Component-based modelling within GroIMP/XL
– Towards a "construction kit" for interactive, visual modelling

Michael Henke, Katarína Smoleňová

Department Ecoinformatics, Biometrics and Forest Growth,
University of Göttingen, Germany

Tutorial and Workshop
"Modelling with GroIMP and XL"
combined with the 5th GroIMP user and developer meeting

Göttingen, 2012-02-28

Scientific modelling

Common definition

"Scientific modelling is the process of generating abstract, conceptual, graphical and/or mathematical models. ... A scientific model can provide a way to read elements easily which have been broken down to a simpler form."

Scientific modelling

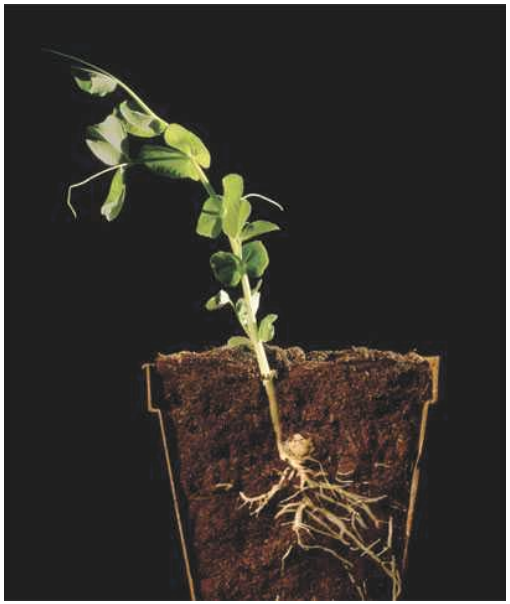
Common definition

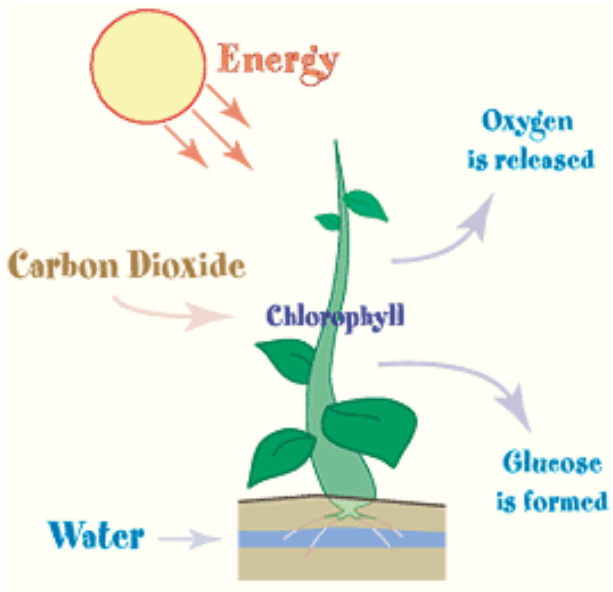
"Scientific modelling is the process of generating abstract, conceptual, graphical and/or mathematical models. . . . A scientific model can provide a way to **read elements easily** which have been **broken down to a simpler form**."

Scientific modelling

Common definition

"Scientific modelling is the process of generating abstract, conceptual, graphical and/or mathematical models. . . . A scientific model can provide a way to **read elements easily** which have been **broken down to a simpler form**."

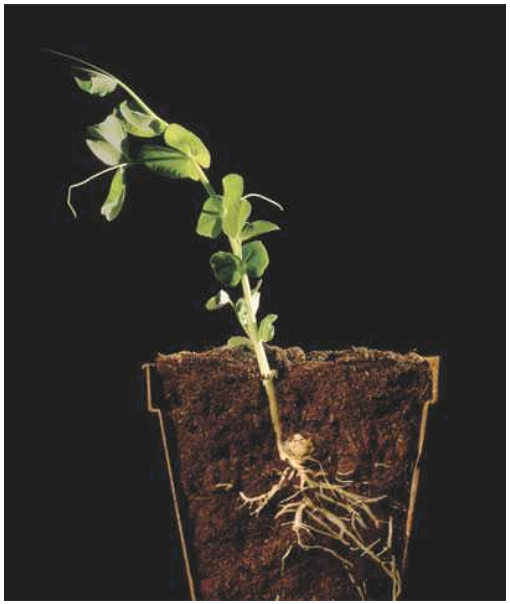


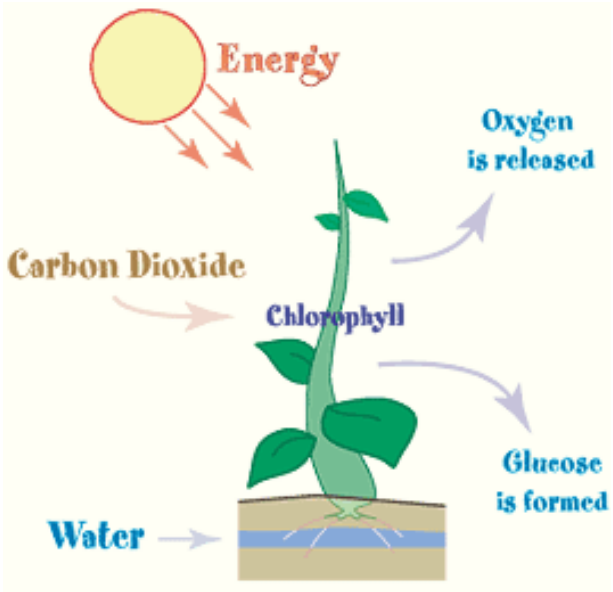


Scientific modelling

Common definition

"Scientific modelling is the process of generating abstract, conceptual, graphical and/or mathematical models. . . . A scientific model **can** provide a way to **read elements easily** which have been **broken down to a simpler form**."





Messy code

SourceFormatX 2.50 Professional - [Example.java]

File Edit Search Tools Configure Window Help

Example.java

```
1  /* Messy Source Example for Java */
2  package cryptix.examples;import
3  java.io.FileInputStream;import java.io.IOException;import cryptix.security.MD5;
4  /** This is a demo of how to use the MD5 or SHA1 classes for hashing data **/
5  public final class MD5AFile{private static final int BUF_LENGTH=1024;public
6  static void main(String argv[]){if(argv.length!=1)System.err.println(
7  "usage java MD5Afile filename");else try{printHash(doHash(argv[0]));}catch(
8  IOException ioe){System.err.println(
9  "There has been an IO exception to the file was not hashed.");
10 ioe.printStackTrace();}}private static void printHash(byte buf[]){
11 System.out.println("hash of file is;");System.out.print("MD5 : ");for(int i=0,j
12 =buf.length;j>0;i++,j--){int val=(int)buf[i];System.out.print(Integer.toString(
13 (val>>4)&0xF,16));System.out.print(Integer.toString(val&0xF,16));}
14 System.out.println();}}
```

```
1  /* Messy Source Example for Java */
2  package cryptix.examples;
3  import java.io.FileInputStream;
4  import java.io.IOException;
5  import cryptix.security.MD5;
6  /** This is a demo of how to use the MD5 ... for hashing data **/
7  public final class MD5AFile
8  {
9    private static final int BUF_LENGTH = 1
10   public static void main(S
11   {
12     if (argv.length != 1)
13       System.err.println("u
14     else
```

Cut
Copy
Paste
Delete
Format

Export the current document as HTML file wi

```
void CMymfc30bView::OnLButtonDown(UINT nFlags, CPoint point)
{
    // TODO: Add your message handler code here and/or call default
    CMymfc30bDoc* pDoc = GetDocument();
    if(m_tracker.HitTest(point) == CRectTracker::hitMiddle) {
        COleDataSource* pSource = SaveDib();
        if(pSource) {
            // DoDragDrop returns only after drop is complete
            CClientDC dc(this);
            OnPrepareDC(&dc);
            CPoint topleft = m_rectTracker.TopLeft();
            dc.LPtoDP(&topleft);
            // 'point' here is not the same as the point parameter in
            //  OnDragEnter, so we use this one to compute the offset
            m_dragOffset = point - topleft;  // device coordinates
            pDoc->m_bDragHere = TRUE;
            DROPEFFECT dropEffect = pSource->DoDragDrop(
                DROPEFFECT_MOVE|DROPEFFECT_COPY, CRect(0, 0, 0, 0));
            TRACE("after DoDragDrop -- dropEffect = %ld\n", dropEffect);
            if (dropEffect == DROPEFFECT_MOVE && pDoc->m_bDragHere) {
                pDoc->OnEditClearAll();
            }
            pDoc->m_bDragHere = FALSE;
            delete pSource;
        }
    }
```

```
.Track(this, point, FALSE, NULL)) {
C dc(this);
eDC(&dc);
d have some way to prevent it going out of bounds
acker = m_tracker.m_rect;
P(m_rectTracker); // Update logical coords
```

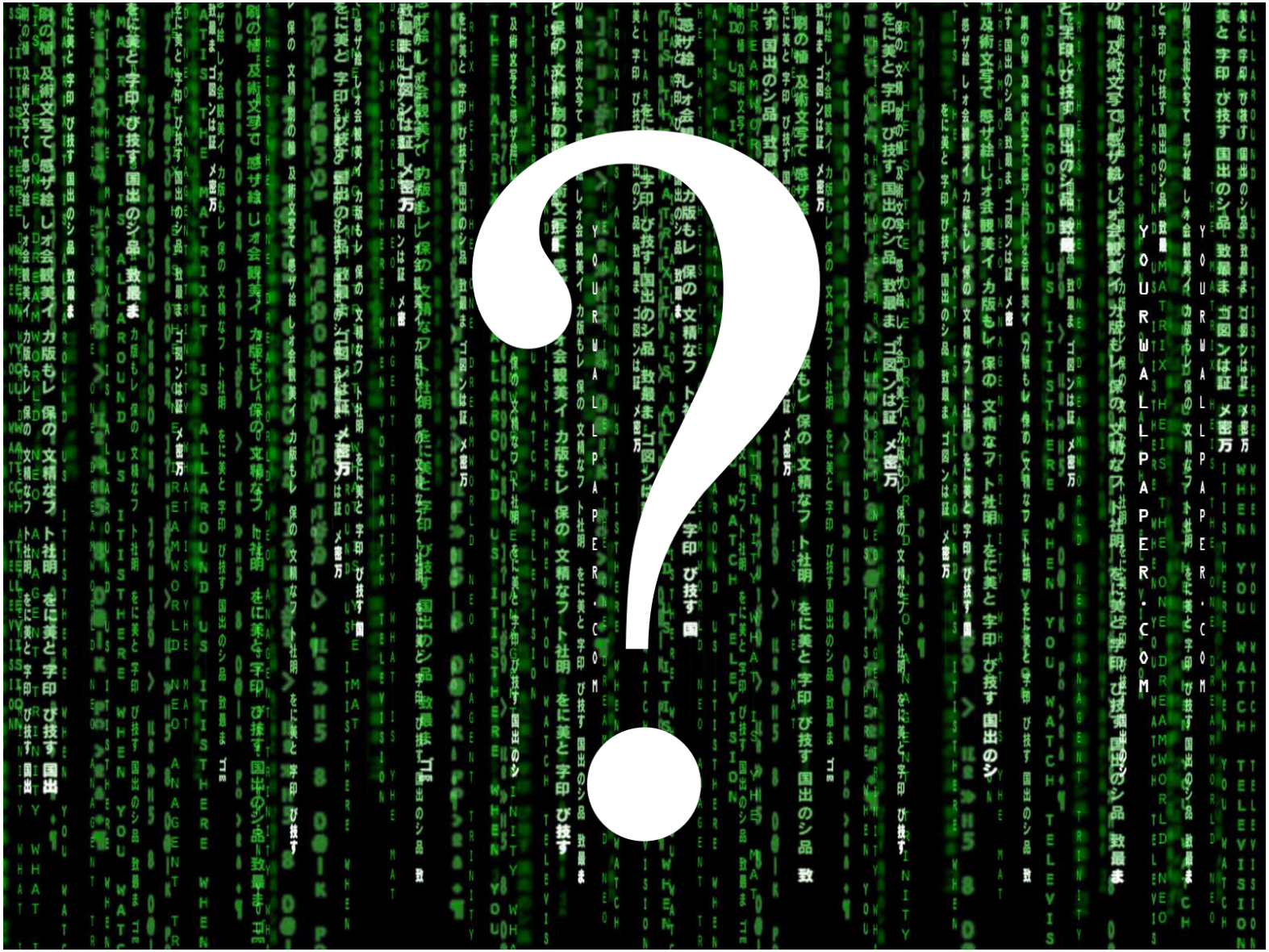
```
00221 Public Function GetChecksum(ByVal sentence As String) As String
00222     Dim Character As Char
00223     Dim Checksum As Integer
00224     For Each Character In sentence
00225         Select Case Character
00226             Case "$"c
00227                 ' Ignore the dollar sign
00228             Case "*"c
00229                 ' Stop processing before the asterisk
00230                 Exit For
00231             Case Else
00232                 ' Is this the first value for the checksum?
00233                 If Checksum = 0 Then
00234                     'Yes. Set the checksum to the value
00235                     Checksum = Convert.ToByte(Character)
00236                 Else
00237                     Checksum = Checksum Xor Convert.ToByte(Character)
00238                 End If
00239         End Select
00240     Next
00241     Return Checksum.ToString("X2")
00242 End Function
```

Salesforce.com Integration Error

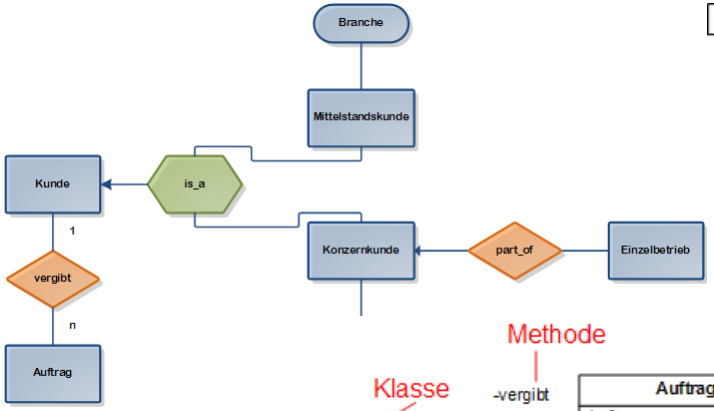
```
Fault Code (0). java.lang.NullPointerException
    at common.udd.object.XmlRpcEntityDescribe.addFields(XmlRpcEntityDescribe.java:515)
    at common.udd.object.XmlRpcEntityDescribe.getDescribe(XmlRpcEntityDescribe.java:75)
    at common.udd.object.EntityObject.getXmlRpcDescribe(EntityObject.java:4137)
    at common.api.xmlrpc.XmlRpcDispatcher.innerDispatch(XmlRpcDispatcher.java:396)
    at common.api.xmlrpc.XmlRpcDispatcher.dispatch(XmlRpcDispatcher.java:280)
    at common.api.xmlrpc.XmlRpcDispatcher.innerExecute(XmlRpcDispatcher.java:255)
    at common.api.xmlrpc.XmlRpcDispatcher.execute(XmlRpcDispatcher.java:118)
    at helma.xmlrpc.XmlRpcServer$Worker.execute(XmlRpcServer.java:161)
    at helma.xmlrpc.XmlRpcServer.execute(XmlRpcServer.java:97)
    at common.api.xmlrpc.Api.doPost(Api.java:253)
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:152)
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:90)
    at com.caucho.server.dispatch.ServletFilterChain.doFilter(ServletFilterChain.java:99)
    at system.filter.PreGzipFilter
```

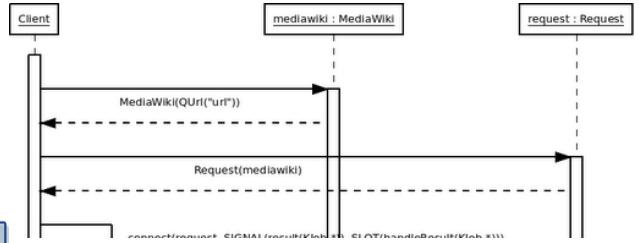
OK

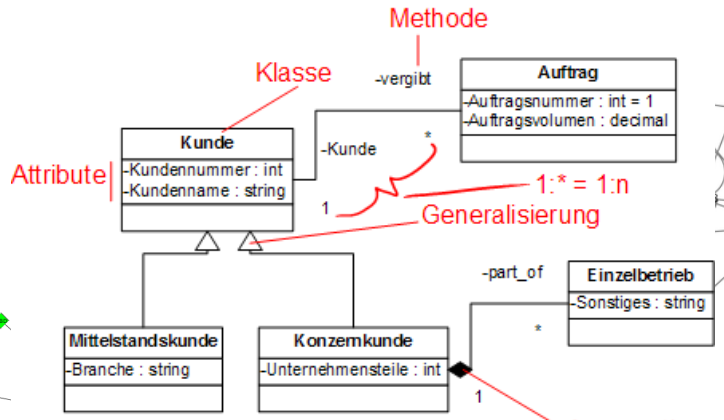
Messy code . . . nobody understands

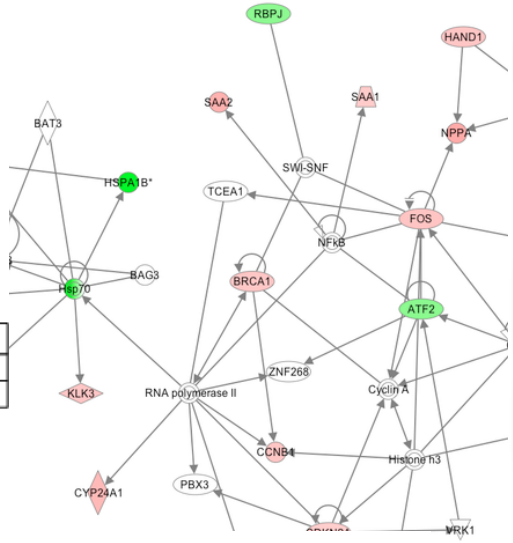


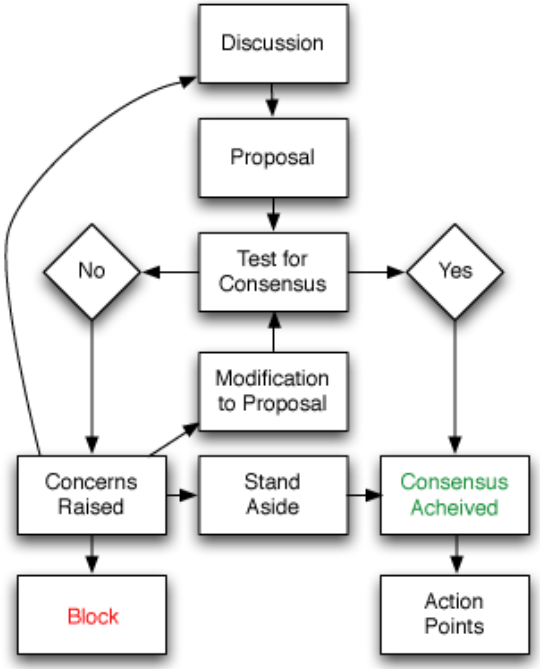
Software engineering tools

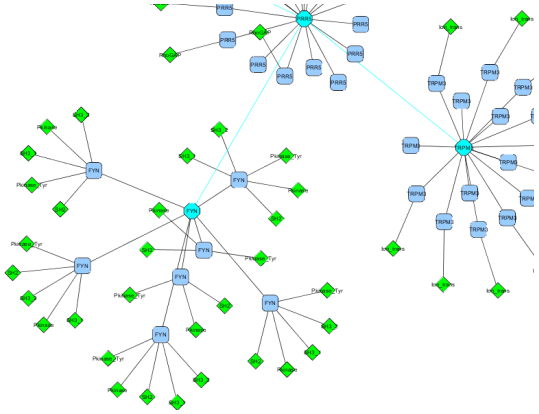


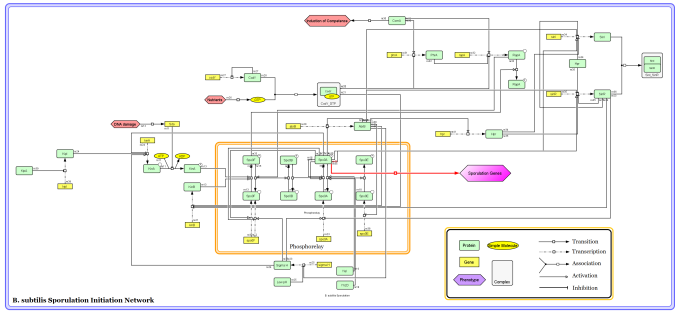


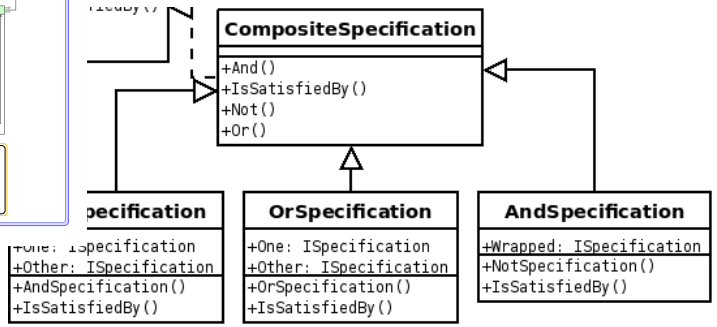












Reasons

- Knowledge is all on the Web
  - distributed
  - very time consuming
- Justifiably sceptical
  - several "great innovations" - without noticeable improvements
- Frustrated by the complexity
  - stick to lowest common denominators

Reasons

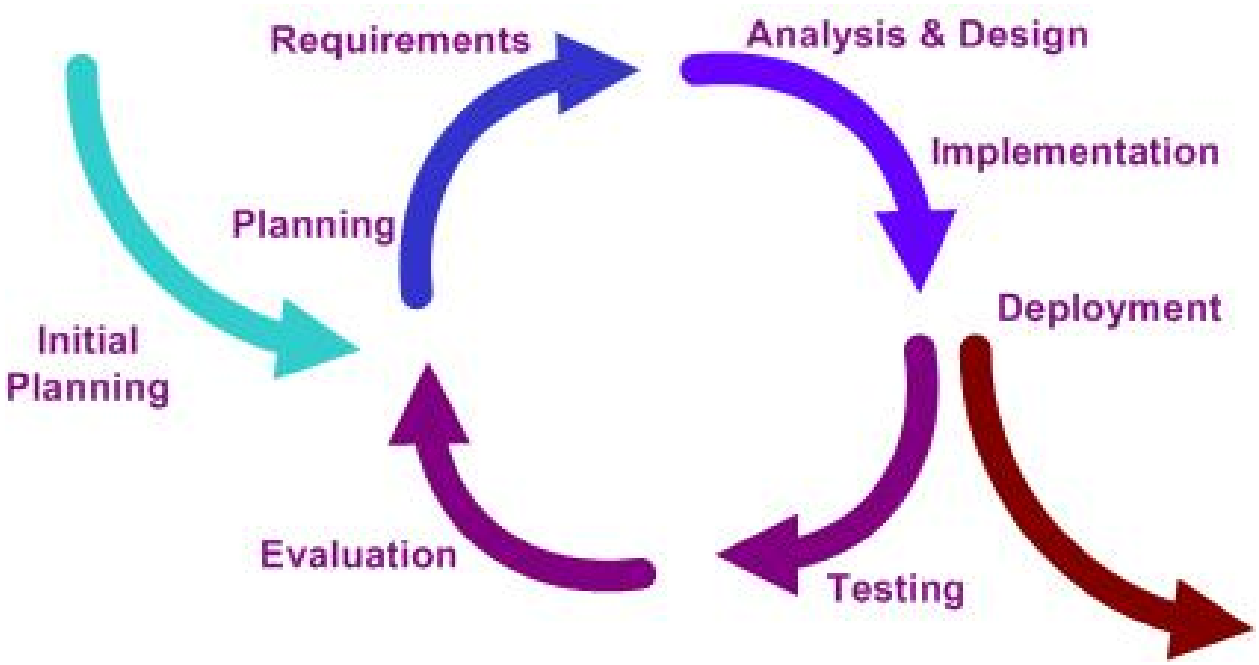
- Knowledge is all on the Web
  - distributed
  - very time consuming
- Justifiably sceptical
  - several “great innovations” - without noticeable improvements
- Frustrated by the complexity
  - stick to lowest common denominators

Reasons

- Knowledge is all on the Web
  - distributed
  - very time consuming
- Justifiably sceptical
  - several “great innovations” - without noticeable improvements
- Frustrated by the complexity
  - stick to lowest common denominators

Modelling process

. . . or "Modeller's treadmill"



$\Longrightarrow$ the **real limit** on what computational scientists can accomplish is **how quickly and reliably** they can **translate their ideas into working code**

Summarize the initial situation

- Modelling is often based on programming
  - Requires programming knowledge / skills
  - → Concentration is divided: modelling vs. programming
- Developer with:
  - No or little programming experience → challenge of programming
  - Little knowledge of biological systems → challenge of modelling
- Common way of modelling:
  - *Ad hoc*
    - Usually starting "from scratch"
    - (initially) no (clear) concept/design
    - Unsystematic
  - Extend/change existing models
    - Getting overview of code more difficult
- No reuse ⟹ "Reinventing the wheel"

Aim

Goals:

- Low entry threshold: make access simpler
- Reuse of software (parts)
  - independently developed
  - development by experts
  - periodic maintenance
- Visual support

Side effects:

- Reduction of time for a model developer to get familiar with a system / language
- Transparent and flexible modelling process and models
- Models that can be evaluated / combined
- Models become comparable
- Enhancement of quality
- Faster model development
- Communication between modeller and experimentator is facilitated

Aim

Goals:

- Low entry threshold: make access simpler
- Reuse of software (parts)
  - independently developed
  - development by experts
  - periodic maintenance
- Visual support

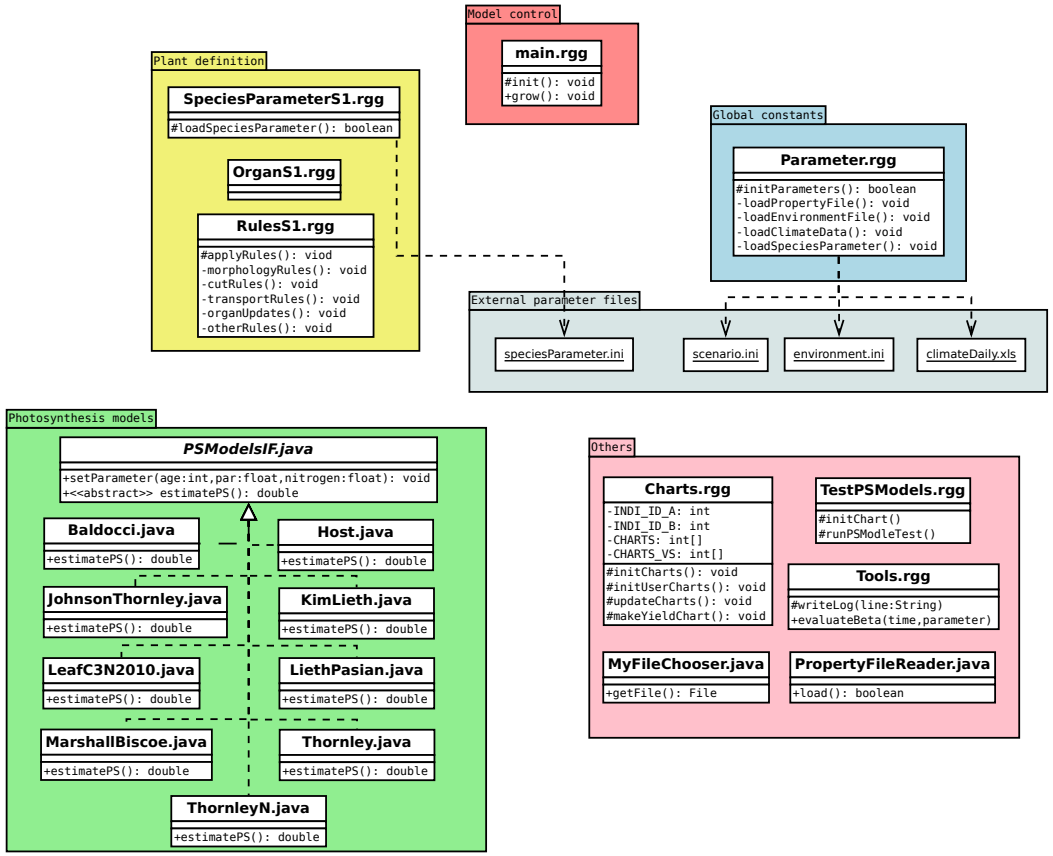
Side effects:

- Reduction of time for a model developer to get familiar with a system / language
- Transparent and flexible modelling process and models
- Models that can be evaluated / combined
- Models become comparable
- Enhancement of quality
- Faster model development
- Communication between modeller and experimentator is facilitated

Current approaches

- Modularization
- Object-orientation

Structure of the FSPM-Prototype[a]:

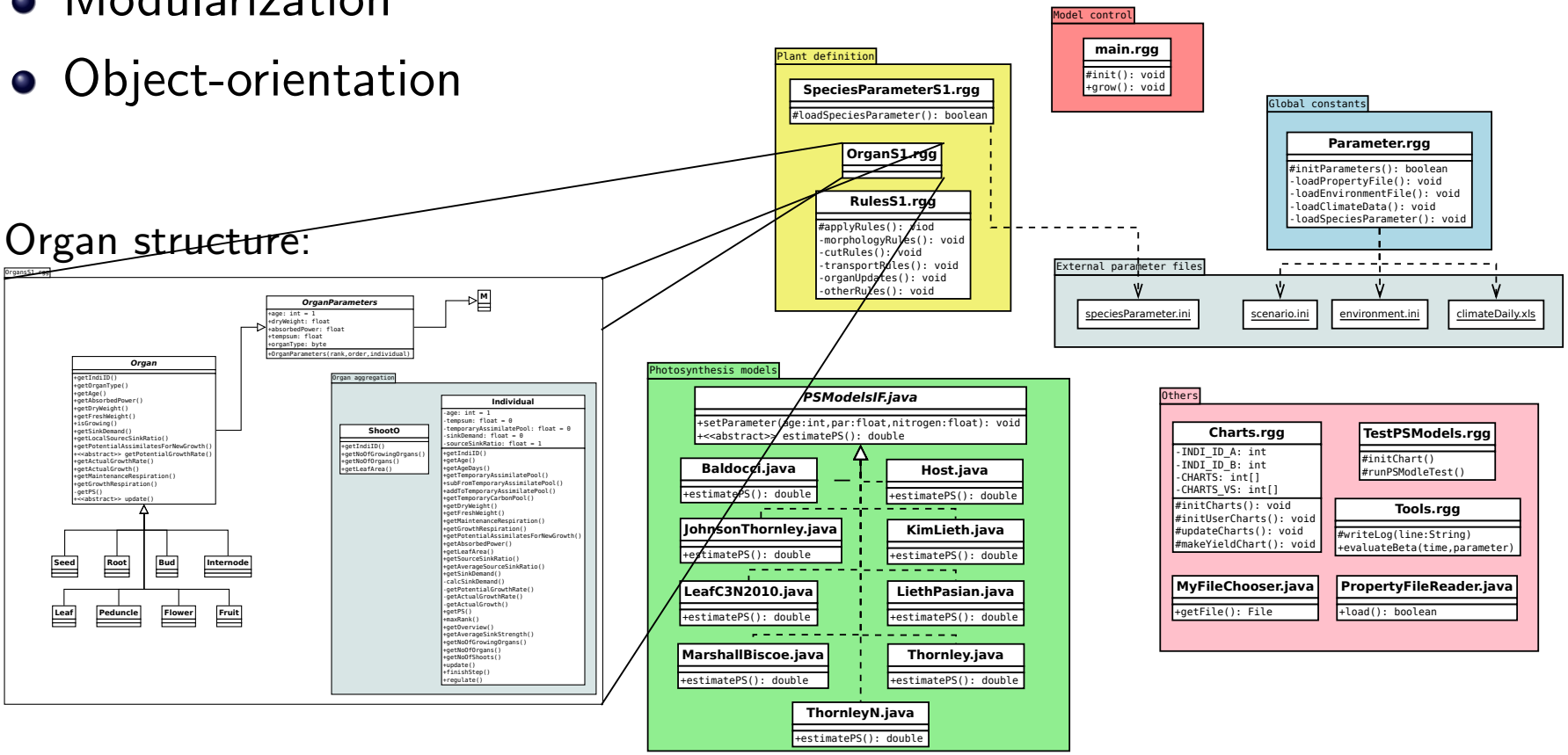


[a] M.Henke and GH. Buck-Sorlin unpubl., 2010

Current approaches

- Modularization
- Object-orientation

Structure of the FSPM-Prototype[a]:



Organ structure:

[a] M.Henke and GH. Buck-Sorlin unpubl., 2010

Component-based engineering

- Well-known engineering technique

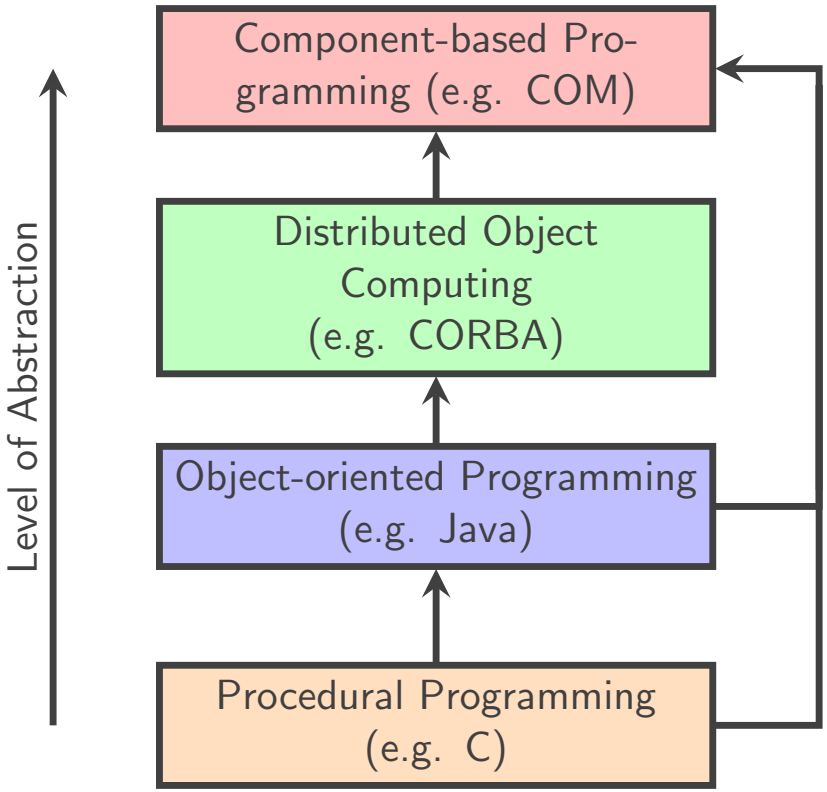
Pre-assembled components

Different, independent subcontractors

Component-based software engineering

- Starts in 1968
- Early 1990s: IBM - System Object Model
- Microsoft: OLE, COM
- Today: many successful software component models exist
  - Microsoft (.NET, COM, DCOM, OLE, ActiveX, COM+)
  - Object Management Group (CORBA)
  - Sun Microsystems (JavaBeans, Servlets, Applets, Enterprise JavaBeans)
  - OSGi Alliance (OSGi)

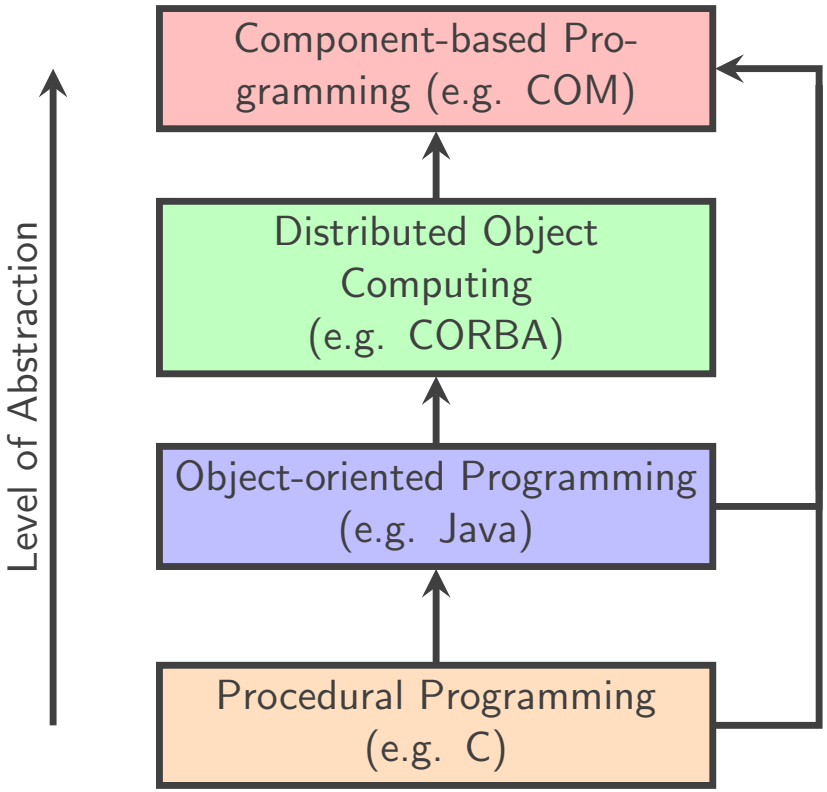
$\Longrightarrow$ dividing SW into re-usable parts



Component-based software engineering

- Starts in 1968
- Early 1990s: IBM - System Object Model
- Microsoft: OLE, COM
- Today: many successful software component models exist
  - Microsoft (.NET, COM, DCOM, OLE, ActiveX, COM+)
  - Object Management Group (CORBA)
  - Sun Microsystems (JavaBeans, Servlets, Applets, Enterprise JavaBeans)
  - OSGi Alliance (OSGi)

$\Longrightarrow$ dividing SW into re-usable parts



Component - Definition

- "A software component is a **unit of composition** with contractually specified **interfaces** and **explicit context dependencies** only. A software component can be **deployed independently** and is subject to **compositions by third parties**." (C. Szyperski)
- "A software component is a **software element**, that can be **connected, used and executed** without changes at any other components according to a **component model** and corresponding to **composition standard**." (W.T. Councill)
- "A small **binary object** or **program** that performs a **specific function** and is designed in such a way to easily **operate with other components** and applications." www.webopedia.com
- "A component consists of diverse **(software-) artefacts**. It is **re-usable**, **closed** and **marketable**, provides **services** via **well-defined interfaces**, **covert her implementation** details and be used in **combination** with other components, ..." (Gesellschaft für Informatik)

Component - Requirements

Basic requirements:

- useful functionality
- strict encapsulation
- re-usability
- a component can provide a self description
- distributed in binary form
- configurable, no persistent state

Additional requirements:

- platform-independent
- location-independent (distributed)
- independent of programming languages
- ready for integration and communication (plug-and-play)
- independent deployment
- GUI aided design

Component - Requirements

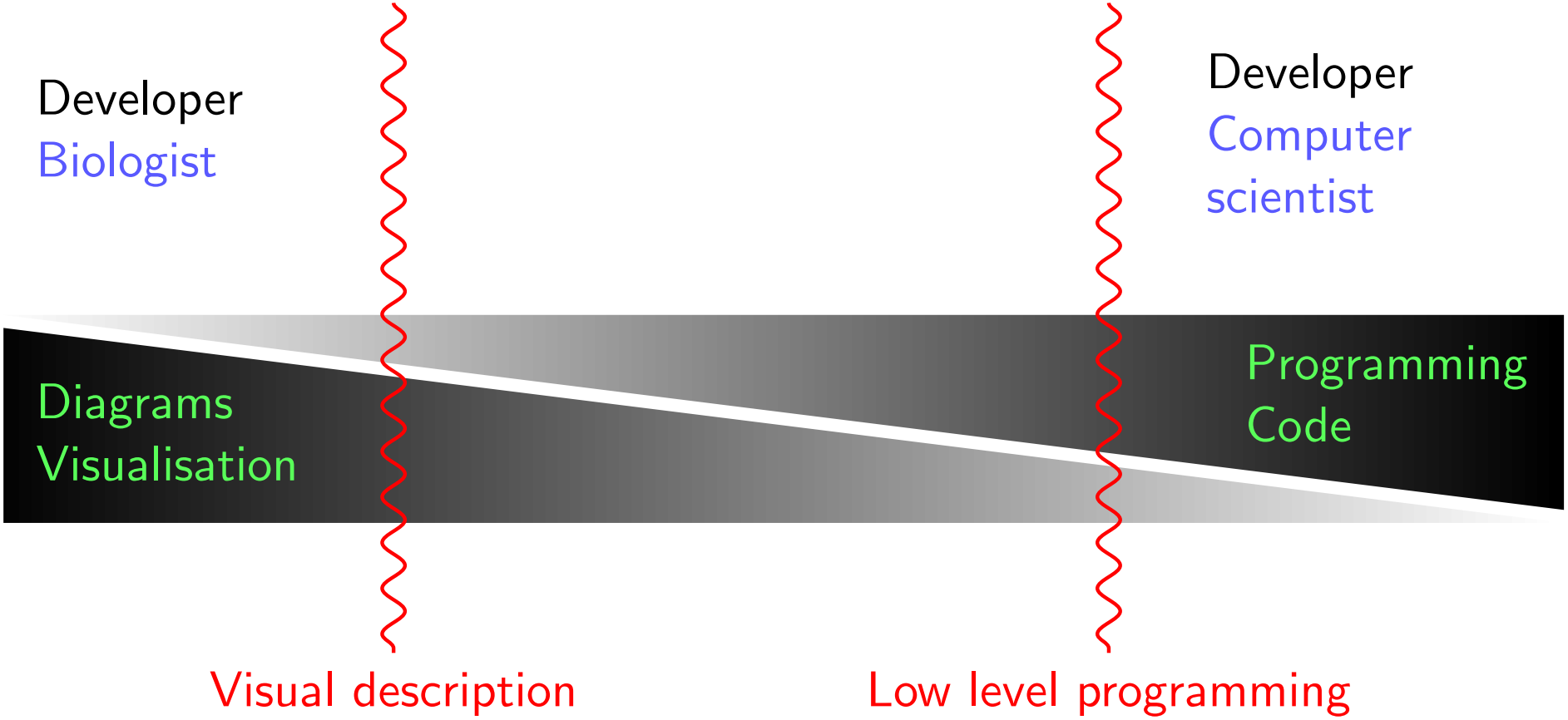
Basic requirements:

- useful functionality
- strict encapsulation
- re-usability
- a component can provide a self description
- distributed in binary form
- configurable, no persistent state

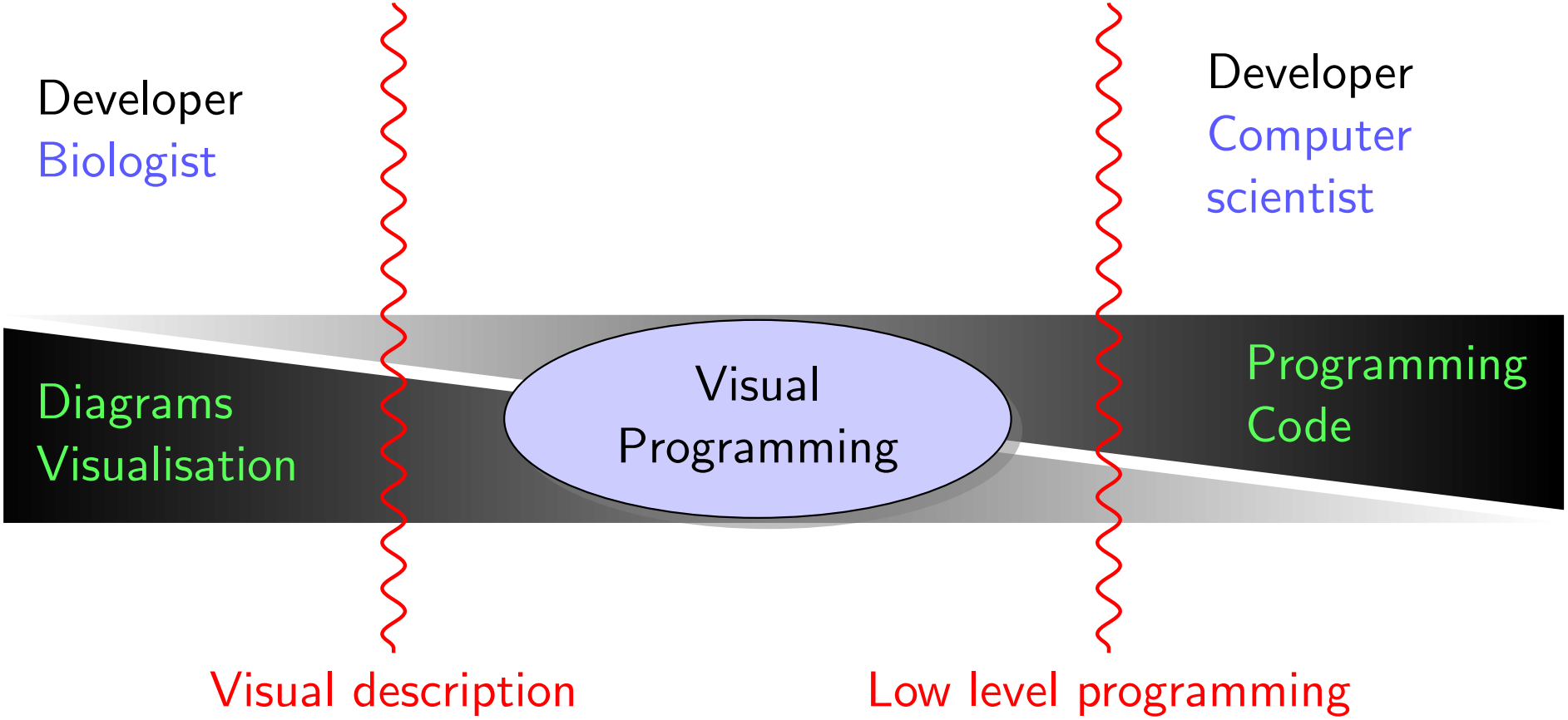
Additional requirements:

- platform-independent
- location-independent (distributed)
- independent of programming languages
- ready for integration and communication (plug-and-play)
- independent deployment
- GUI aided design

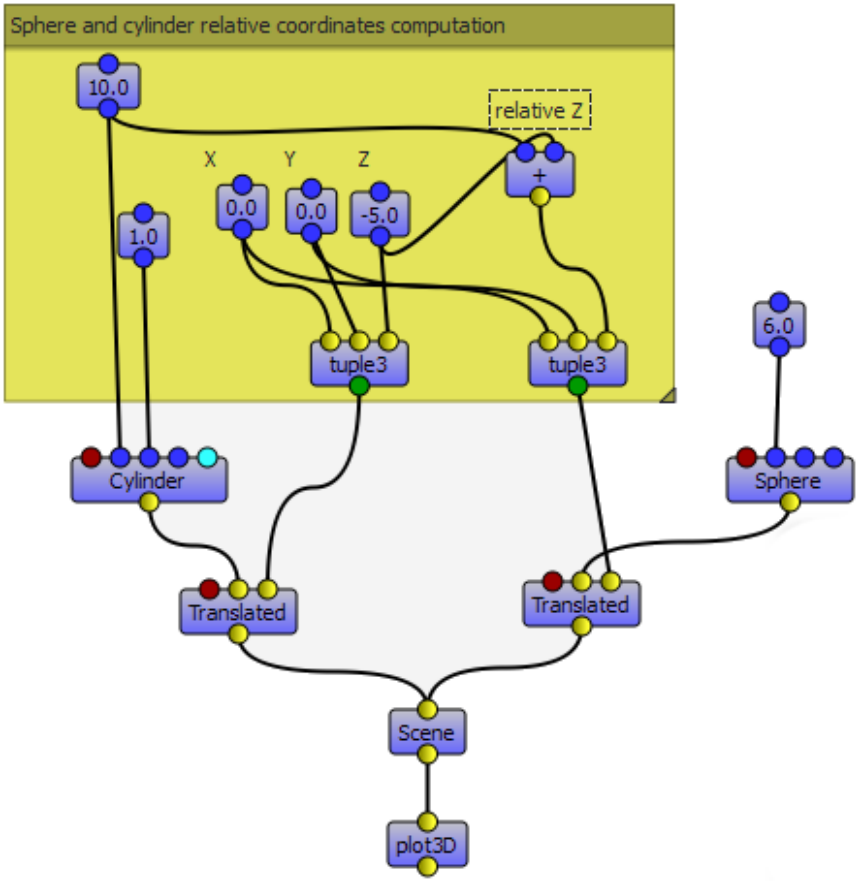
Actors - Tools - Level of Abstraction



Actors - Tools - Level of Abstraction



Visual programming – OpenAlea-like

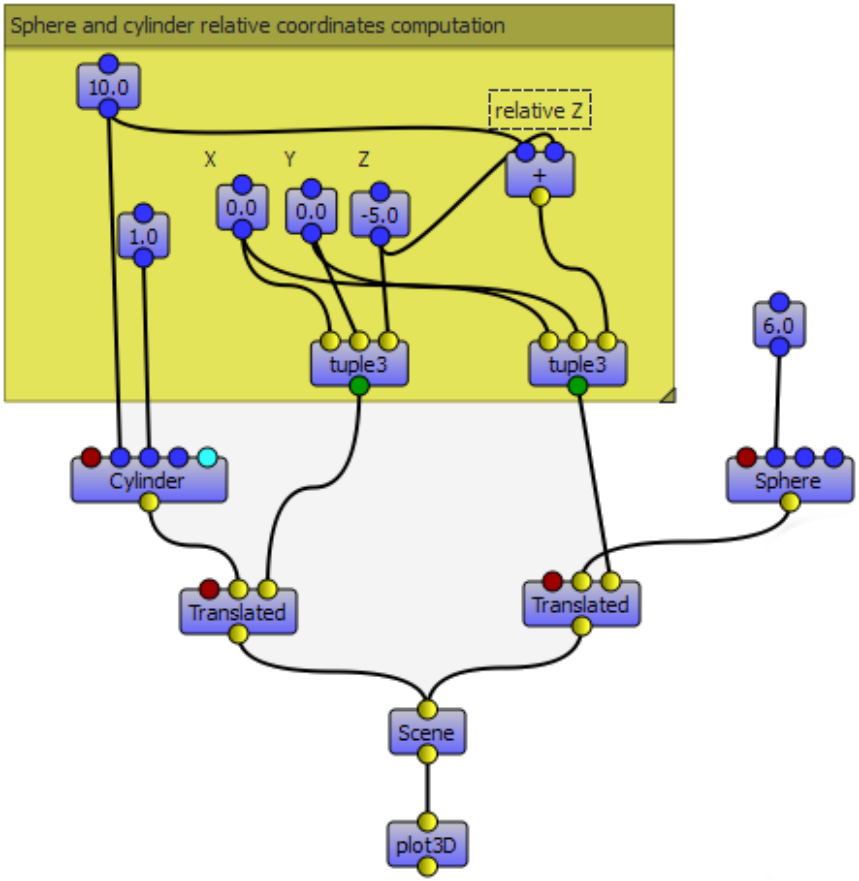


Mathematical calculations:

- atomic elements for numbers and operations
- visual low level programming with a "quasi" one-to-one translation of code to visual objects
- would only shift problems (and add some we did not have previously?!)

$\Longrightarrow$ NO equivalent within VisualGroIMP

Visual programming – OpenAlea-like



Mathematical calculations:

- atomic elements for numbers and operations
- visual low level programming with a "quasi" one-to-one translation of code to visual objects
- would only shift problems (and add some we did not have previously?!)

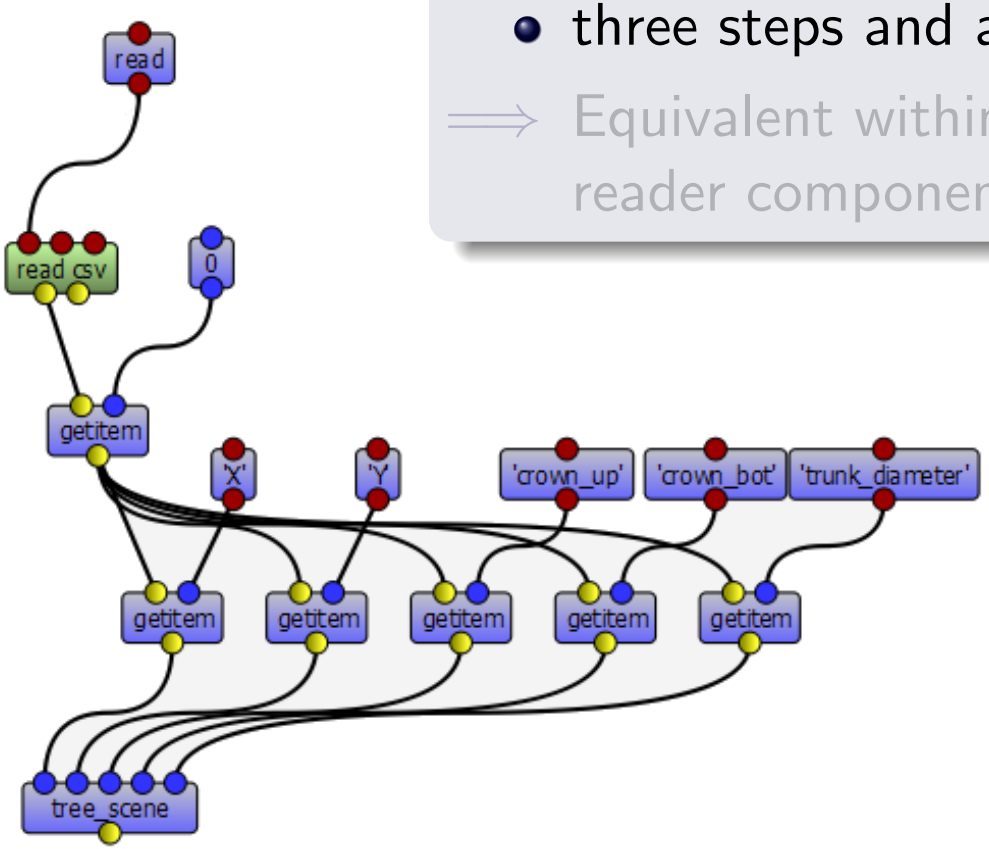
$\Longrightarrow$ NO equivalent within VisualGroIMP

Visual programming – OpenAlea-like

File input:

- three steps and a loop are needed to read a file

⟹ Equivalent within VisualGroIMP: generic file reader component

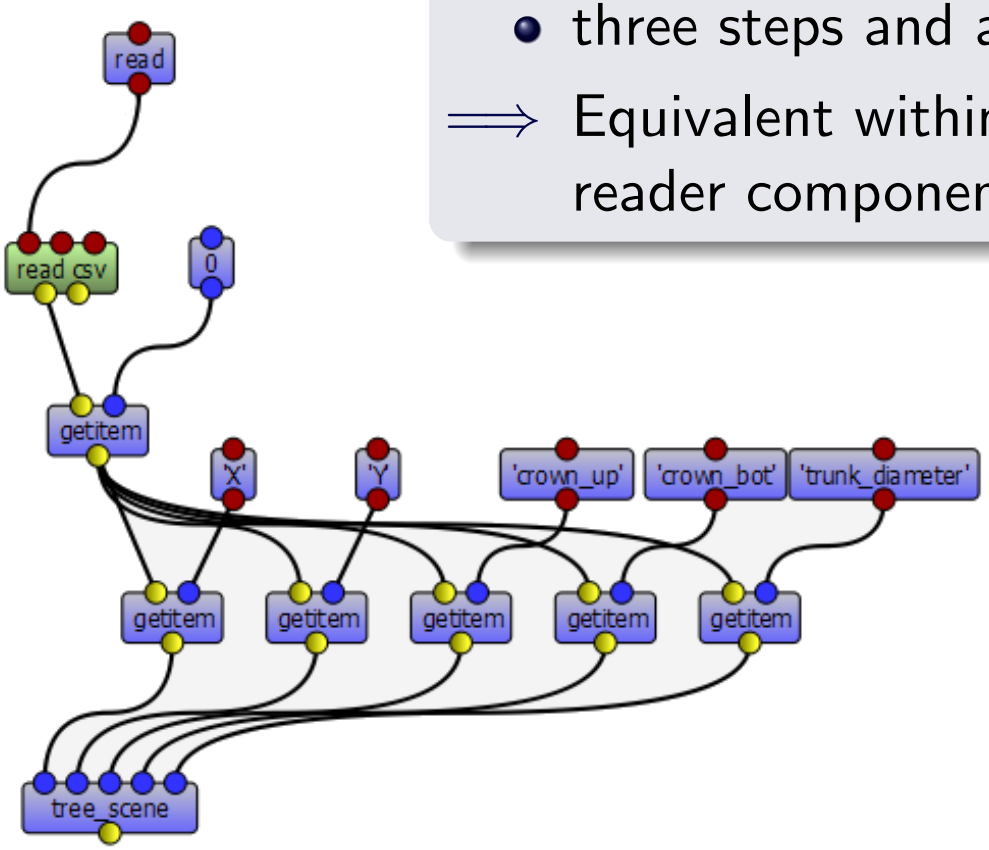


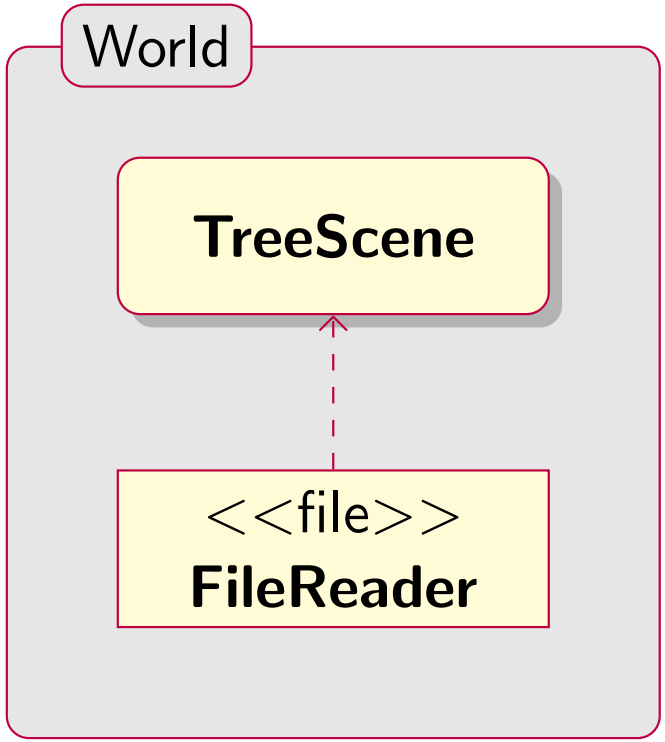
Visual programming – OpenAlea-like

File input:

- three steps and a loop are needed to read a file

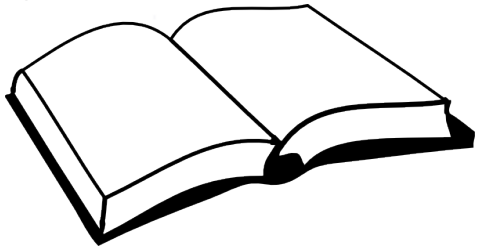
$\Longrightarrow$ Equivalent within VisualGroIMP: generic file reader component



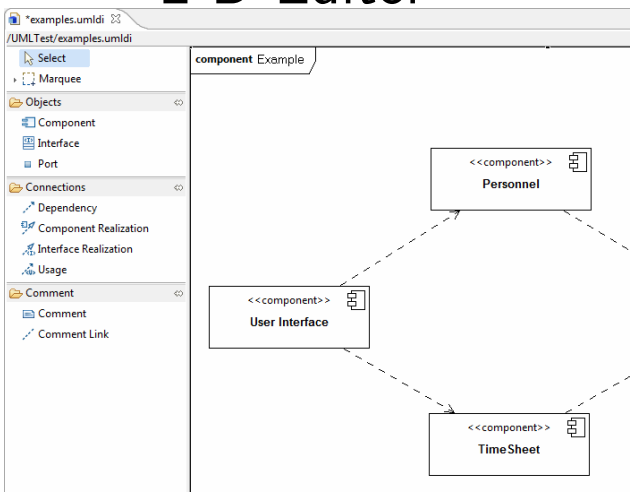


VisualGroIMP - Project

User manual



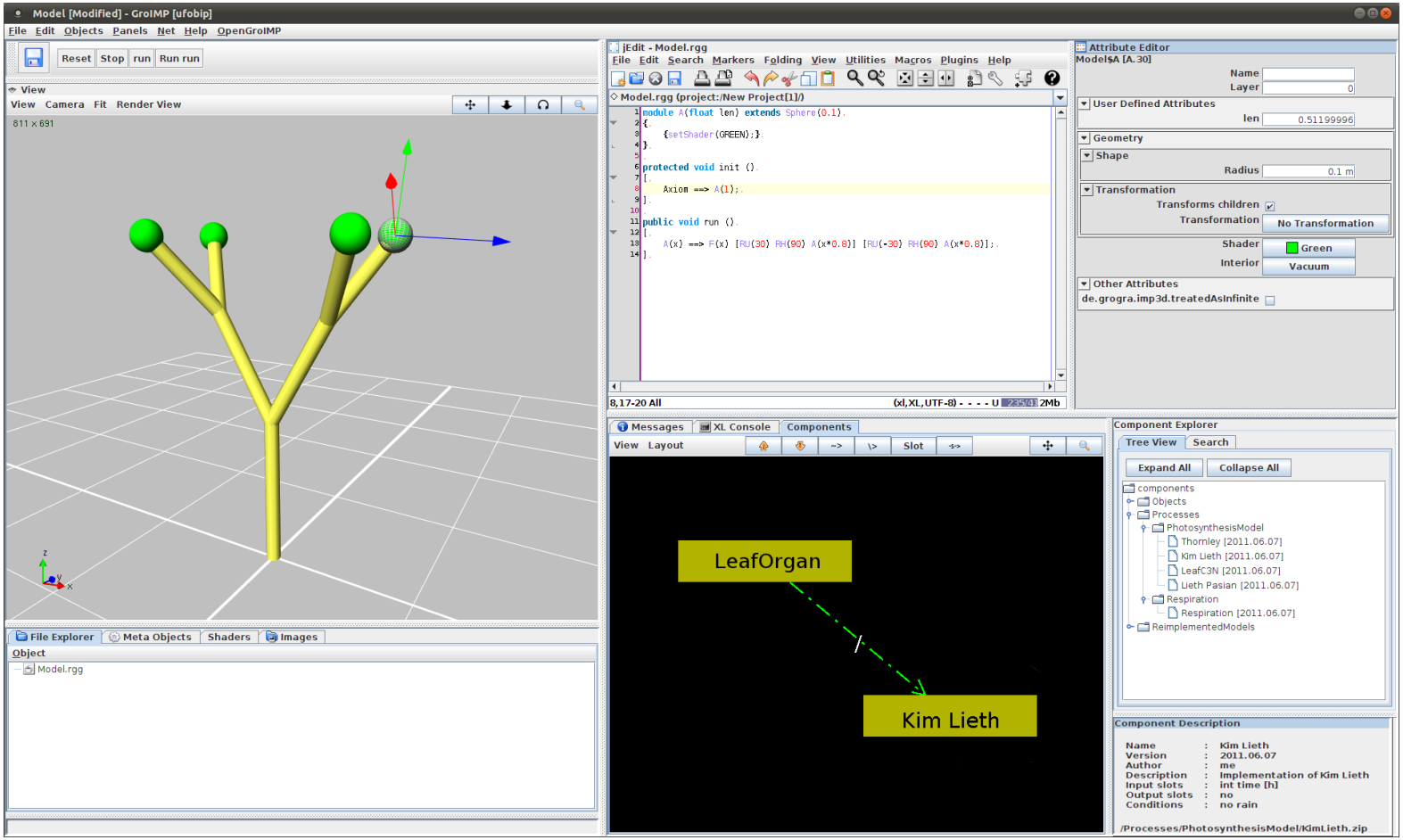
2-D Editor



Library

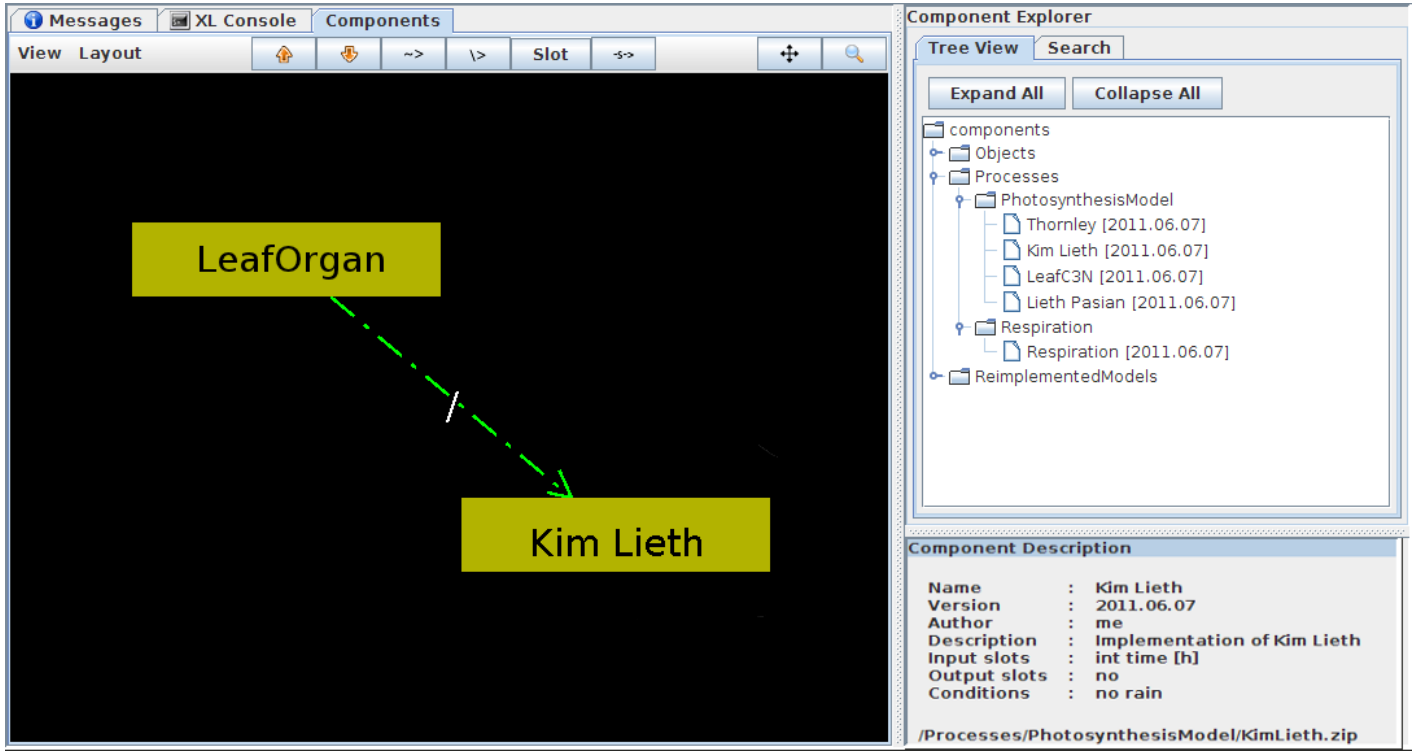
VisualGroIMP - Current state

GroIMP Screenshot



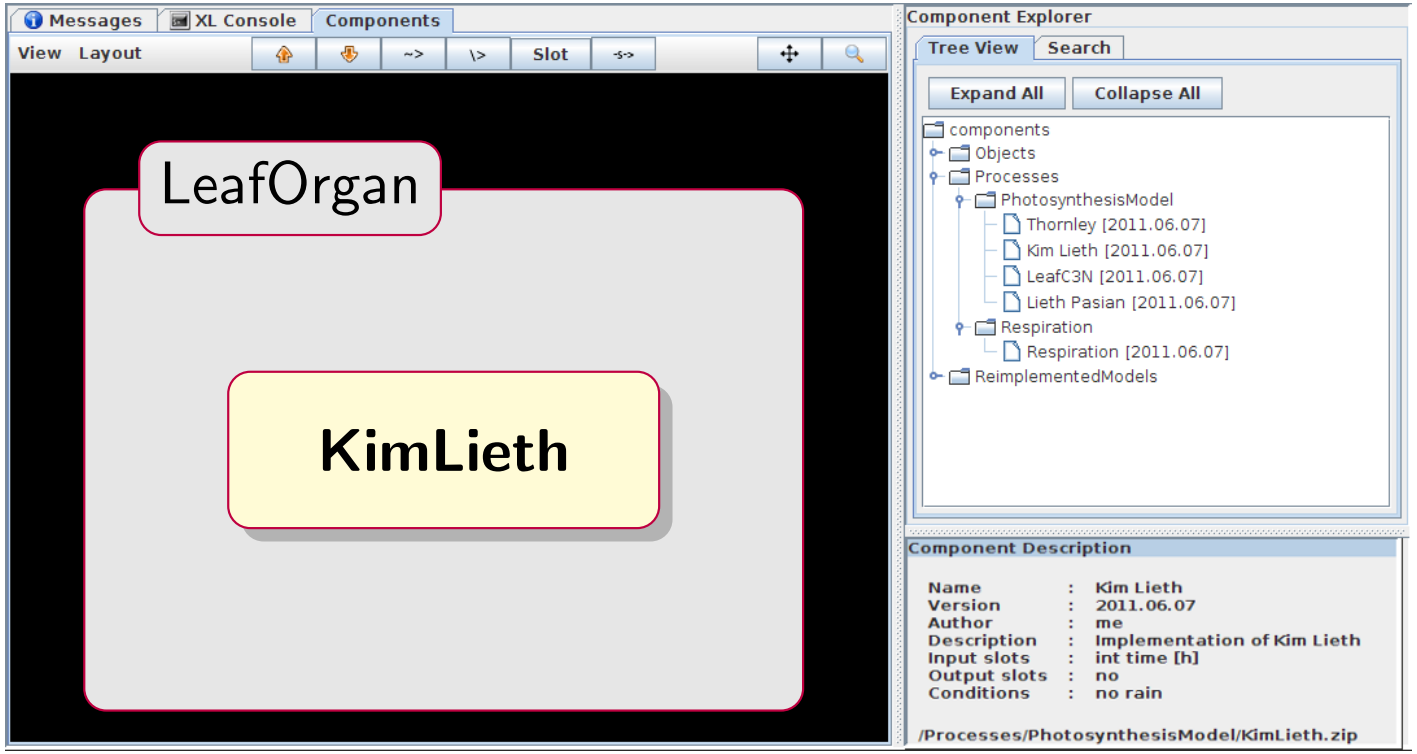
VisualGroIMP - Current state

GroIMP Screenshot

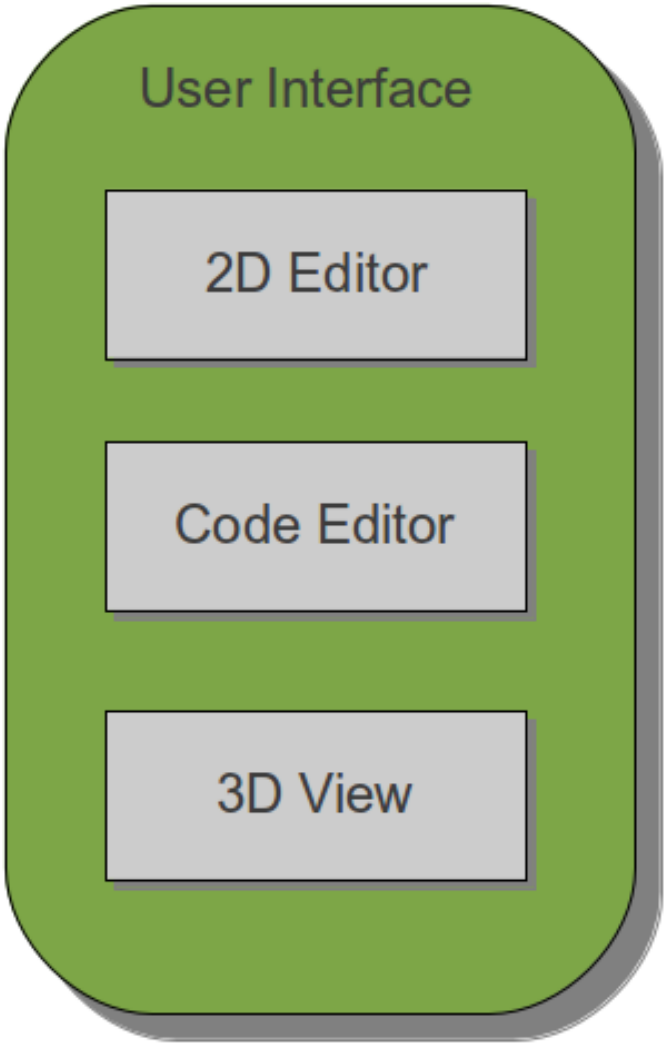


VisualGroIMP - Current state

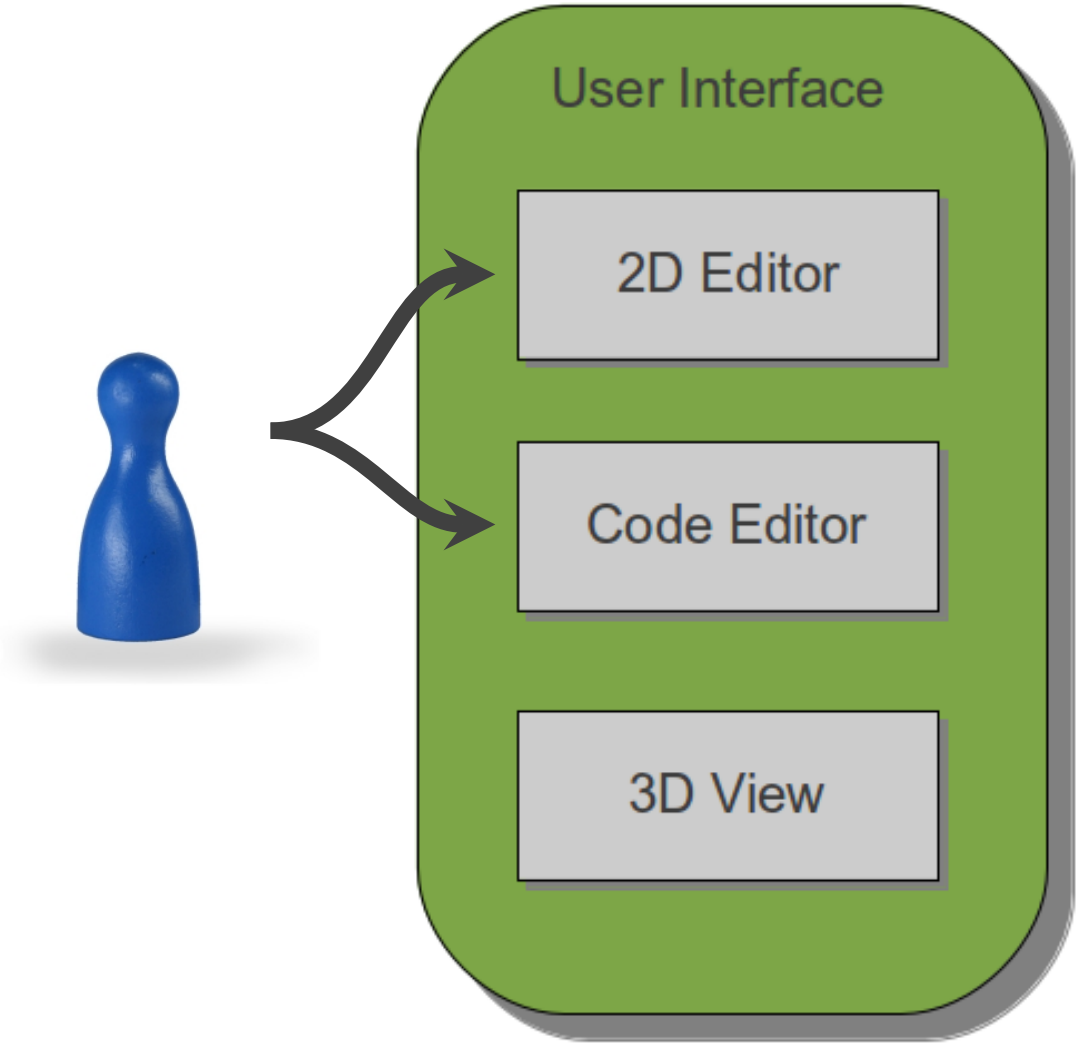
GroIMP Screenshot



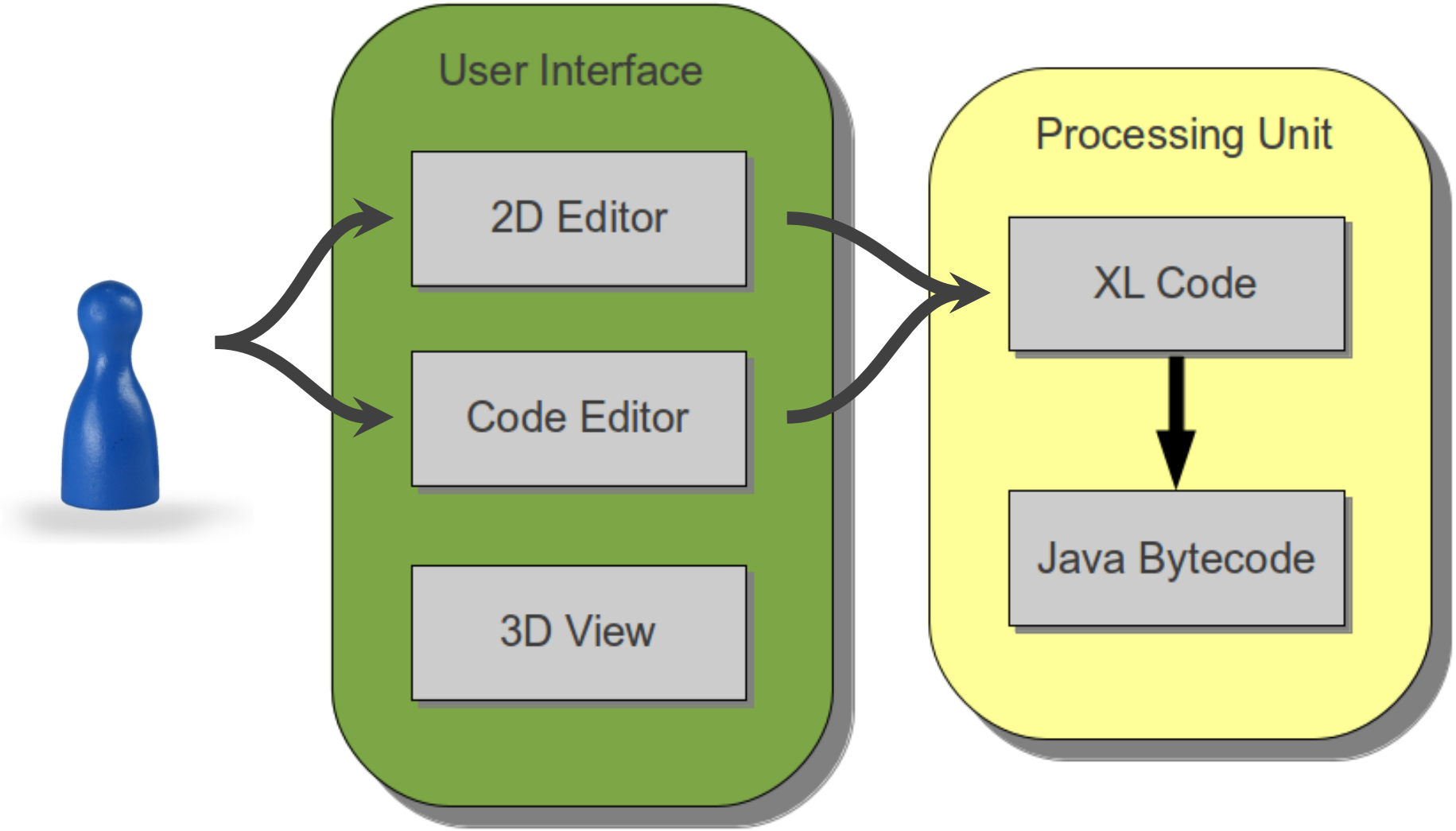
VisualGroIMP - Workflow



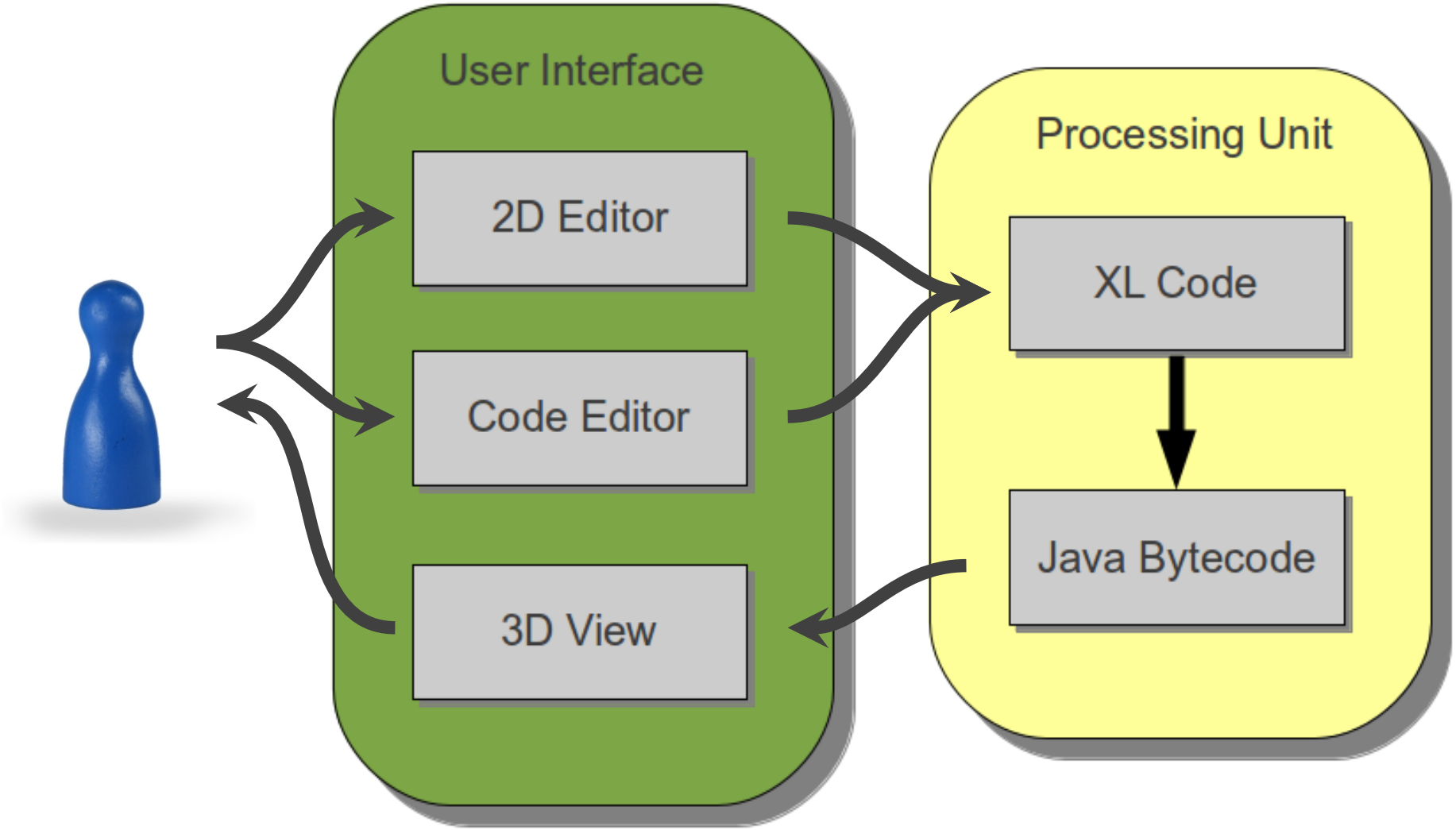
VisualGroIMP - Workflow



VisualGroIMP - Workflow



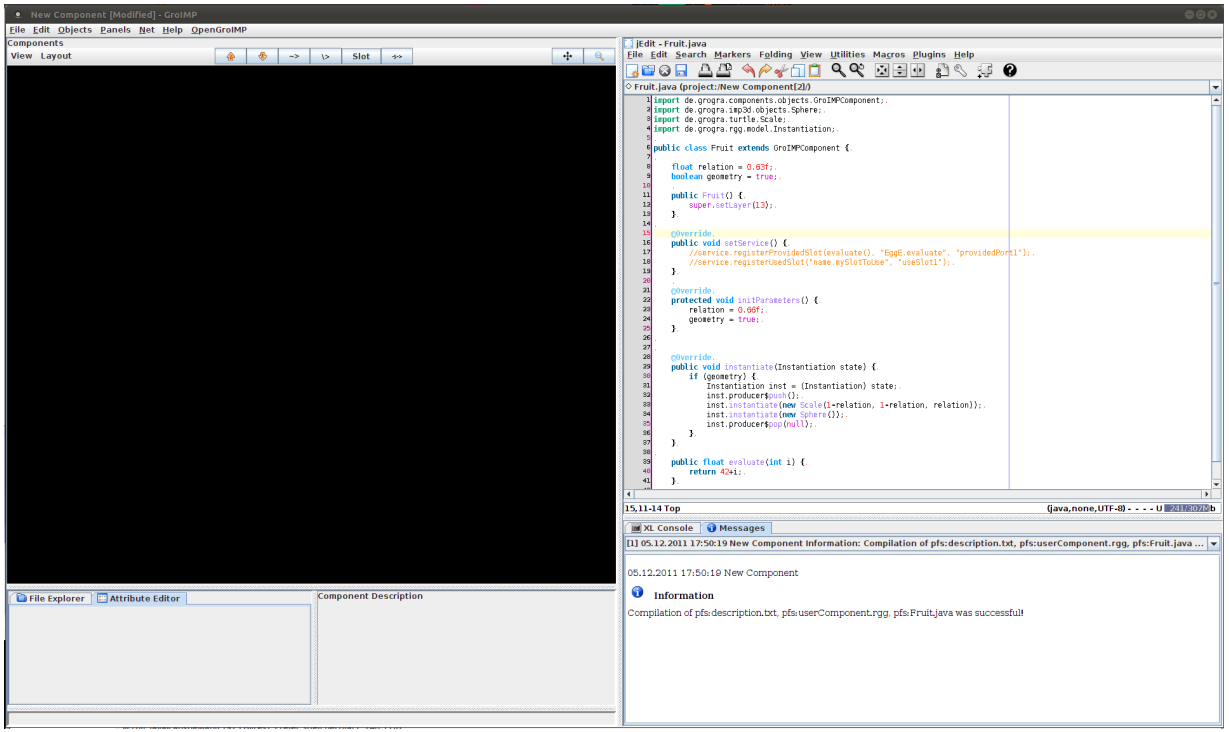
VisualGroIMP - Workflow



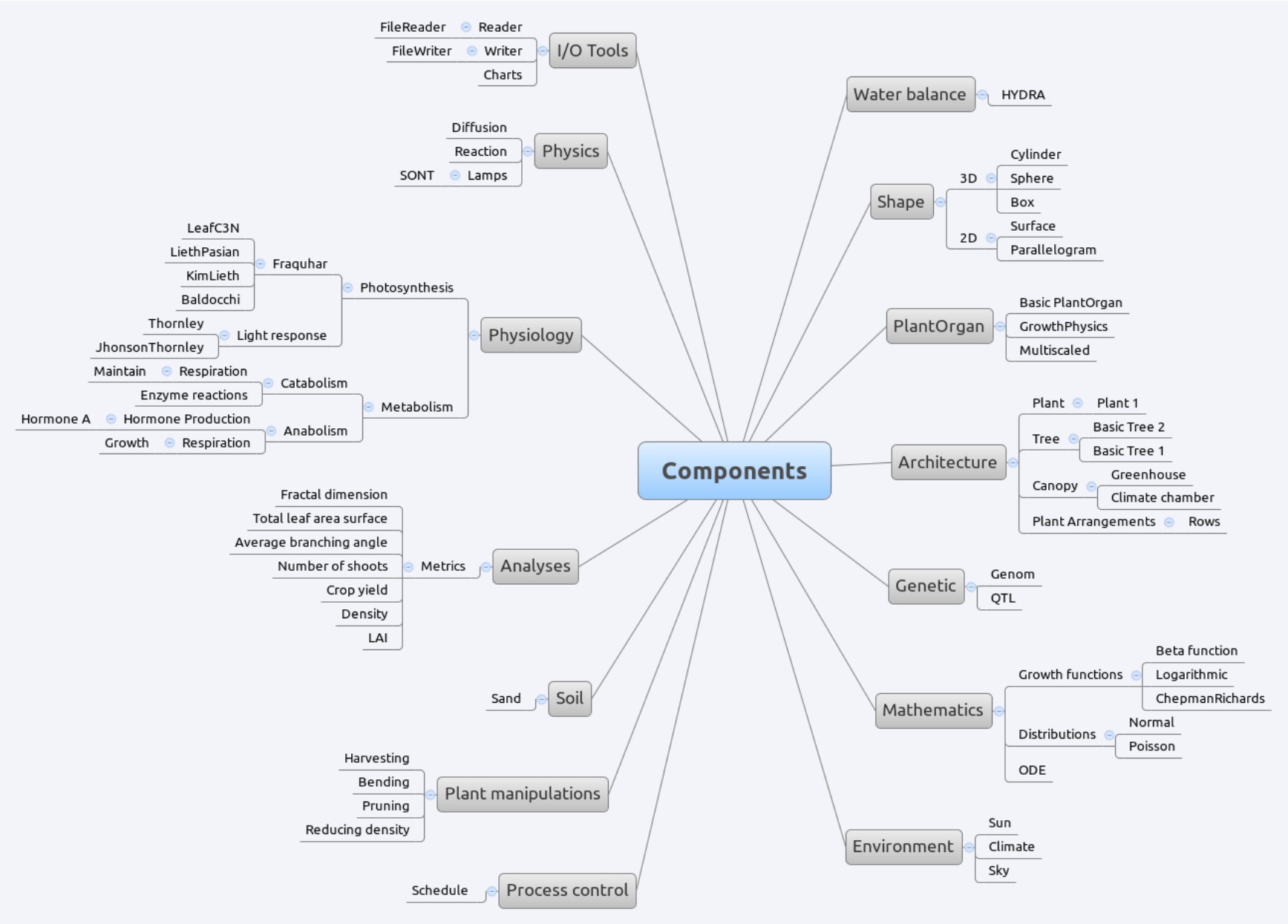
VisualGroIMP - Current state

Component developer view

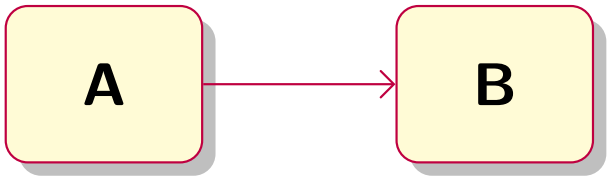
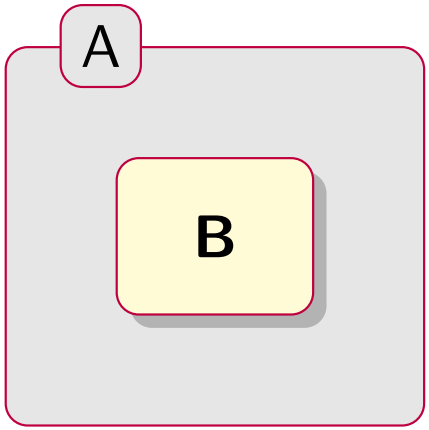
- Special layout for developers
- Not visible for “normal” users
- Additional functions: possibility to develop components



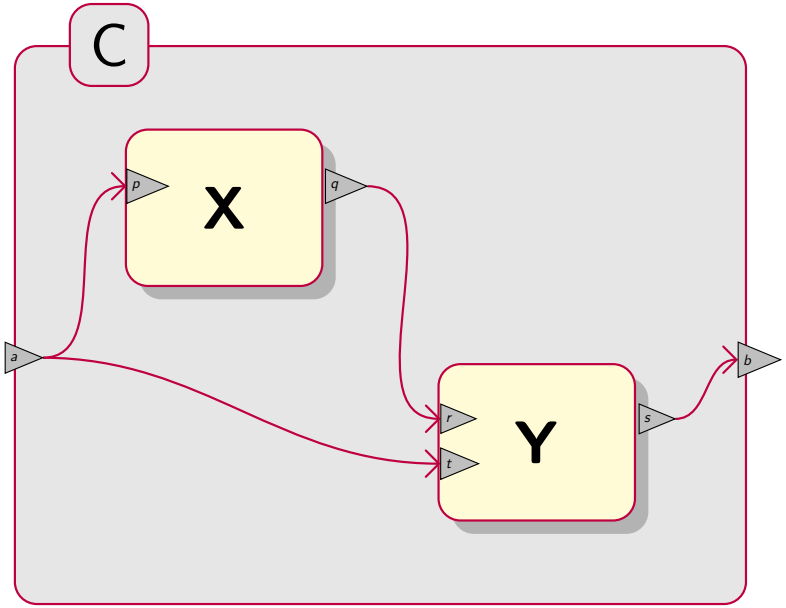
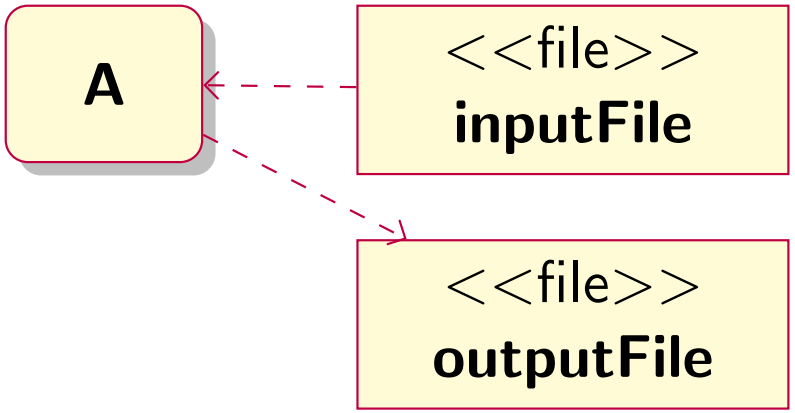
Component library



Connectors

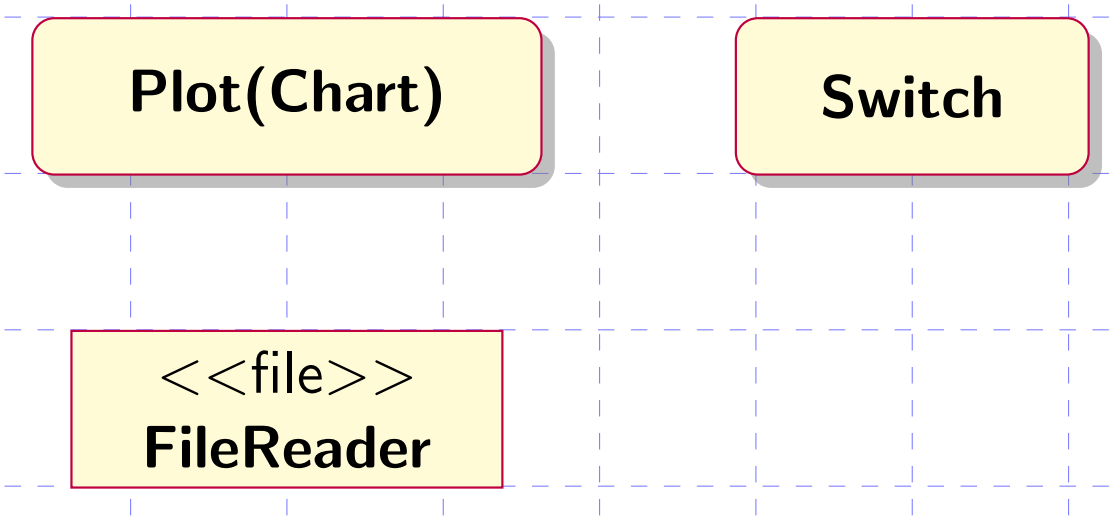
Relation	Edge type	Graph
Usage	~>	
Composition	/>	

Connectors

Relation	Edge type	Graph
Slot	$-\$-> $	
Send	$-s-> $	

Tool components

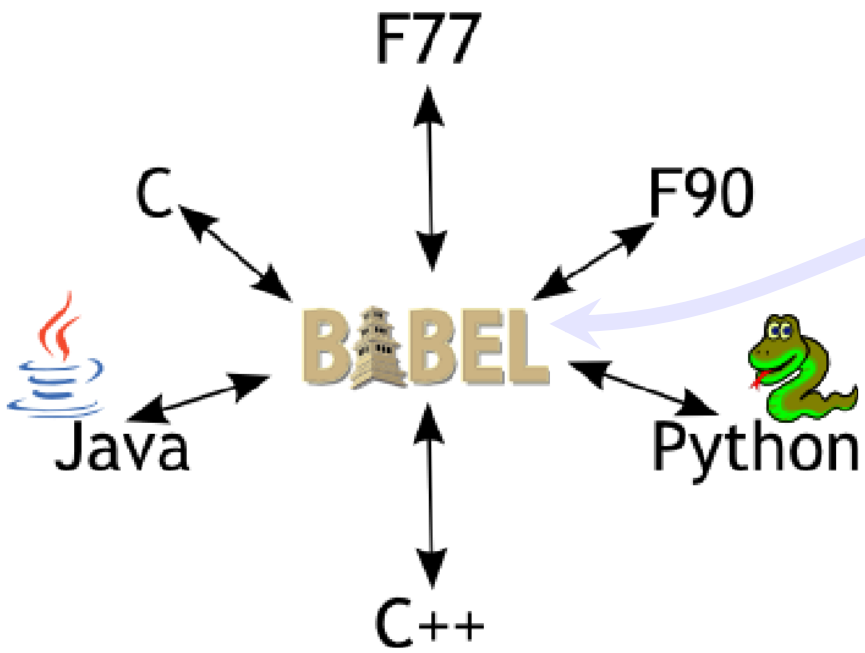
"Re-useable" components

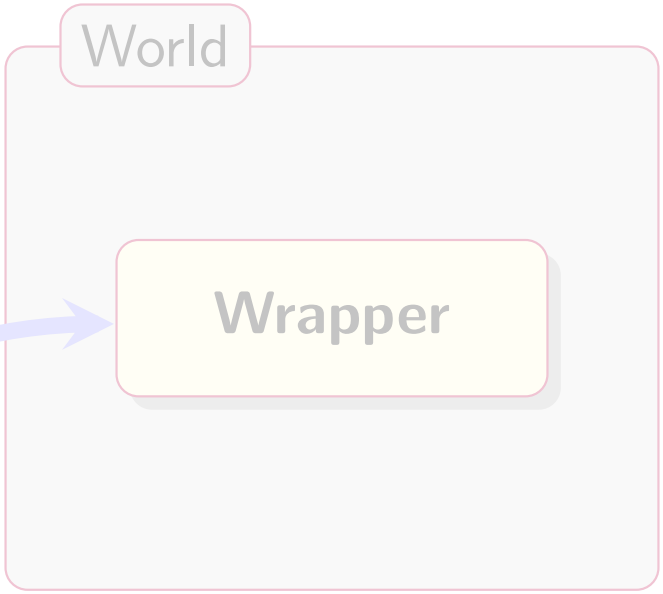


Wrapper Components for Multi-Language Scientific Software

Babel tool:

- to make scientific software libraries equally accessible from all of the standard languages
- generated glue code

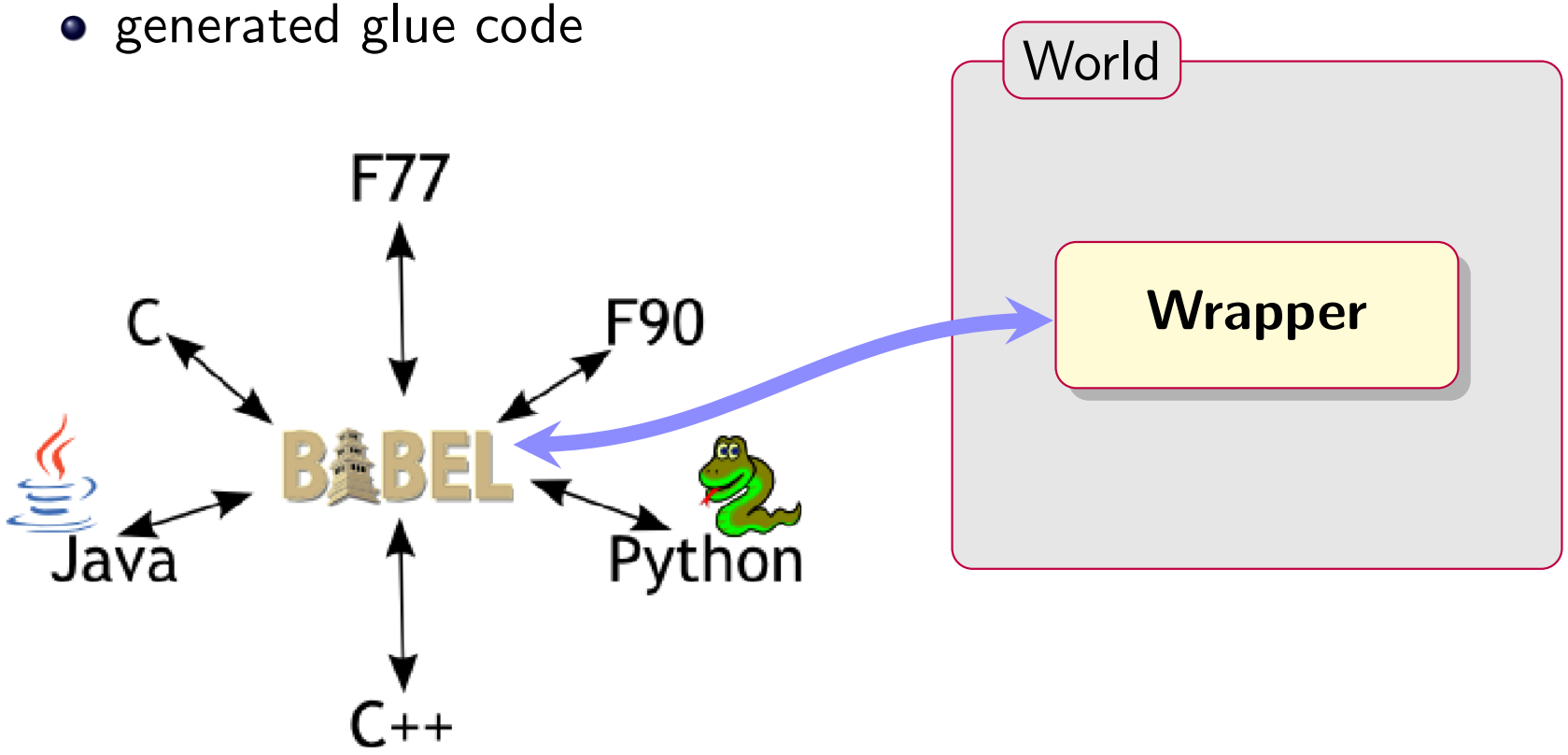




Wrapper Components for Multi-Language Scientific Software

Babel tool:

- to make scientific software libraries equally accessible from all of the standard languages
- generated glue code



Team

Project leader	:	W. Kurth
Project member	:	K. Smoleňová, M. Henke
Co-worker / Student assistant	:	Wu Shining, Ding Cong, Ong Yongzhi
Consulting	:	R. Hemmerling

Thank you for your attention!

References

William T. Councill, George T. Heineman
Component-Based Software Engineering: *Addison-Wesley*, 2001, ISBN 0-201-70485-4

Clemens Szyperski
Component Software, Beyond Object-Oriented Programming: *Addison-Wesley, London*, 2002, ISBN 0-201-74572-0

Greg Wilson
Where's the Real Bottleneck in Scientific Computing?: *American Scientist*, January-February 2006, Volume 94, Number 1, Page: 5, DOI: 10.1511/2006.1.5