CHROME: A Model-Driven Component-Based Rule Engine

Jairson Vitorino

PhD Thesis, CIn-UFPE
February 2009

Supervisor: Prof. Jacques Robin
Contents

1. Context of thesis: the ORCAS project
   - State-of-the art in automated reasoning and reused-oriented software engineering
   - Extending both in synergy: the ORCAS project
   - Goals of the thesis within project

2. Base technologies:
   - Model-Driven Architecture (MDA) languages and methods
   - Constraint Handling Rules with Disjunctions (CHR\(^{\vee}\))

3. CHROME: Model-driven component assembly for an easy to extend, scalable, adaptive CHR\(^{\vee}\) engine

4. Model-transformation based CHR\(^{\vee}\) rule base to Java compiler

5. Contributions

6. Limitations and future work
Thesis Context: ORCAS Project Motivation

- Limitations of Automated Reasoning (AR) technology:
  - Few reuse of AR services
  - Lack of modern Software Engineering
  - Poor Integration with mainstream Software

- Limitations of Software Engineering technology:
  - Lack of tools for MDE and CBD
  - Lack of conceptually complex examples (AR, Compilers)
  - Potential AR use overlooked (e.g. model checking)
Thesis context: Orcas Goals

Orcas: A framework of AR components

Based on CHRv

Model-driven Engineered

AR services

- Constraint solving over any domain
- Classical First-Order Logic (CFOL) satisfiability and entailment
- Description Logic (DL) subsumption
- Abduction
- Agent belief update and action planning
- Truth-maintenance, default reasoning and inheritance

cf. Früwirth, Abdennadher, Fages, Martinez, Rodrigues, Christiansen, Thielscher, Martin, Yin, Wolf, Silva, Robin
Goals of the Thesis

- Most reused component and basis for ORCAS
- Deployable on a mainstream platform (Java)
- With built-in conflict-directed backjumping for efficient search
- With built-in truth-maintenance for adaptive, anytime, online reasoning
- With CHR base to Java code compiler designed as a pipeline of model-transformations.

- Demonstrate the synergy between: Software components, MDE and Model transformation
- Demonstrate the applicability to engineer: CHROME Engine

Compilers between languages from different paradigms Artificial intelligence software (Large, non-toy model-transformations, 4358 lines of ATL)
1. Context of thesis: the ORCAS project
   ➣ State-of-the art in automated reasoning and reused-oriented software engineering
   ➣ Extending both in synergy: the ORCAS project
   ➣ Goals of the thesis

2. Base technologies:
   ➣ Model-Driven Architecture (MDA) languages and methods
   ➣ Constraint Handling Rule with Disjunctions (CHR\(^\lor\))

3. Model-driven component assembly for an easy to extend, scalable, adaptive CHR\(^\lor\) engine

4. Model-transformation based CHR\(^\lor\) rule base to Java compiler

5. Contributions

6. Limitations and future work
Base Technologies: MDE Languages and Methodologies

- Unified Modeling Language 2.1 (UML):
  - OMG main MDE standard to specify Platform Independent Models (PIMs) and Platform-Specific Models (PSM, using self-extension profile mechanism)
  - Integrates concepts from imperative, Object-Oriented (OO), concurrent, distributed and component-based paradigms
  - Covers structural, behavioral, functional and deployment aspects of a software

- Object Constraint Language 2.0 (OCL):
  - Textual part of UML2 to specify arbitrary first-order logic constraints among UML2 model elements
  - Allows modeling “executable” PIMs, i.e., refined enough to be fully automatically translated into running code by MT
  - Functional OO syntax concise and intuitive for mainstream developers
Atlas Transformation Language (ATL, Bezevin, INRIA-Rennes)

- Hybrid rule-based and imperative MT language
- Pattern-matching model element rewrite rules with embedded procedures
- Input and output models conform to a meta-model in Ecore (a MOF2 variant)
- Core is an OCL2 execution engine for input model element pattern matching and output model element construction ($\approx 80\%$ of an ATL program is OCL)
- Eclipse project, largest user community, *de facto* standard
Base Technologies:

MDECBD Method
UML/OCL based
16 views of Comp.

CHROME is its first large application case study for Kobra2
Base technologies: CHR\textsuperscript{v} rule base concrete syntax and logical semantics

Constraint simpagation rules:
- Rule syntax: $K \setminus R \iff G \mid B.$, with:
  - $K$ keep heads, and $R$ remove heads, both conjunctions of so called User-Defined Constraints (UDC)
  - $G$ guards, conjunction of so called Built-In Constraints (BIC)
  - $B$ bodies, disjunction of conjunction of either RDC or BIC

Constraint store $S$ (volatile CHR\textsuperscript{v} KB) = $S_r \land S_b$ with $S_r$ conjunction of UDC and $S_b$ conjunction of BIC

Query $Q$: conjunction of constraints, either RDC or BIC

CHR\textsuperscript{v} propagation rule $K \Rightarrow G \mid B,$ syntactic variant of simpagation rule $K \setminus \text{true} \iff G \mid B.$

CHR\textsuperscript{v} simplification rule $R \iff G \mid B,$ syntactic variant of simpagation rule true $\setminus R \iff G \mid B.$
CHR$^\vee$ by Example: Justification and solution adaptation

r1@ a \ b <=> c
r2@ a,c <=> e,d
r3@ g <=> f

query: a{1}, b{2}, g{3}

remove: a{1}

remove all labeled {1}

re-adding removed constraints
Outline

1. Context of thesis: the ORCAS project
   - State-of-the art in automated reasoning and reused-oriented software engineering
   - Extending both in synergy: the ORCAS project
   - Goals of the thesis
2. Base technologies:
   - Model-Driven Architecture (MDA) languages and methods
   - Constraint Handling Rule with Disjunctions (CHR\(^\vee\))
3. CHROME: Model-driven component assembly for an easy to extend, scalable, adaptive CHR\(^\vee\) engine
4. Model-transformation based CHR\(^\vee\) rule base to Java compiler
5. Contributions
6. Limitations and future work
CHROME: Top-Level Assembly
KobrA2 Realization Structural Class Service View
\[ r3 \text{ label } @ \text{label}(X), \text{indomain}(X, l(A, l(B, l(C, [])))) \iff \text{true} \mid X = A; X = B; X = C. \]
CHROME Run-Time PIM Assembly

- 8 KobrA2/UML2 Components
- 40 KobrA2/UML2 Classes
- 33 KobrA2 view packages
- 30 UML2 diagrams
- 187 lines of OCL2 expressions
Outline

1. Context of thesis: the ORCAS project
   - State-of-the art in automated reasoning and reused-oriented software engineering
   - Extending both in synergy: the ORCAS project
   - Goals of the thesis
2. Base technologies:
   - Model-Driven Architecture (MDA) languages and methods
   - Constraint Handling Rule with Disjunctions (CHR$^\vee$)
3. CHROME: Model-driven component assembly for an easy to extend, scalable, adaptive CHR$^\vee$ engine
4. Model-transformation based CHR$^\vee$ rule base to Java compiler
5. Contributions
6. Limitations and future work
CHROME: Compiler Pipeline

CHR$^v$

CHR2CHRCore >> CHRCore
948 lines

CHRCore2UMLOCLJ.atl >> 2489 lines

UMLOCL

UMLOCL2JAVA.atl >> 720 lines

JAVA

JAVA2String.atl >> 192 lines

Java code
CHROME Compiler Stage 1: from full CHR$^\triangleright$ to CHRCORE

Example input (textual):
\[ k(1, X) \setminus d(X) \leq g \mid b. \]

Example output (textual):
\[ k \setminus d \leq g, k.at(1) = 1, k.at(2) = d.at(1) \mid b. \]

- ✔ Simpler matching
- ✔ Sequence of Equalities
Basic idea: Every Constraint is converted into a UML Class

r1@ a \ b <==> g | c ; e
r2@ a,c <==> g | false
r3@ a,e ==> g | true

AConstraint
+execute()
-executeR1()
-executeR2()
-executeR3()
....

BConstraint
+execute()
-executeR1()
....
CHROME Compiler Stage 2: from core CHR to UML2/OCL2

<table>
<thead>
<tr>
<th>AConstraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>+execute()</td>
</tr>
<tr>
<td>-executeR1()</td>
</tr>
<tr>
<td>-executeR2()</td>
</tr>
<tr>
<td>-executeR3()</td>
</tr>
<tr>
<td>-checkGuardR1()</td>
</tr>
<tr>
<td>-checkGuardR2()</td>
</tr>
<tr>
<td>-checkGuardR3()</td>
</tr>
<tr>
<td>-addR1Body()</td>
</tr>
<tr>
<td>-addR2Body()</td>
</tr>
<tr>
<td>-addR3Body()</td>
</tr>
</tbody>
</table>
1. Context of thesis: the ORCAS project
   - State-of-the art in automated reasoning and reused-oriented software engineering
   - Extending both in synergy: the ORCAS project
   - Goals of the thesis
2. Base technologies:
   - Model-Driven Architecture (MDA) languages and methods
   - Constraint Handling Rule with Disjunctions (CHR\(^\lor\))
3. Model-driven component assembly for an easy to extend, scalable, adaptive CHR\(^\lor\) engine
4. Model-transformation based CHR\(^\lor\) rule base to Java compiler
5. Testing and benchmarking
6. Contributions
7. Limitations and future work
Contributions

To CHR and CLP:

- First justification-based adaptive CHR$^\vee$ engine (crucial for practical applications and tracing);
- First CHR$^\vee$ engine with intelligent search (CDBJ);
- First component-based CHR$^\vee$ engine (easy to extend);
- First MDE CHR$^\vee$ engine (easy to port to other OO host platforms).

To MDE:

- CHROME: Largest case study to date to integrate MDE with MT and components for AR.
- First MDE/MT compiler from source language to target language from different structural paradigm (4358 ATL lines).
Outline

1. Context of thesis: the ORCAS project
   - State-of-the art in automated reasoning and reused-oriented software engineering
   - Extending both in synergy: the ORCAS project
   - Goals of the thesis

2. Base technologies:
   - Model-Driven Architecture (MDA) languages and methods
   - Constraint Handling Rule with Disjunctions (CHR\(^{\vee}\))

3. Model-driven component assembly for an easy to extend, scalable, adaptive CHR\(^{\vee}\) engine

4. Model-transformation based CHR\(^{\vee}\) rule base to Java compiler

5. Testing and benchmarking

6. Contributions

7. Limitations and future work
Limitations

- Only 3 built-in constraints: =, true, false
- No visual tracing IDE
- Still an order of magnitude slower than CHR Prolog platforms on run benchmark
- Untested scalability for other AR tasks beyond finite domain solvers
- Compiler not component-based and verbose (ATL)

Future work

- Visual tracing IDE (MSc. Thesis of Rafael Oliveira 2010)
- Create variable size benchmark for variety of AR tasks
- Port to Python to test scalability of transparent distribution to Google's cloud
- Extend to run OO CHR bases (cf. MSc. of Marcos Silva 2009)
Thank you!
Any Questions?