> : type Dictionary
var <ownerSlice> : type Dictionary
var <padCount> = 0; // type int
var <lastSlice>; // type SFSlice
```

Class-Library

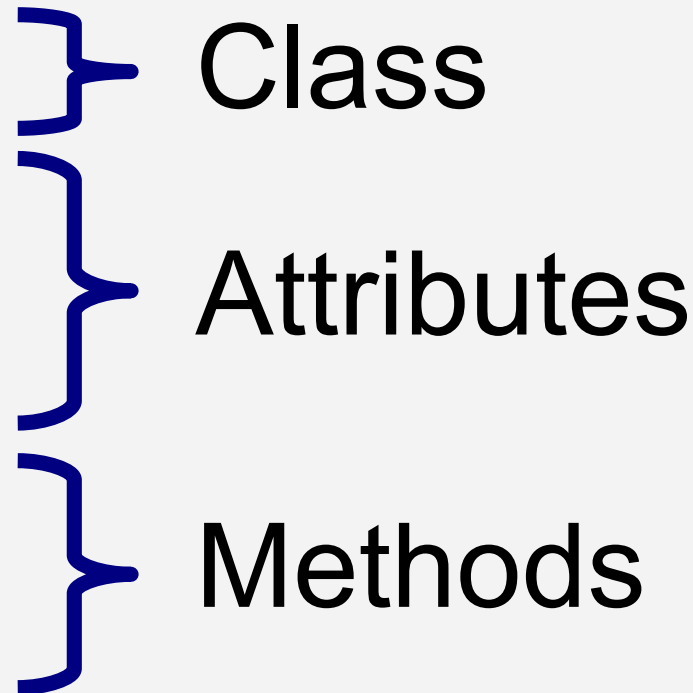
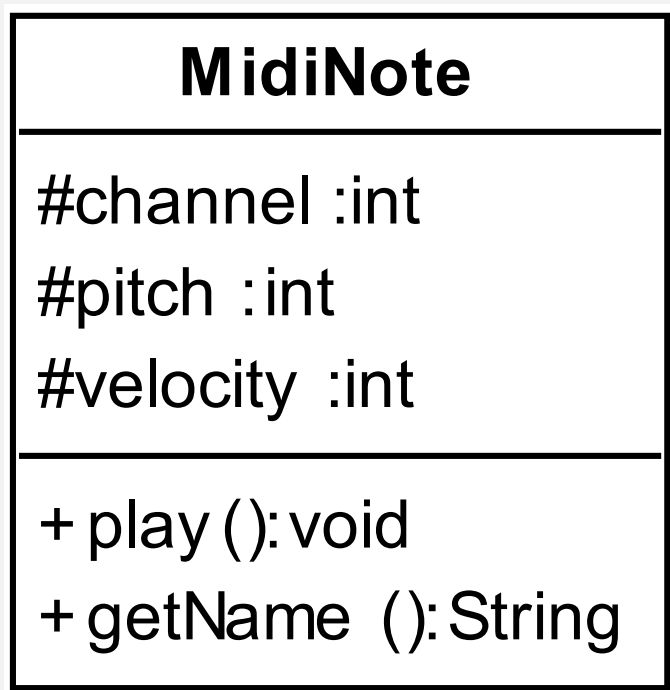


# The Unified Modeling Language (UML)

- **Visual symbols** that represent elements of object-oriented programming languages
- *Classes / Methods / Attributes, Inheritance, Relationships* etc.

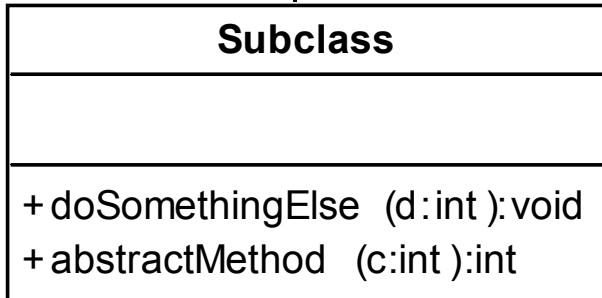
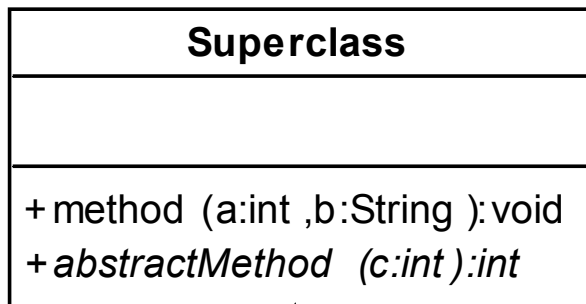


# The Unified Modeling Language (UML)

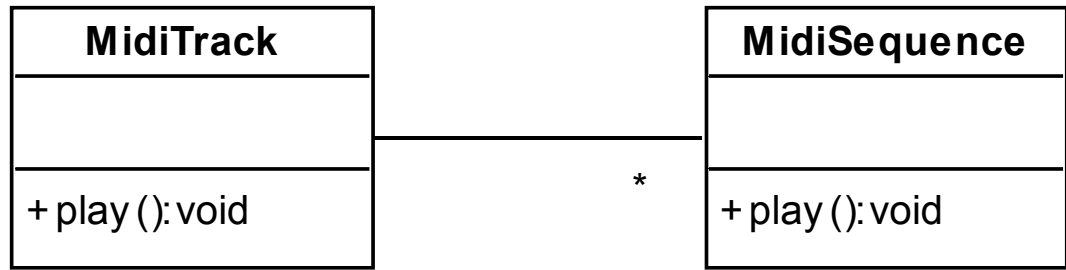




# The Unified Modeling Language (UML)



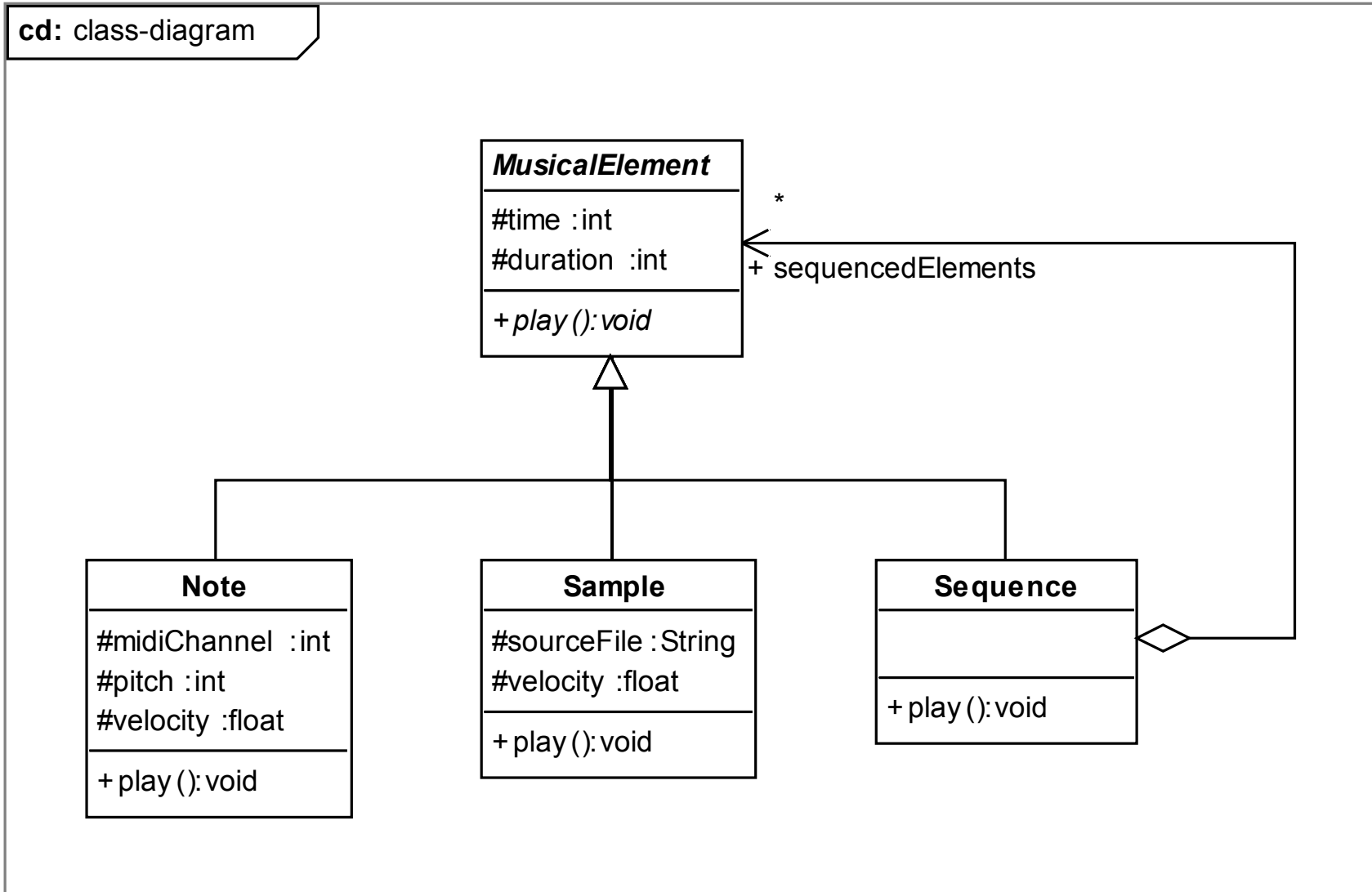
} Inheritance



Relationship



# Software-Development using the UML: Example





# Benefits of Visual Modeling

- **Navigability**
  - all accessible at a glance
- **Comprehension**
  - **spatial relationships** describe the software
  - *distances, links, groupings* etc.



# Some UML-Tools available

- **Enterprise Architect**

<http://www.sparxsystems.com>

- **Fujaba**

<http://www.fujaba.de/>

- free, GPL-license, Eclipse integration

- **Poseidon for UML (used here)**

<http://www.gentleware.de/>



# Some UML-Tools available

- **EclipseUML**

<http://www.eclipseuml.com/>

- **Rational Rose ('grandfather')**

<http://www.rational.com/>

- **Visual Paradigm**

<http://www.visual-paradigm.com/>

- **and many more...**

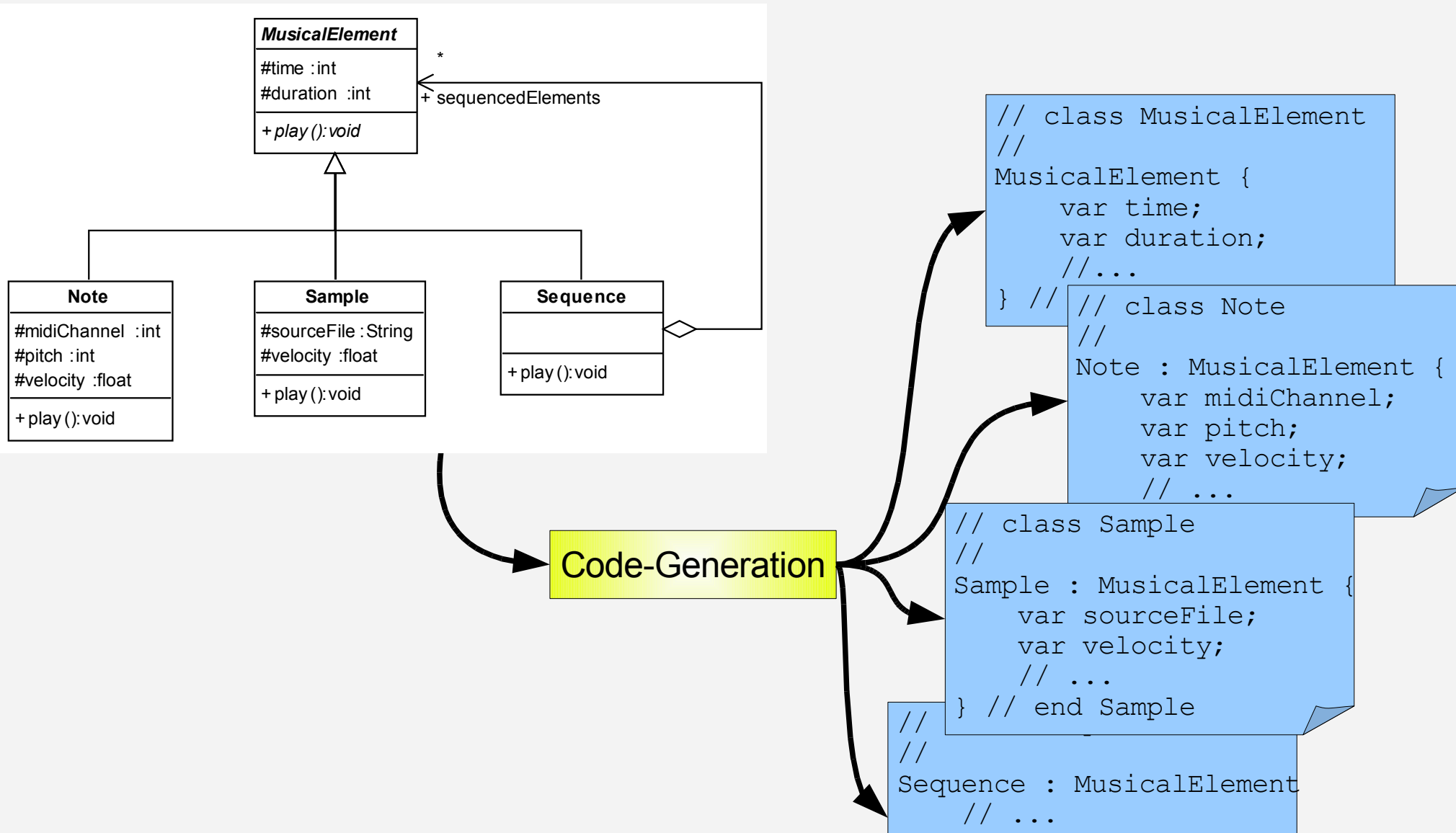


# Mapping from UML to SuperCollider

- *Not* a programming language itself
- → Translate the model to a **target programming language** to get runnable software




# Mapping from UML to SuperCollider





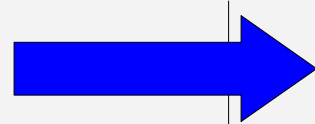
# Mapping from UML to SuperCollider

<i>UML</i>		<i>SuperCollider</i>
<b>Class</b>		<i>Class</i>
<b>Package</b>		<i>None</i>
<b>Attribute</b>		(Instance-) <i>Variable</i>
<b>Static Attribute</b>		<i>Class-Variable (*)</i>
<b>Types</b>		<i>None, comments</i>



# Mapping from UML to SuperCollider

*UML*



*SuperCollider*

**Method**

('operation')

*Method*

('message'/'fctn.')

**Static Method**

*Class-Method (\*)*

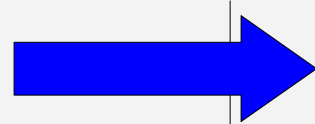
**Abstract Method**

method always  
throwing an error



# Mapping from UML to SuperCollider

*UML*



*SuperCollider*

**0..1/1..1-**  
Relationships

*Variable,*  
reference to  
single instance

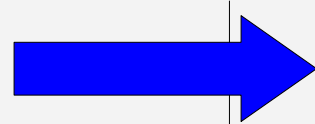
**0..\*/1..\*-**  
Relationships

*Variable, list* with  
references to  
instances



# Mapping from UML to SuperCollider

*UML*



*SuperCollider*

**Public (+)**  
visibility of  
attributes

Variable with  
*getter / setter*  
("⟨⟩").

**Package (~),**  
**Protected (#),**  
**Private (-)**

*None*, neither  
attributes nor  
methods.



# Code Templates for SuperCollider

- **Templates:**  
one way to implement mapping from UML to Code
  - here tool „Poseidon“,  
but principle applicable to others

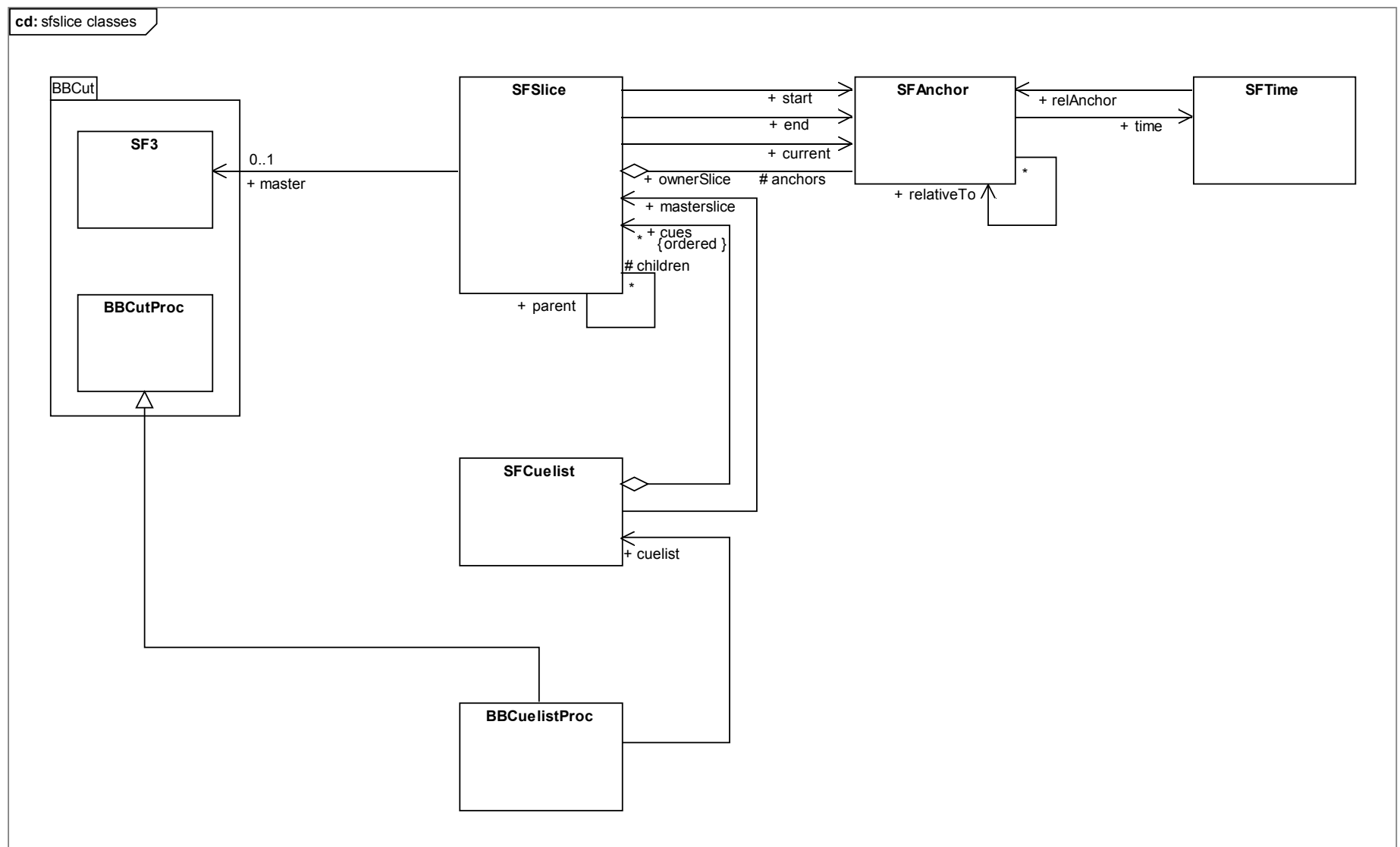


# Code Templates for SuperCollider: Excerpt

```
## --- Render a single attribute. ---  
#macro (renderOneAttribute $preparedAttr)  
...  
#if ($visibility.indexOf("public") != -1)  
#set ($SCvis = "<>")  
#end  
    ${SCvar}${SCvis}${name}${initialVal}; \  
// type \  
#stripPkg ($preparedAttr.getTypeAsString())  
#end  
...
```



# Example: SFSlice Classes



sc3 - Poseidon for UML Professional Edition

File Edit View Create Diagram Align Generation Plug-Ins Help

Package Diagram Index

SFSlice classes SFSlice classes code

SFSlice

- SFSlice->Playable
- ankersByName
- childrenByName
- lastSlice
- name
- padCount
- anker
- ankers
- ankers\_
- children
- children\_
- get
- init
- initMaster
- inner
- length
- makeSlice
- makeSliceRange
- makeSliceRangeLength
- makeSliceTo
- masterslice
- new
- newMaster
- outer

SFSlice

```

+name String
+childrenByName:Dictionary
+ankersByName:Dictionary
#padCount:int=0
#lastSlice SFSlice

+new():SFSlice
+newMaster(wave:Object):SFSlice
+initMaster(wave:Object):void
#init(part:SFSlice,n:String,strt:SFAnchor,en:SFAnchor):void
+length():SFTIME
+slice(name:String,time:Object):SFSlice
+pad(length:Object):SFSlice
+reset():SFSlice
+position(pos:Object):SFSlice
+inner(i:int):SFSlice
+outer(i:int):SFSlice
+get(name:String):SFSlice
+children():Array
+children_(newChildren:Array):void
+anker(name:String,time:Object,relativeTo:SFAnchor):SFSlice
+ankers():Array
+ankers_(newAnkers:Array):void
+sliceTo(name:String,endTime:Object):SFSlice
+sliceRange(name:String,startTime:Object,endTime:Object):SFSlice
+masterslice():SFSlice
+post(level:int):void

```

SFAnchor

```

+name String
+new(name:String,slice:SFSlice,relativeTo:SFAnchor):SFAnchor
+init(name:String,sk:SFSlice,rel:SFAnchor):SFAnchor
+relative(name:String):SFAnchor
+post():void
+postln():void

```

Diagram relationships:

- SFSlice has a +parent (SFSlice) and #children (SFSlice).
- SFSlice has a #ownerSlice (SFSlice) and #anchors (SFAnchor).
- SFSlice has a +master slice (SFSlice) with +cues (ordered).
- SFSlice has a +start (SFAnchor), +end (SFAnchor), and +current (SFAnchor).
- SFAnchor has a +relativeTo (SFSlice) and a +reAnchor (SFSlice).

Operation

Properties Style Documentation SourceCode Constraints Tagged Values C++ Properties

Language Java

```

//
initMaster { arg wave; // type Object          /** lock-end */
    if ( (not (wave.isKindOf(SF3))) ), {
        // filename has been passed as string, load sample now
        wave = SF3(wave, 1000); // type SFSlice
    };
    name = "master";
    master = wave;
    // implicitly created most outer-bounds master ankers
    start = SFAnker.new("master-start", this, nil);
    start.time_( SFTIME.new(start).rel(0.0) );
    end = SFAnker.new("master-end", this, nil).time_( SFTIME.new(start).rel(1.0) );
    current = SFAnker.new("master-current", this, start).time_( SFTIME.new(start).rel(0.0) );
}

```

Birdview

75%

7:1 Einfg

The screenshot displays the sc3 - Poseidon for UML Professional Edition IDE. The main window shows a UML class diagram with two classes: SFSlice and SFAnchor. SFSlice has attributes like name, childrenByName, and methods like newMaster, initMaster, length, etc. SFAnchor has attributes like name and methods like new, relative, etc. A red circle highlights the 'initMaster' method in the code editor, which is linked to the 'initMaster' method in the class diagram.

**UML Class Diagram:**

- SFSlice Class:**
  - Attributes: +name String, +childrenByName:Dictionary, +ankersByName:Dictionary, #padCount:int=0, #lastSlice:SFSlice
  - Operations: +new():SFSlice, +newMaster(wave:Object):SFSlice, +initMaster(wave:Object):void, #init(parnt:SFSlice,n:String,strt:SFAnchor,en:SFAnchor):void, +length():SFTIME, +slice(pos:String,time:Object):SFSlice, +pad(length:Object):SFSlice, +reset():SFSlice, +position(pos:Object):SFSlice, +inner(i:int):SFSlice, +outer(i:int):SFSlice, +get(name:String):SFSlice, +children():Array, +children\_(newChildren:Array):void, +anker(name:String,time:Object,relativeTo:SFAnchor):SFSlice, +ankers():Array, +ankers\_(newAnkers:Array):void, +sliceTo(name:String,endTime:Object):SFSlice, +sliceRange(name:String,startTime:Object,endTime:Object):SFSlice, +masterslice():SFSlice, +post(level:int):void
- SFAnchor Class:**
  - Attributes: +name String
  - Operations: +new(name:String,slice:SFSlice,relativeTo:SFAnchor):SFAnchor, +init(name:String,sk:SFSlice,rel:SFAnchor):SFAnchor, +relative(name:String):SFAnchor, +post():void, +postln():void

**Java Source Code:**

```

initMaster { arg wave; // type Object          /** lock-end */
    if ( (not (wave.isKindOf(SF3))) ), {
        // filename has been passed as string, load sample now
        wave = SF3(wave, 1000); // type SFSlice
    };
    name = "master";
    master = wave;
    // implicitly created most outer-bounds master ankers
    start = SFAnker.new("master-start", this, nil);
    start.time_( SFTIME.new(start).rel(0.0) );
    end = SFAnker.new("master-end", this, nil).time_( SFTIME.new(start).rel(1.0) );
    current = SFAnker.new("master-current", this, start).time_( SFTIME.new(start).rel(0.0) );
  
```



# Example: Generated Code

```
/*
 * SuperCollider3 source file "SFSlice.sc"
 * Written by Jens Gulden, jgulden@cs.tu-berlin.de.
 * Licensed under the GNU General Public License (GPL),
 * this software comes with NO WARRANTY.
 */

// --- class SFSlice -----
//
SFSlice {
  // --- attributes
  var <>name; // type String
  var <>childrenByName; // type Dictionary
  var <>ankersByName; // type Dictionary
  var padCount = 0; // type int
  var lastSlice; // type SFSlice

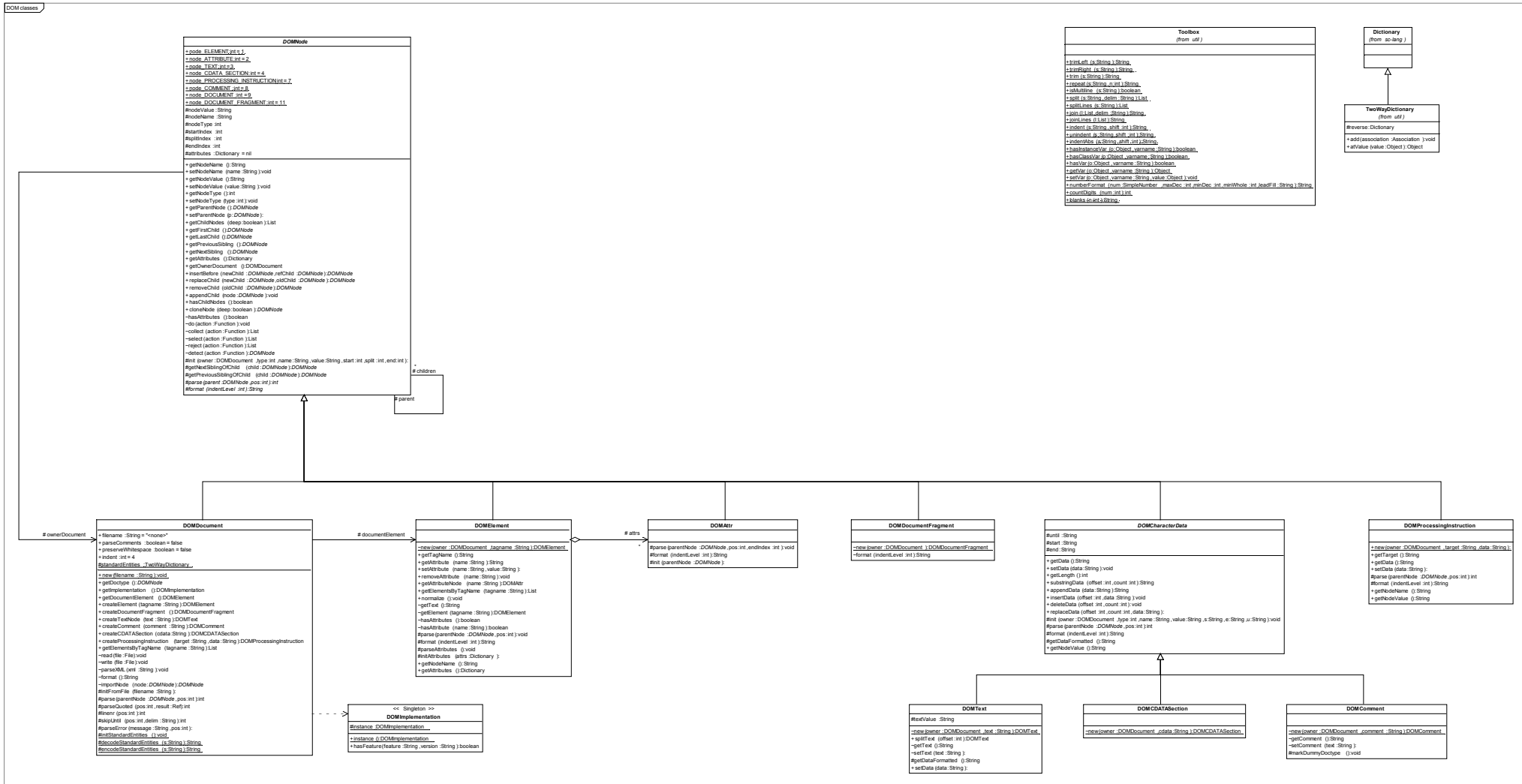
  // --- relationships
  var <>master; // 0..1-relation to type SF3
  var <>start; // 0..1-relation to type SFAnker
  var <>end; // 0..1-relation to type SFAnker
  var <>parent; // 0..1-relation to type SFSlice
  var children; // 0..*-relation to type SFSlice
  var ankers; // 0..1-relation to type SFAnker
  var <>current; // 0..1-relation to type SFAnker

  // --- new() : SFSlice-----
  //
  *new {
    ^super.new.ankersByName_(Dictionary.new).childrenByName_(Dictionary.new);
  } // end new

  ...
}
```



# Example: XML DOM Implementation

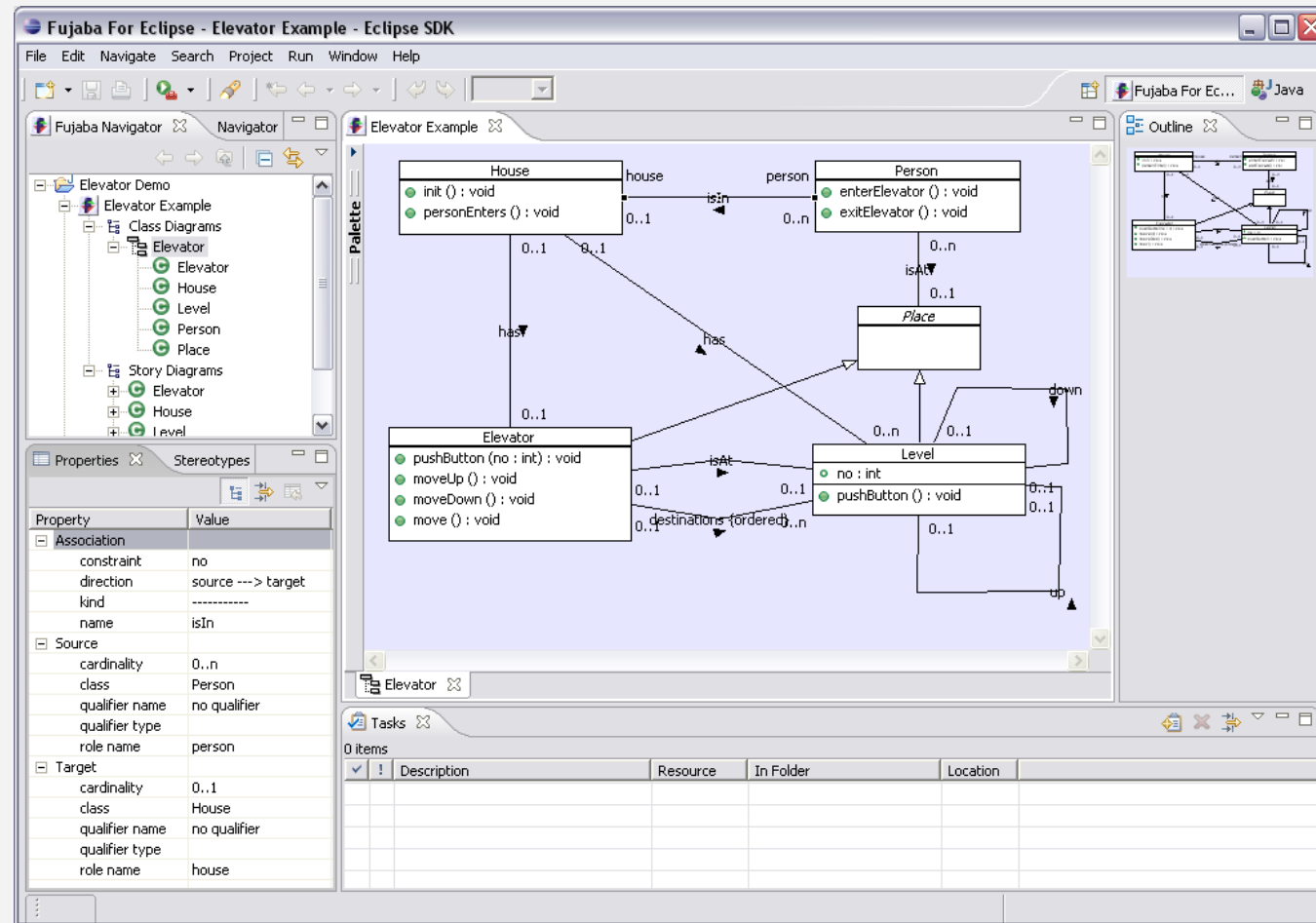




# Future Plans

- Migration to a **free** UML-tool, maybe Fujaba

[www.fujaba.de](http://www.fujaba.de)





# Conclusion

- *Visual modeling* techniques are **practically useful**, adoptable to **non-mainstream** languages
- *SuperCollider* is equipped with all necessary constructs to adopt **state-of-the-art** development techniques



# Thank You!

# Questions?

Jens Gulden, [jgulden@cs.tu-berlin.de](mailto:jgulden@cs.tu-berlin.de)

<http://swiki.hfbk-hamburg.de/MusicTechnology/751>

The sc-cube logo originates from [http://snowofbutterflies.com/m/sc\\_cube/](http://snowofbutterflies.com/m/sc_cube/).

Linux Audio Conference LAC 2007, March 24 – Jens Gulden, [jgulden@cs.tu-berlin.de](mailto:jgulden@cs.tu-berlin.de)