Three studies on Model Transformations
- Parsing, Generation and Ease of Use

Håkan Burden
Model

- Description
- Abstraction
- Usable

- C code: 1001110…
- Grammar: "one bottle of milk,
two bottles of milk, …"
Model Transformations

Source → Target

• Rules
• Algorithm
• Manual … Automatic
• Endogeneous or Exogeneous
• Abstraction level
# Examples

<table>
<thead>
<tr>
<th>Level of abstraction</th>
<th>S&lt;T</th>
<th>S=T</th>
<th>S&gt;T</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Endogenous</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Abstracts in publications</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Re-factoring/re-ordering</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Optimisations</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parsing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Exogenous</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Subtitles Generation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Google Translate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code generation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ease of Use</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Parsing Linear Context-Free Rewriting Systems

Håkan Burden and Peter Ljunglöf

IWPT´05
International Workshop on Parsing Technologies
2005
Dutch Subordinate Clauses

...dat Jan Piet Marie de kinderen zag helpen leren zwemmen
...that Jan Piet Marie the children saw help teach swim
A Grammar

- \( \text{Prop} \rightarrow \text{jz}[\ ] := \{ \text{subj}="\text{Jan}" ; \text{pred}="\text{zag}" \} \)
- \( \text{Prop} \rightarrow \text{ph}[\ ] := \{ \text{subj}="\text{Piet}" ; \text{pred}="\text{helfen}" \} \)
- \( \text{Prop} \rightarrow \text{ml}[\ ] := \{ \text{subj}="\text{Marie}" ; \text{pred}="\text{leren}" \} \)
- \( \text{Prop} \rightarrow \text{kz}[\ ] := \{ \text{subj}="\text{de kinderen}" ; \text{pred}="\text{zwemmen}" \} \)

- \( \text{Prop} \rightarrow \text{rec}[\text{Prop}_1, \text{Prop}_2] := \{ \text{subj}=\text{Prop}_1.\text{subj} \ \text{Prop}_2.\text{subj} ; \text{pred}=\text{Prop}_1.\text{pred} \ \text{Prop}_2.\text{pred} \} \)

- \( \text{SCI} \rightarrow \text{subcl}[\text{Prop}] := \{ \text{cl}="\text{dat}" \ \text{Prop}.\text{subj} \ \text{Prop}.\text{pred} \} \)
Bottom-up Parsing

s="dat Jan Piet Marie de kinderen zag helpen leren zwemmen"
Natural Language Generation from Class Diagrams

Håkan Burden and Rogardt Heldal
MoDeVVa’11
8th International Workshop on Model-Driven Engineering, Verification and Validation
2011
Background

Natural Language Generation:
• Text planning
• Sentence planning
• Linguistic realisation

Class Diagram:
• Software Model
Two Models

Grammatical Framework:
cat Concept, Prop
fun Aircraft, Airline : Concept
Statement : Concept × Concept → Prop

lincat Concept = N, Prop = S
lin Airline = mkN "airline"
Aircraft = mkN "aircraft" "aircraft"
Statement = mkS (mkCl (mkNP a_Det Airline))
(mkV2 "operate"
(mkV2 "operate"
(mkNP many_Det Aircraft)))

"an airline operates many aircraft"

Class Diagram:

```
Aircraft
  * operate ▲
     1

Airline
```


Executable and Translatable UML – How Difficult Can it Be?

Håkan Burden, Rogardt Heldal and Toni Siljamäki
APSEC’11
18th Asia-Pacific Software Engineering Conference
2011
From Idea to Software

- Idea
- Analysis Models
- Design Models
- C
- 100110
Analysis Models
Design Models

```
// this unrelate should be AoT
select one reset related by self -> LAPRESET[R4];
if (not_empty reset)
    unrelate self from reset across R4;
end if;

self.seconds = self.seconds + 1;
create event instance tick of WTIMR2: 'tick' to self;
t = TIM::timer_start({ microseconds: 1000000, event:
    LOG::LogInfo(message: "timer tick")};
```
Can teams of four students solve a problem that is complex enough to require using the full potential of the design models within a total of 300 hours?

<table>
<thead>
<tr>
<th></th>
<th>2009</th>
<th>2010</th>
<th>2011</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Teams</td>
<td>22</td>
<td>28</td>
<td>24</td>
<td>74</td>
</tr>
<tr>
<td>#Success</td>
<td>18</td>
<td>25</td>
<td>23</td>
<td>66</td>
</tr>
<tr>
<td>#Failure</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>8</td>
</tr>
</tbody>
</table>
Conclusion

Parsing:
- Endogenuous
- Refinement

Ease of Use:
- Exogenuous
- Synthesis

Generation:
- Exogenuous
- Reverse engineering
Future Work

• Further explore generation from SW models
  — Action language to natural language

• Models and transformations in industry
  — Transformations as re-usable assets
  — Modelling cross paradigms

• Educational implications