Freie Universität Berlin

Milestone 1

Vorstellung der Gruppe *fuc* und des aktuellen Standes

Sven-Kristofer Pilz
Freie Universität Berlin

Softwareprojekt Übersetzerbau

Inhalt

Organisation

- Arbeitsgruppen
- Kommunikation
- Entwicklung

Milestone 1

- Lexer
- Parser
- Semantische Analyse
- Drei-Adress-Code
- LLVM-Backend
- Controller
- Zusammenfassung

Outline

Arbeitsgruppen mit Ansprechpartner

- Lexer
  - Thomas.
- Parser
  - Björn und Samuel.
- Semantische Analyse
  - Christoph, Eduard und Sven.
- Drei-Adress-Code
  - Frank, Danny und Manuel.
- LLVM-Backend
  - Roman, Moritz, Jens

Kommunikation

- Arbeitsgruppen organisieren interne Kommunikation eigenständig.
  - Detailfragen entscheidet die Arbeitsgruppe.
- Jeder spricht mit jedem.
- Eigene Mailingliste.
- Jeden Donnerstag Treffen der gesamten Gruppe.
  - Jede Arbeitsgruppe berichtet Status.
  - Größere Entscheidungen per Abstimmung.
- . . . für alles andere gibt es den Projektleiter.

Entwicklung

- Einzige Regel: Master Branch muss immer lauffähig sein.
- Jeder darf in Master *pushen*.
- Arbeitsgruppen arbeiten in eigenen Branches.
- Test Driven
  - Tests werden per ANT ausgeführt, nutzen CI von GitHub.

Outline

Lexer

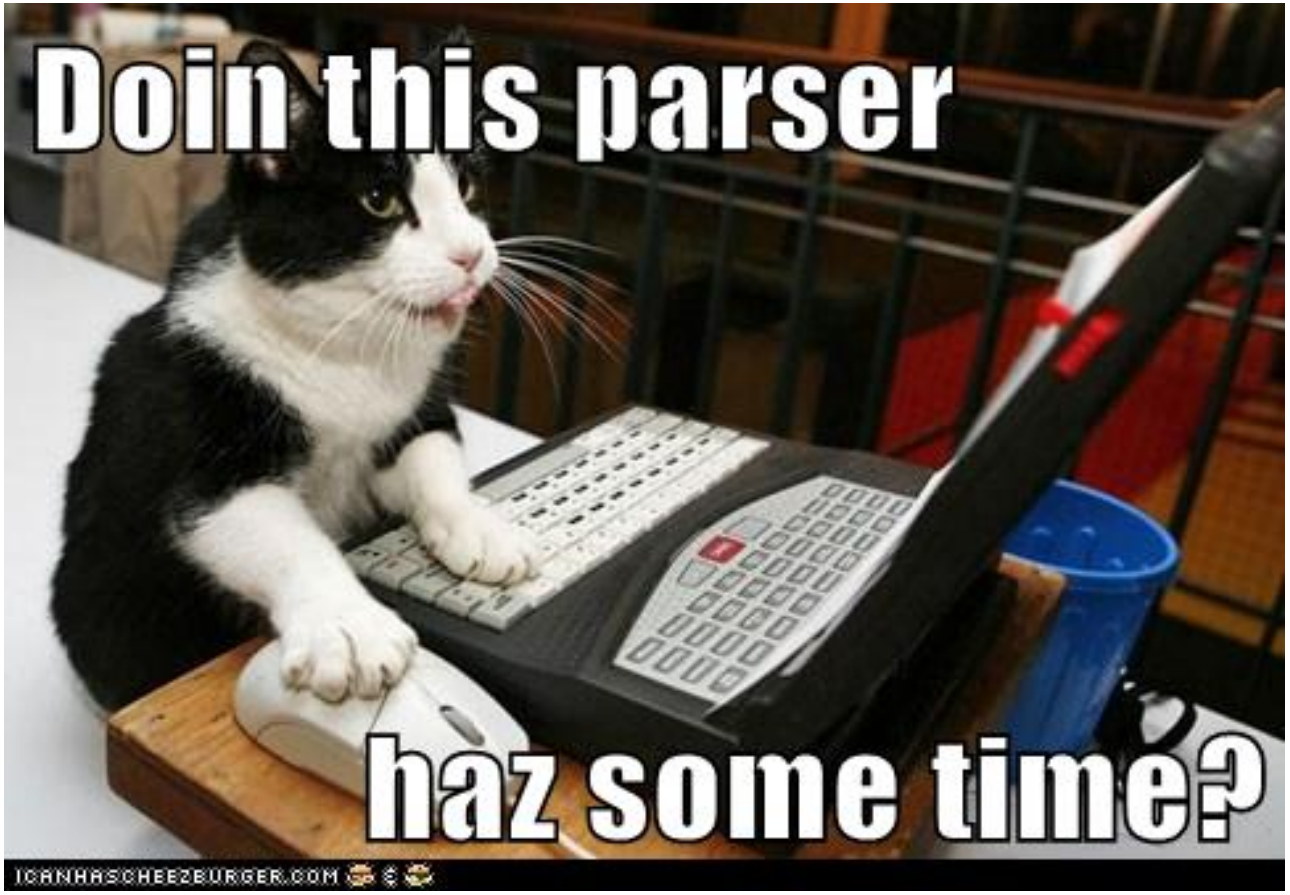
- Sprachumfang für Milestone 1 implementiert.
- Quelltext wird aus InputStream als Liste von Zeilen eingelesen.
- Token als reguläre Ausdrücke spezifiziert.
  - Bereits gesamter Sprachumfang spezifiziert.

Parser

work in progress

Parser

work in progress



Semantische Analyse

- Fehler aus Sprachumfang für Milestone 1 implementiert.
- Findet folgenden Fehler:
  - Verwendung von Variablen ohne Initialisierung.

```
long i
return i
```

Drei-Adress-Code

- Sprachumfang für Milestone 1 implementiert.
- Weiterer Sprachumfang bereits als Stubs vorhanden.
- Hat auch den AST implementiert.
  - Testabdeckung von etwa 100%.
- Visualisierung: Quadruple, Triple und Code.
- . . . wohl bereits sehr beliebt.

Drei-Adress-Code

- Quadruple *(! = leer)*

```
000: (DECLARE_LONG | ! | ! | l)
001: (DECLARE_LONG | ! | ! | tmp0)
002: (ADD_LONG | #3 | #3 | tmp0)
003: (ASSIGN_LONG | tmp0 | ! | l)
004: (RETURN | l | ! | !)
```

- Code

```
000: long l
001: long tmp0
002: tmp0 = #3 + #3
003: l = tmp0
004: return l
```

LLVM-Backend

Freie Universität Berlin

- Drei-Adress-Code für Milestone 1 ist implementiert.
- Kann den Drei-Adress-Code auch aus Textdatei lesen.
- Erzeugter LLVM-Code ließ sich erfolgreich testen.
- Kann Drei-Adress-Code auch direkt ausführen und Rückgabewert prüfen.
  - Dazu ist LLVM auf dem System erforderlich.

Controller

- Ruft die einzelnen Module auf.
- Spricht die Interfaces an.
- Öffnet Quelltextdatei und schreibt Ausgabe.
- Module werden dynamisch als Plug-In geladen.
  - ServiceLoader lädt Implementierungen anhand Interfaces.

Zusammenfassung (Milestone 1)

- Fertig (samt Tests):
  - Lexer
  - Semantische Analyse
  - Drei-Adress-Code
  - LLVM-Backend
- In Arbeit:
  - Parser
  - Visualisierung des Token-Stream
  - Integration in Controller

Auf Wiedersehen

Fragen?