Freie Universität Berlin

Milestone 3 (of 3)

Softwareprojekt Compilerbau (19517e)

Instructor: Prof. Dr. Elfriede Fehr

Free University Berlin

Institute of Computer Science

Freie Universität Berlin

Outline

General Goals

- Task
- Specification & Interfaces

Team Javabite

- Team & Results

Team FUC

- Team & Results

Crosstesting

Task

Modularized Compiler Framework

Modules Lexer, Parser, Semantical Analyzer, Code Generation, Backend

User Interface Error reporting and **Visualisation** of data structures like AST and TAC

Two Module Sets

Modules intercompatibility

Targets **LLVM** & **JVM**

Clear Interface Design

Implementation implementing by spec and verification by runtime cross tests

Specification & Interfaces

Documentation & Source

Github Wiki & Issues	history and discussion
Shared	
Interface Library	common
Interfaces	modules and data structures

Language

types	bool, long, double, string, array, struct
control	if-else, while, do-while
I/O	no input, UTF-8 output

History

Three Milestones

Milestone 1	Arithmetic, Return, Declarations, Assignments
Milestone 2	Logic, Print, If-Else
Milestone 3	Blocks, Arrays, Loops, AST-Visualization
Nice 2 Have	Records, more GUI-stuff

Interface Design Process

Iterative based on milestone feature sets

1. Proposals
2. Discussion with members of both teams on Github
3. Changes
4. Decision

Modules

Lexer

Input	UTF-8 character stream
Output	Token stream (Type, Value, Line, Column)

Parser

Input	Token stream (consumed token by token)
Output	AST, Reports

Semantical Analyzer

Input	AST
Output	wellformed AST, Reports

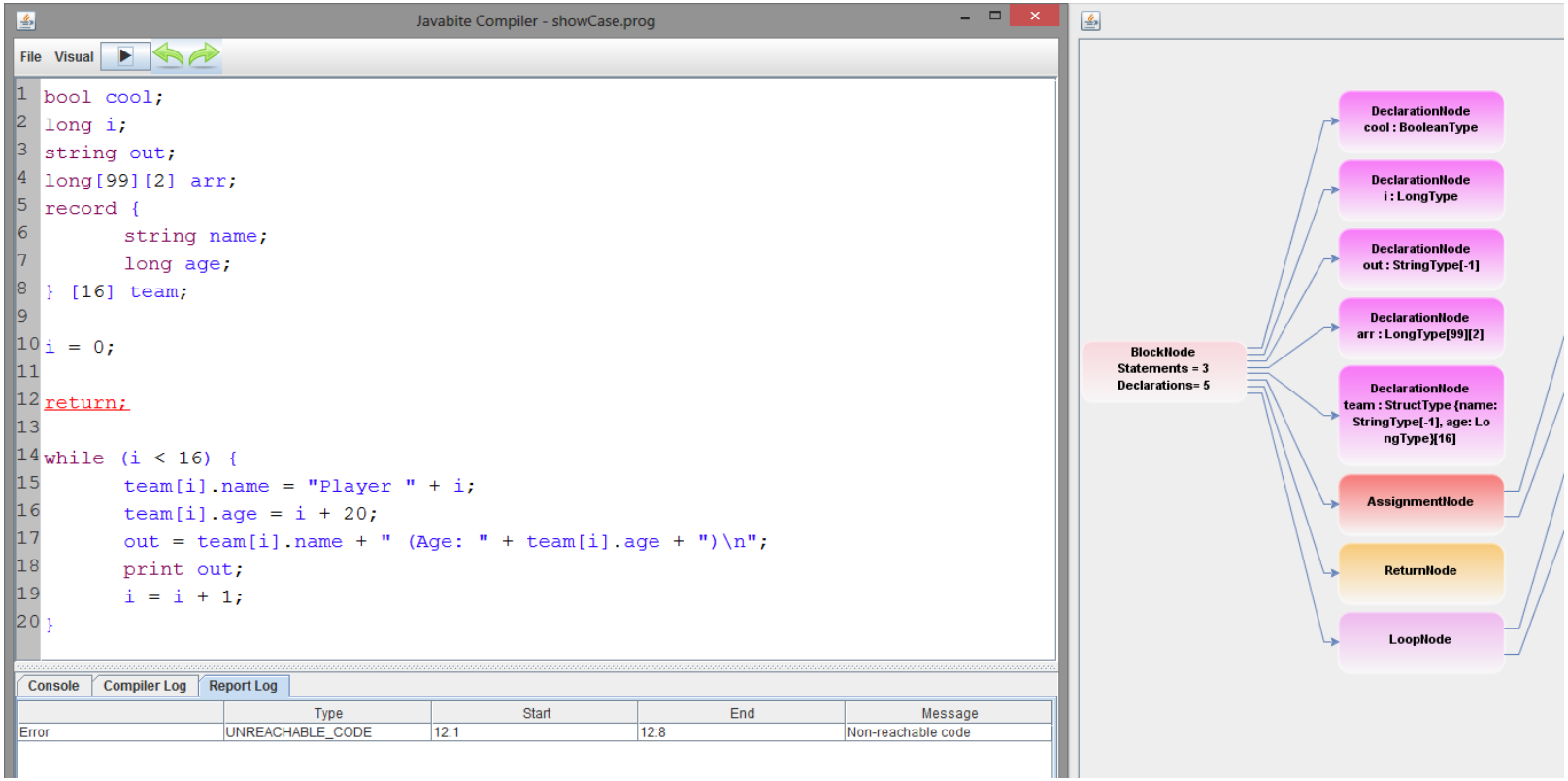
Modules

Code Generator

Input	wellformed AST
Output	TAC as list of Quadruples

Backend

Input	TAC
Output	streams of targetcode



TEAM JAVABITE

Outline

- The Team
- The Tools
- The Modules
  - Lexer
  - Parser
  - Semantical Analyser
  - Code Generator
  - Backend
- The GUI

The Team

4 Teams + Organizer

Lexer + GUI

Damla, Ferhat, Sebastian

Parser + Semantical Analyzer + AST-Visualization

Till, Ahmet, Mahmoud, Khalid

Intermediate Code Generator

Florian, Vivienne, Alpin, Yang

Backend

Marco, Eike, Robert

Organization

Weekly meeting

status updated and task planning

Communication

Mail, Github, Skype

The Tools

Source & Documentation

Git / Github

IDE

Eclipse

Build

Gradle

The Modules - Lexer

Complex Java-Regex

Java-Regex ⊃ Regular Expressions

Helper Token

Java Regex -> Automaton (Matcher)

```
TRUE(TokenType.TRUE, "true(?!\\w)"),
FALSE(TokenType.FALSE, "false(?!\\w)"),
```

```
(?<TRUE>true(?!\\w))|(?<TRUE>false(?!\\w))
```

The Modules - Parser

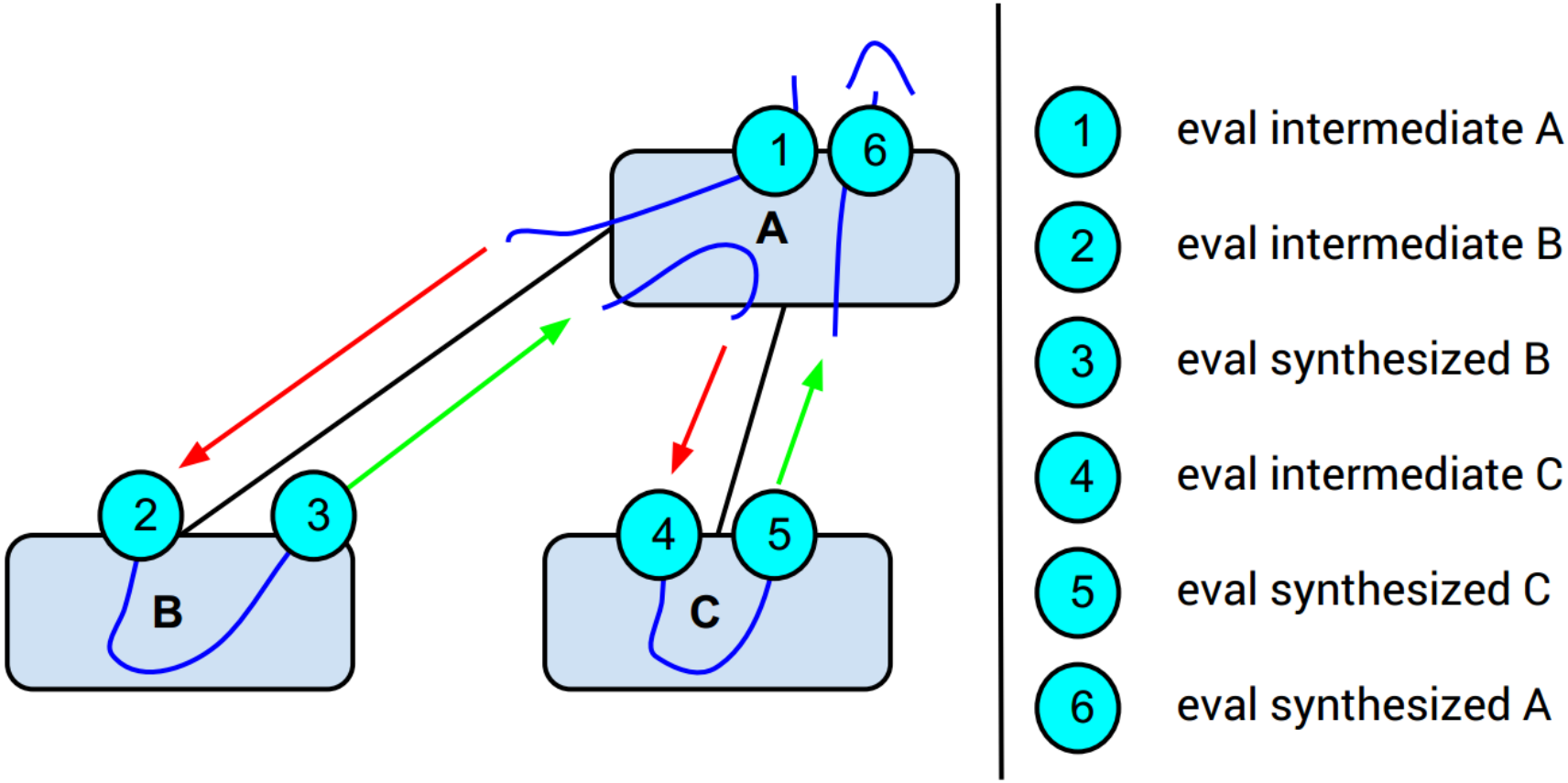
Grammar

- LR-Automaton
- derivates word
- applicable for arbitrary LR-parsable grammar
- AST generated by derivation-driven SDT

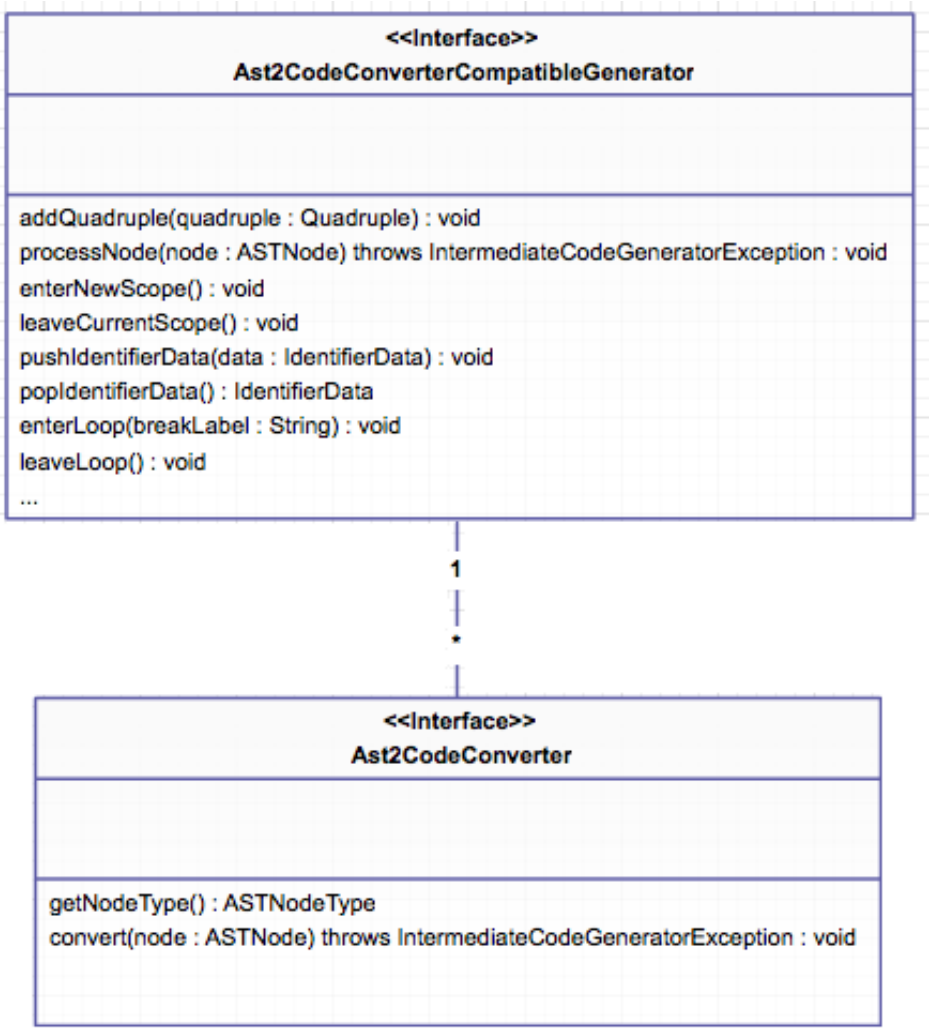
Possible issues

- word is not in grammar
- ambiguous derivation (dangling-else)
- (handle unrecognized tokens)

The Modules - Semantical Analyzer



The Modules - Code Generation



The Modules - Backend

Target

JVM 1.5

Classfile

Structure defined by Java® Virtual Machine Specification

Narrowed to necessary

Translator

TAC preprocessing

TAC to ByteCode

Tools

javac / javap

jclasslib (integrated in GUI)

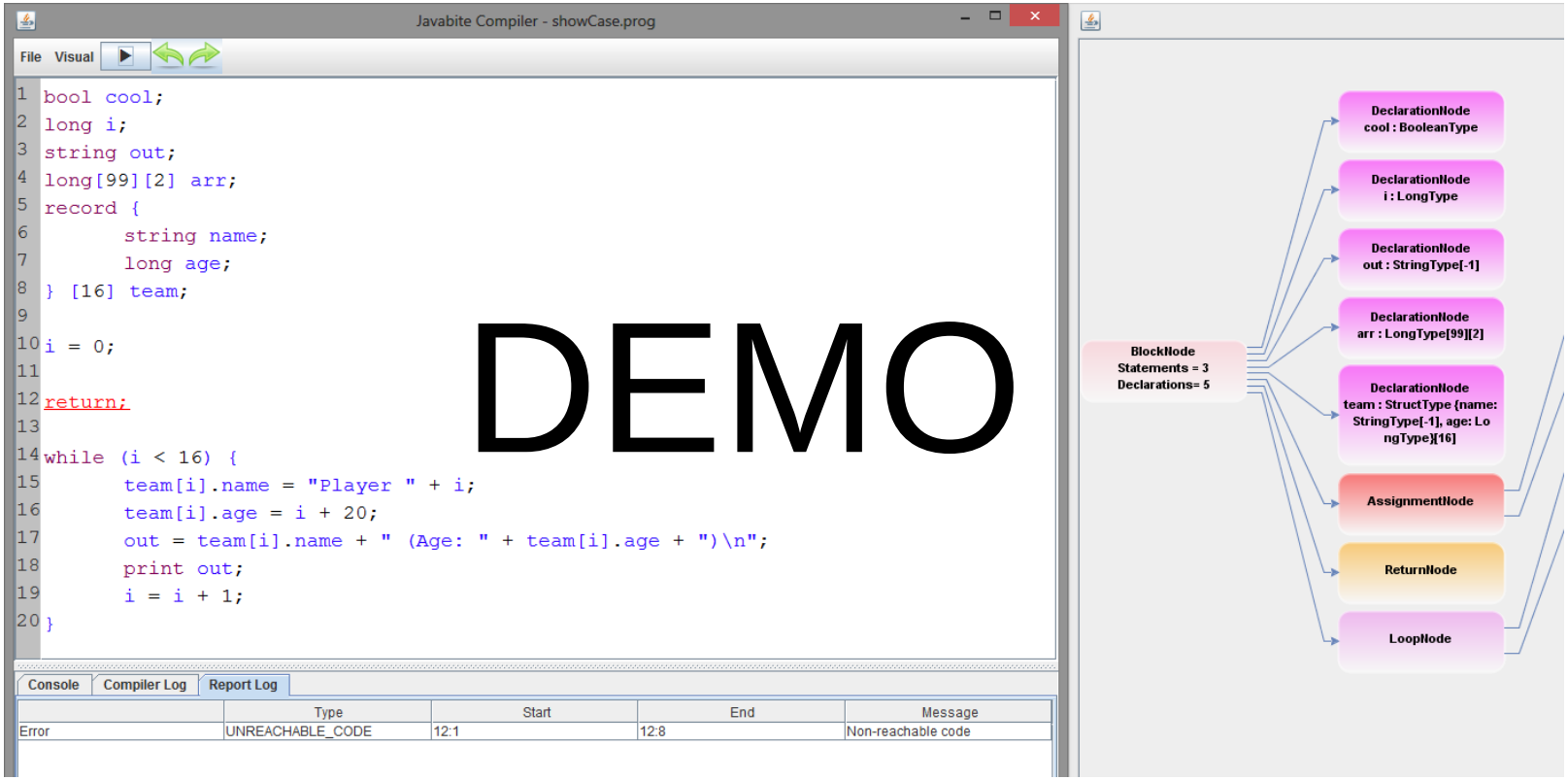
hexdump (integrated in GUI)

The GUI

Features

- Syntax Highlighting + TokenType-Tooltips
- Error Feedback
- AST-/TAC-/Byte-Code-Visualization
- Java-Byte-Code-Execution
- Partly configurable

The GUI


Javabite Compiler - showCase.prog
File
Visual
bool cool;
long i;
string out;
long[99][2] arr;
record {
string name;
long age;
} [16] team;
i = 0;
return;
while (i < 16) {
team[i].name = "Player " + i;
team[i].age = i + 20;
out = team[i].name + " (Age: " + team[i].age + ")\n";
print out;
i = i + 1;
}
DEMO
Console
Compiler Log
Report Log
Type
Start
End
Message
Error
UNREACHABLE_CODE
12:1
12:8
Non-reachable code
BlockNode
Statements = 3
Declarations= 5
DeclarationNode
cool : BooleanType
DeclarationNode
i : LongType
DeclarationNode
out : StringType[-1]
DeclarationNode
arr : LongType[99][2]
DeclarationNode
team : StructType {name: StringType[-1], age: LongType}[16]
AssignmentNode
ReturnNode
LoopNode

Lessons learned

teams should have equal strength

-> good decision, hard to reach

check specs extensively

-> details

use tools know to team members

-> learning effort

The Conclusion

all necessary

features

+

records

Q&A