Towards Merging PlatΩ and PGIP

David Aspinall\textsuperscript{1} \quad Serge Autexier\textsuperscript{2}
Christoph Lüth\textsuperscript{2} \quad Marc Wagner\textsuperscript{2,3}

\textsuperscript{1}University of Edinburgh, Scotland
\textsuperscript{2}DFKI Bremen, Germany
\textsuperscript{3}Saarland University, Saarbrücken, Germany

8th International Workshop
User Interfaces for Theorem Provers
Montréal, Canada
22. August 2008
PGKIT Architecture

Prover Components

Prover 1

Prover 2

Middleware

Broker

Graphical User Interface

Text Editor

Display Components

File System Theory Store

Aspinall, Autexier, Lüth, Wagner

Towards Merging PlatΩ and PGIP

German Research Center for Artificial Intelligence
PGKIT Architecture

- ACL2, Isabelle, HOL, Lambda-Clam, Lego, Plastic, Phox, ...
- ASCII Files (Input-only)
- Specific parsers
- Command-Line oriented
- Single focus
- Display of proof state
PlatΩ Architecture

Prover Components  Middleware  Display Components

Prover  \( \xrightarrow{\text{PLATO}_P} \)  PlatΩ  PL  \( \xrightarrow{\text{PLATO}_D} \)  TEXmacs

OMDoc  URI to Theories in Text-Editor Document Format or OMDoc
PlatΩ Architecture

- Writing $\textsf{TEXMACS}$ documents assisted by $\Omega\text{MEGA}$
- $\Omega\text{MEGA}$: OMDoc Input/Output
- $\textsf{TEXMACS}$: Specific, flexible proof document style (PL)
- Multiple foci
- Bidirectional: proof steps done by the prover are propagated into the $\textsf{TEXMACS}$ document
- XUPDATE/XMLDIFF based
- On the fly conversions between OMDoc and PL
- Context-sensitive menus for assistance
<table>
<thead>
<tr>
<th></th>
<th>PlatΩ Display</th>
<th>PG Kit Display</th>
<th>PGIP 2 Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document format</td>
<td>XML</td>
<td>Plain text</td>
<td>XML</td>
</tr>
<tr>
<td>Document syntax</td>
<td>TeX MACS</td>
<td>ASCII</td>
<td>Generic</td>
</tr>
<tr>
<td>Change protocol</td>
<td>XUpdate</td>
<td>PGIP&lt;sub&gt;D&lt;/sub&gt;</td>
<td>XUpdate</td>
</tr>
<tr>
<td>Operations</td>
<td>Context-dependent menus</td>
<td>Global menus, typed operations</td>
<td>Context-dependent menus, typed operations</td>
</tr>
</tbody>
</table>

Towards Merging PlatΩ and PGIP
Aspinall, Autexier, Lüth, Wagner

German Research Center for Artificial Intelligence
<table>
<thead>
<tr>
<th></th>
<th>PlatΩ Prover</th>
<th>PG Kit Prover</th>
<th>PGIP 2 Prover</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document format</td>
<td>XML</td>
<td>Plain text</td>
<td>XML</td>
</tr>
<tr>
<td>Document syntax</td>
<td>OMDoc</td>
<td>Native prover syntax</td>
<td>Generic</td>
</tr>
<tr>
<td>Change protocol</td>
<td>XUpdate</td>
<td>PGIP\textsubscript{P}</td>
<td>XUpdate</td>
</tr>
<tr>
<td>Prover support</td>
<td>ΩMEGA</td>
<td>Generic (Coq, Isabelle, etc)</td>
<td>Generic (Coq, Isabelle, ΩMEGA, etc)</td>
</tr>
<tr>
<td>Operations</td>
<td>Context-dependent menus</td>
<td>Global menus, typed operations</td>
<td>Context-dependent menus, typed operations</td>
</tr>
</tbody>
</table>

Towards Merging PlatΩ and PGIP
Aspinall, Autexier, Lüth, Wagner
Overview of Extensions for PGIP 2

1. Support for Semi Structured Documents
   - Multiple Foci
   - XUpdate/Semantic XMLDiff
   - Protocols

2. Context-Sensitive Service Menus

3. Multiple Editing Displays in Parallel

4. Multiple Document Formats

Working hypothesis: Do not require any change for current displays/provers when using the new PGIP 2

Towards Merging PlatΩ and PGIP
Aspinall, Autexier, Lüth, Wagner

German Research Center for Artificial Intelligence
1 Support for Semi Structured Documents
   Multiple Foci
   XUpdate/Semantic XMLDiff
   Protocols

2 Context-Sensitive Service Menus

3 Multiple Editing Displays in Parallel

4 Multiple Document Formats
theory NN
imports Main
begin

text {∗ The \texttt{datatype} of natural numbers\footnote{Not the real ones!}. ∗}
datatype N = Z | S N

text {∗ Induction is a derived concept: \texttt{@\{thm N.induct\}} ∗}

fun "add" :: "N \Rightarrow N" where
  "add Z x = x"
| "add (S x) y = S (add x y)"

notation add (infixr "+" 65)

text {∗ We first need two lemmas. ∗}

lemma add\_Z\_r: "x + Z = x"
  by (induct x, simp+)

lemma add\_S: "S (x+ y) = x+ (S y)"
  by (induct x, simp+)

text {∗ Now we can prove commutativity. ∗}

theorem add\_commute: "x+ y = y+ (x::N)"
  proof (induct x rule: N.induct)
    case Z thus ?case by (simp add: add\_Z\_r)
    case S thus ?case by (simp add: add\_S)
  qed

end

Towards Merging PlatΩ and PGIP
Aspinall, Autexier, Lüth, Wagner
<table>
<thead>
<tr>
<th>Display</th>
<th>Broker</th>
<th>Prover</th>
</tr>
</thead>
<tbody>
<tr>
<td>User requests file to be loaded.</td>
<td>&lt;loadparsefile&gt;</td>
<td>Text is successfully parsed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Users edits a text region.</td>
<td>&lt;newcmd&gt;</td>
<td>Text is successfully parsed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Old text is deleted, edited text is inserted.</td>
<td>&lt;delcmd&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User request command to be processed.</td>
<td>&lt;newcmd&gt;</td>
<td>Several prover commands are successfully executed.</td>
</tr>
<tr>
<td></td>
<td>&lt;setcmdstatus&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;cmdstatus&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
theory NN
imports Main
begin

datatype N = Z | S N

text {∗ The \textit{datatype} of natural numbers \footnote{Not the real ones!}. ∗} <theoryitem>

datatype N = Z | S N </theoryitem>

ncase → "opengoal/">
<theoryitem objtype = "theory declaration">text {∗ Induction is a derived concept: \texttt{thm N.induct} ∗}</theoryitem>

<theoryitem objtype = "theory declaration">text {∗ We define addition via recursive equations, and \textit{declare} its \textit{syntax} as the usual infix operator. ∗}</theoryitem>

<theoryitem objtype = "theory declaration">fun &quot;add&quot; :: &quot;N \Rightarrow N \Rightarrow N&quot; where

\texttt{add \ Z \ x = x};
\texttt{add \ (S \ x) \ y = S \ (add \ x \ y)}</theoryitem>

<pgip>
Semi Structured Documents

theory thyname="NN" parentnames="Main">theory NN
imports Main
begin
<block objtype="theory body">

<theoryitem>text {∗ The datatype of natural numbers\footnote{Not the real ones!}. ∗} </theoryitem>
<theoryitem type="adt">datatype N = Z | S N</theoryitem>

<theoryitem>text {∗ Induction is a derived concept: @{thm N.induct} ∗} </theoryitem>
<theoryitem>text {∗ We define addition via recursive equations, and declare its syntax as the usual infix operator. ∗} </theoryitem>

<theoryitem objtype="fundef">fun "add" :: "N ⇒ N ⇒ N" where "add Z x = x"
| "add (S x) y = S (add x y)"
<theoryitem>
<proofstep/>
<endproof/>
</theoryitem>
<assertion>
</theoryitem>
</block>
</theory>
</theory>
</document>
</pgip>
Supporting Semi Structured Documents

- New internal PGIP markup (PGML), tree compatible
  - Use xml:id to flexibilise document structure
    - Example: theorems and proofs need not occur together physically
  - Rigid structure computed by the broker for the prover

- Provide on-the-fly converters between old and new markup
  ⇒ provers need not be adapted

- Configuration
Towards Merging PlatΩ and PGIP
Aspinall, Autexier, Lüth, Wagner
Towards Merging PlatΩ and PGIP
Aspinall, Autexier, Lüth, Wagner
Towards Merging PlatΩ and PGIP
Aspinall, Autexier, Lüth, Wagner
Single Focus

Towards Merging PlatΩ and PGIP
Aspinall, Autexier, Lüth, Wagner

German Research Center for Artificial Intelligence
Towards Merging PlatΩ and PGIP
Aspinall, Autexier, Lüth, Wagner

German Research Center
for Artificial Intelligence
Multiple Foci

Towards Merging PlatΩ and PGIP
Aspinall, Autexier, Lüth, Wagner
Supporting Semi Structured Documents

- New internal PGIP markup, tree compatible
- Provide on-the-fly converters between old and new markup
  ⇒ provers need not be adapted
- Configuration
Old PGIP \(_D\) protocol (asymmetric):

- Display → Broker: `<createcmd>`, `<editcmd>`
- Broker → Display: `<newcmd>`, `<delcmd>`, `<replacecmd>`

New PGIP \(_D\) protocol:

- Rephrase in terms of Xupdate-modifications
  `<insert-after>`, `<insert-before>`, `<remove>`
Compatibility

- XUPDATEs cannot be generated by all displays or provers
- Allow components to send only new document version and compute differences inside the broker
- Use semantic XMLDIFF tool inside the broker
  - Normal tree diff
  - Configurable to take some “semantics” into account
  - Configurable level of granularity for modifications
- Configuration if component can handle XUPDATEs
  Broker generates appropriate representation for each component
Protocol between Prover and Broker

- `<paresescript>` and `<parseresult>`
- Prover commands in native prover command syntax
- prover answers `<normalresponse>`, `<errorresponse>`, `<ready>`

Unchanged, but contextualized (multiple foci)

Extension:

- `<normalresponse>` can contain XUPDATES for document which are relayed to the display
Protocols between Display with Broker

In addition to PGIP\(_D\) provide

- XML-RPC interface and
- HTTP in plain REST architecture

<table>
<thead>
<tr>
<th>HTTP Request</th>
<th>HTTP answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>/initsession</td>
<td>path to new session (/session1)</td>
</tr>
<tr>
<td>/session1/initdocument?CONTENT</td>
<td>HTML rendered document after parsing</td>
</tr>
<tr>
<td>/session1/document/1</td>
<td>version 1 of the document</td>
</tr>
<tr>
<td>/session1/document/HEAD</td>
<td>the current version of the document</td>
</tr>
</tbody>
</table>
1 Support for Semi Structured Documents
   Multiple Foci
   XUpdate/Semantic XMLDiff
   Protocols

2 Context-Sensitive Service Menus

3 Multiple Editing Displays in Parallel

4 Multiple Document Formats
Service Menus

- Allow requests of service menus for arbitrary elements in the display
- Requests relayed by the broker to the prover
- Prover returns menu description (PGML menu format)
  - Menus contain descriptions and actions
  - Actions can be
    - tactic calls (traditional)
    - requests for axiom/lemma applications ($PlatΩ$/$TEXMACS$)
- Computing the complete menus is expensive, most of the menu is discarded
- Support lazy menu computation
  - Menu entries trigger computations which results are modifications of the menu
1. Support for Semi Structured Documents
   Multiple Foci
   XUpdate/Semantic XMLDiff
   Protocols

2. Context-Sensitive Service Menus

3. Multiple Editing Displays in Parallel

4. Multiple Document Formats
Multiple Editing Displays

Prover Components

Prover 1

PGIP_P

Prover 2

PGIP_P

Middleware

Broker

PGIP_D

Display Components

Graphical User Interface

Text Editor

Eclipse

File System Theory Store

Towards Merging PlatΩ and PGIP
Aspinall, Autexier, Lüth, Wagner

German Research Center for Artificial Intelligence
Multiple Editing Displays

Prover Components

- Prover 1
- Prover 2

Middleware

- Broker

Display Components

- Graphical User Interface
- Text Editor
- Eclipse

File System Theory Store

Requires Synchronisation by the Broker

Towards Merging PlatΩ and PGIP
Aspinall, Autexier, Lüth, Wagner

German Research Center for Artificial Intelligence
Multiple Displays

- Simple kind of version control inside the broker
- Broker stores for each display the latest version communicated
- Uses semantic XMLMerge (diff3)
- Conflicts are marked in special markup extending XUpdate markup.

Towards Merging PlatΩ and PGIP
Aspinall, Autexier, Lüth, Wagner
1. Support for Semi Structured Documents
   Multiple Foci
   XUpdate/Semantic XMLDiff
   Protocols

2. Context-Sensitive Service Menus

3. Multiple Editing Displays in Parallel

4. Multiple Document Formats
Multiple Document Formats

- Have: Display D communicates in format A
- Have: Prover P communicates in format B
- Wanted: Use display D with prover P

For instance:
- $\text{Plat}\Omega$ connects $\text{TEX}_{\text{MACS}}$ with format PL to $\Omega_{\text{MEGA}}$ with format OMDoc
- Now want to use $\text{TEX}_{\text{MACS}}$ with Isabelle (Isar)
Multiple Document Formats

- Have: Display D communicates in format $A$
- Have: Prover P communicates in format $B$
- Wanted: Use display D with prover P

For instance:
- $\text{Plat} \Omega$ connects $\text{TeX}_{\text{MACS}}$ with format $\text{PL}$ to $\Omega\text{MEGA}$ with format $\text{OMDoc}$
- Now want to use $\text{TeX}_{\text{MACS}}$ with Isabelle (Isar)
- Provide autonomous converters from $\text{PL}$ to $\text{OMDoc}$ and $\text{OMDoc}$ and Isar
- Bidirectional
- Conversion of Xupdates
- Maintain mappings to relay service requests
Related Work

- Semi-structured input documents
  *MathsTiles* (structured input to multiple domain reasoners, unidirectional)

- Multiple foci
  *ΩMEGA* (status management added), Isabelle (multi-foci under the way)

- Multiple displays
  *Matita* (proof script, goal display), *LΩUI*, *GeoProof*/*Coq*

- Support modifications to the document from the prover:
  *Matita* (tinycals)
  Could be realised by the generic functionality proposed here?
Next Steps

- Implementation (any feedback/remarks welcome before we start...)
- Implement $Plat\Omega$ functionality on PGIP 2 basis
- Connect other editing displays (Word 2007, OpenOffice, Netbeans)
- Multiple foci, changes from prover could serve interaction styles in other provers (Agda, Mizar, PVS)