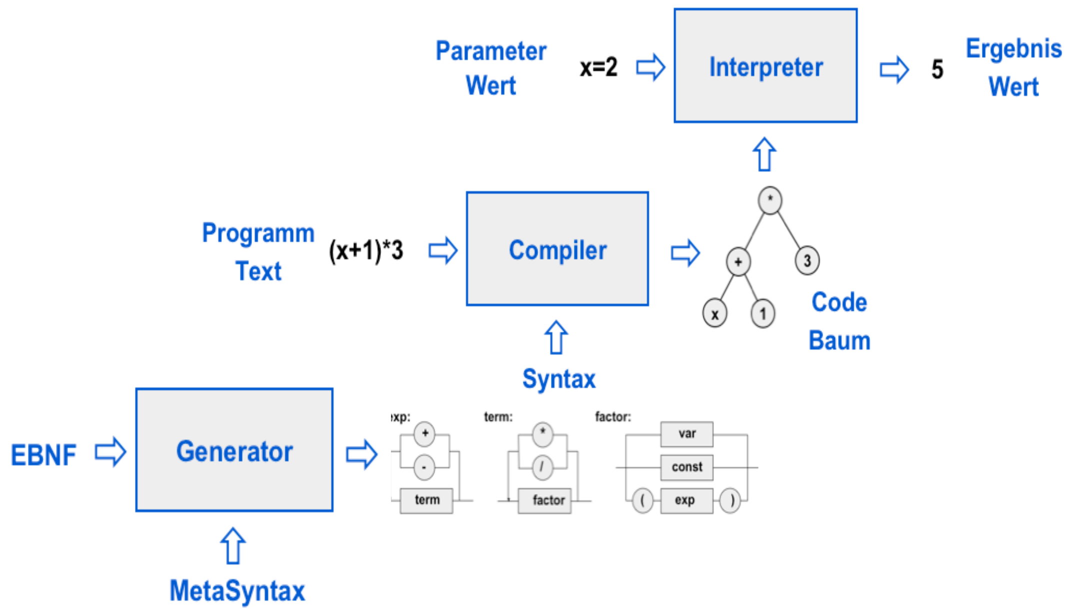
Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Syntaxanalyse



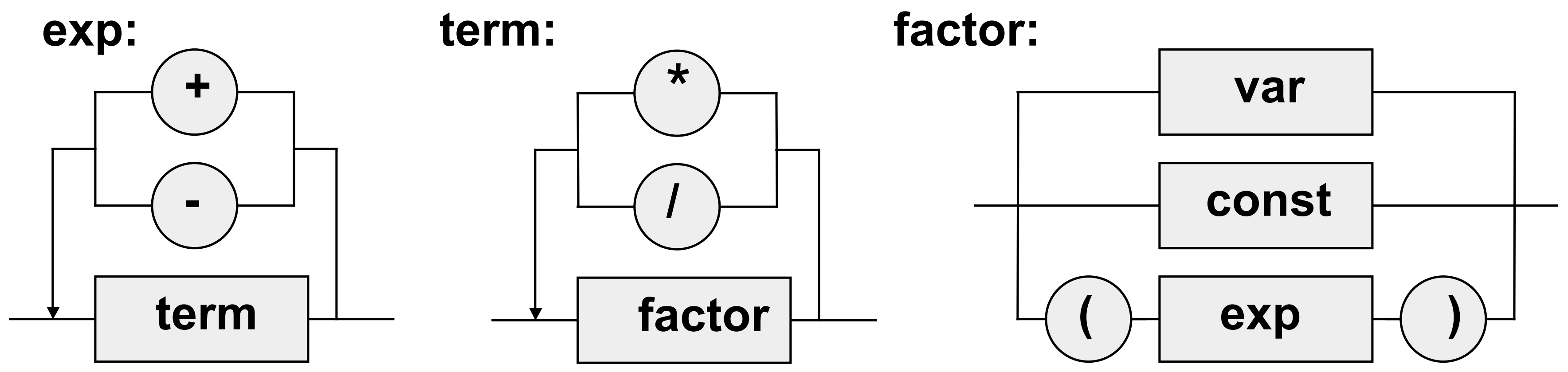
Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Beispiel:
Syntax für arithmetische Ausdrücke in Erweiterter Backus-Naur Form (EBNF)

```
exp = term {("+"|"-") term}.
term = factor {("*"|"/") factor}.
factor = var | const | ( "(" exp ")" ).
```

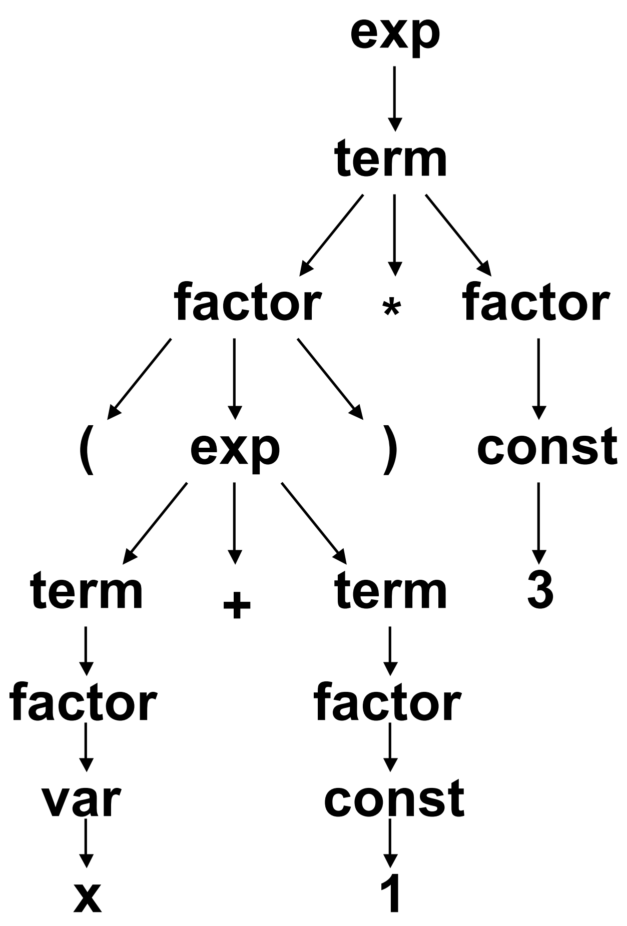
Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Beispiel: Syntaxdiagramme für arithmetische Ausdrücke

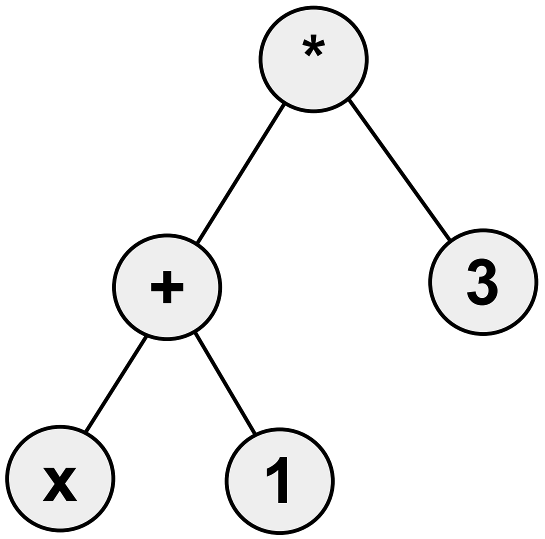


Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Beispiel: (x+1)*3



Parsebaum



Codebaum

Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Beispiel:
Syntax für Lexical Scan

var = letter { letter | digit }.
const = digit { digit }.
letter = "a"|"b"| ... |"z".
digit = "0"|"1"| ... |"9".

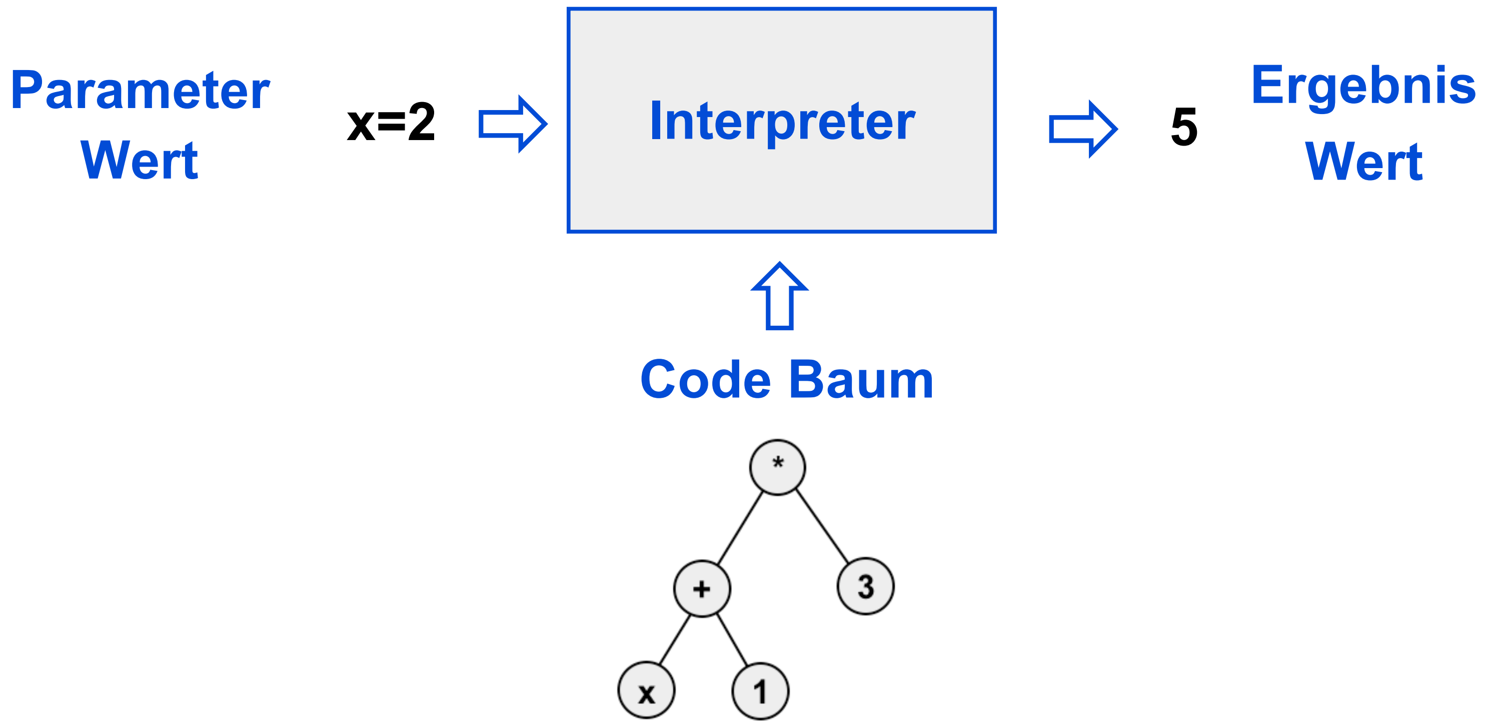
Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Beispiel:
Lexical Scan

```
public class LexicalScanner extends StringTokenizer {
    public LexicalScanner(String string) {
        super(string.trim(),"+-*/^=?()' \n",true);
    }
    public String nextToken() { //  return words, numbers, operators,
                                    brackets or empty string
        String token;
        do {
            if (hasMoreElements()) token = super.nextToken().trim();
            else return ""; // return empty string for end of text
        }
        while (token.equals("")||token.equals("\n")); // skip spaces and \n
        return token;
    }
}
```

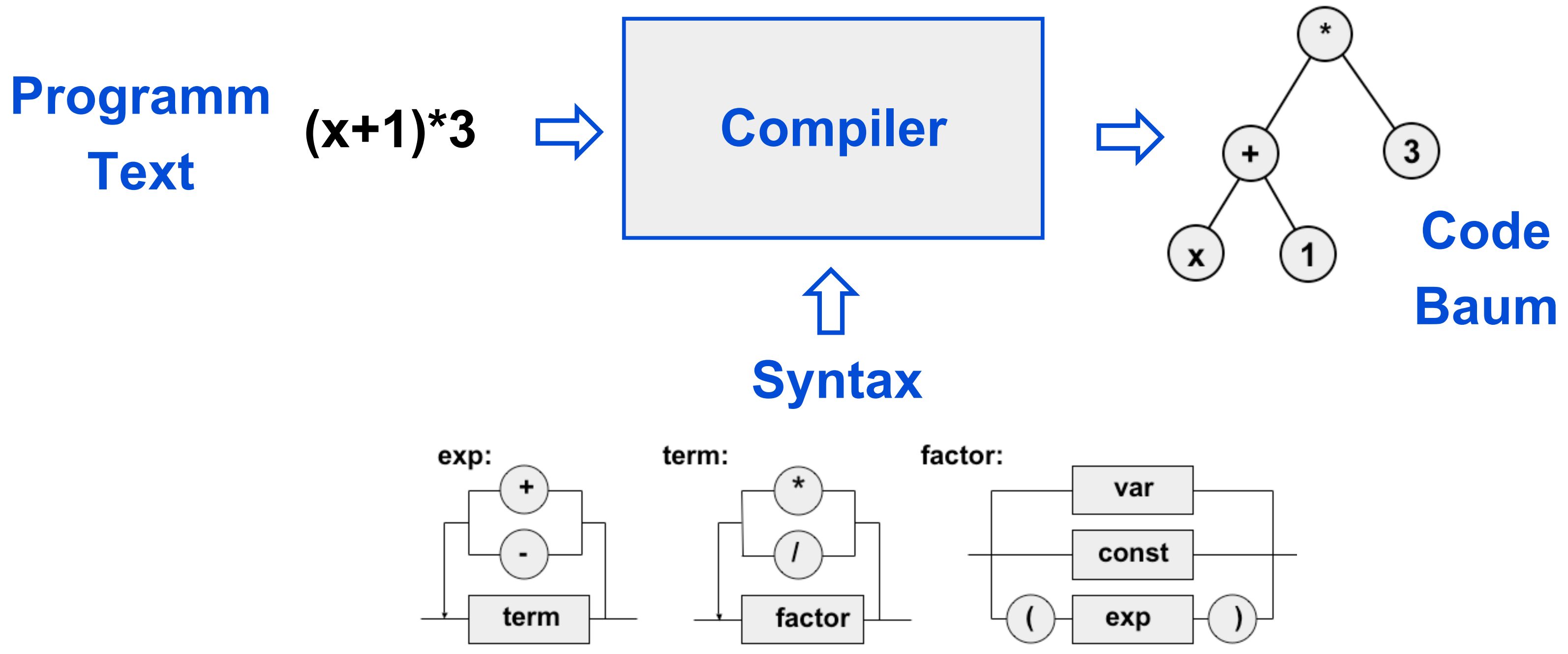
Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Interpreter



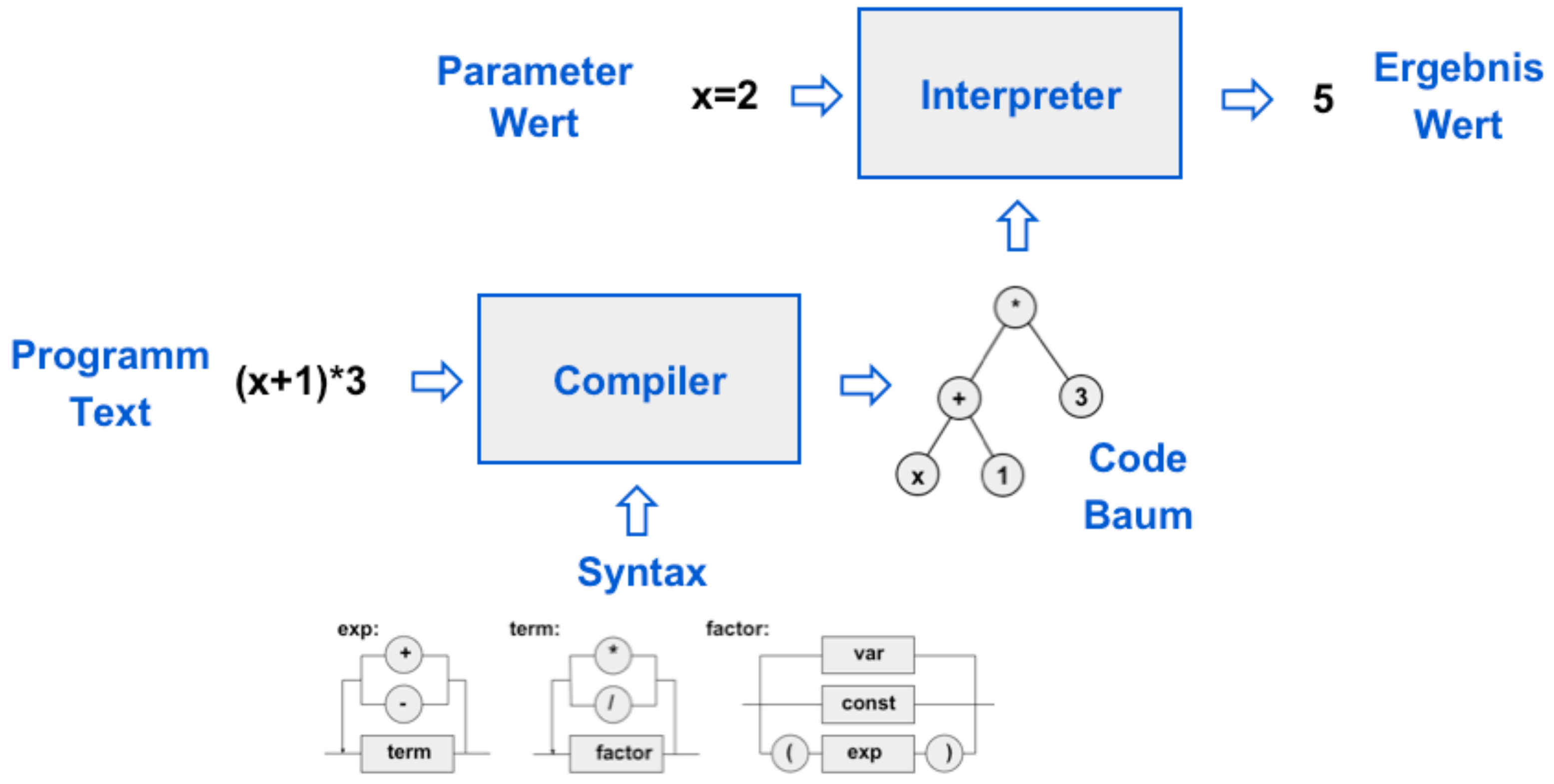
Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Compiler



Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Compiler und Interpreter

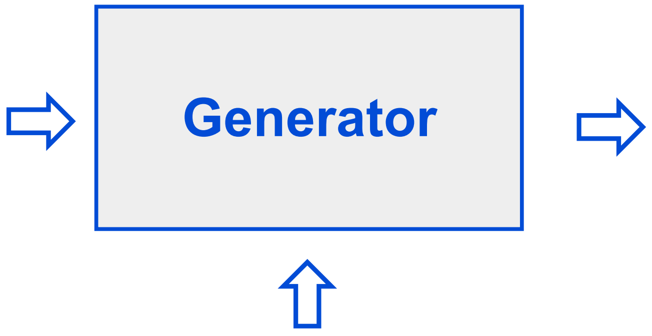


Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

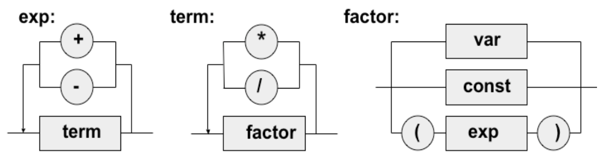
Generator

exp = term {("+"|"-") term}.
term = factor {("*"|"/") factor}.
factor = var | const | ("(" exp ")").

EBNF



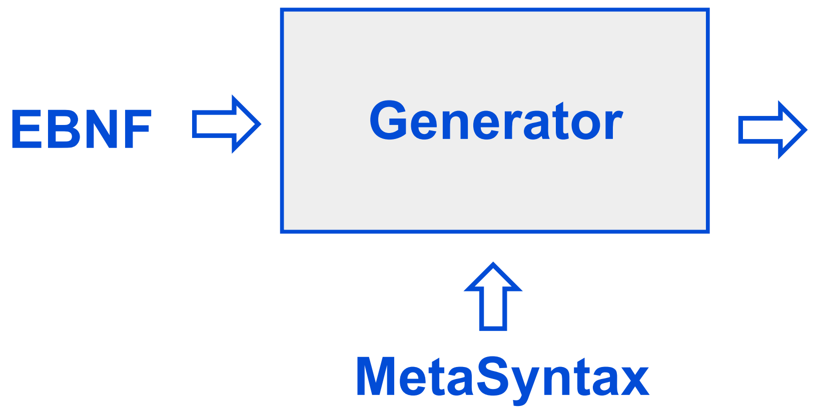
MetaSyntax



Syntax

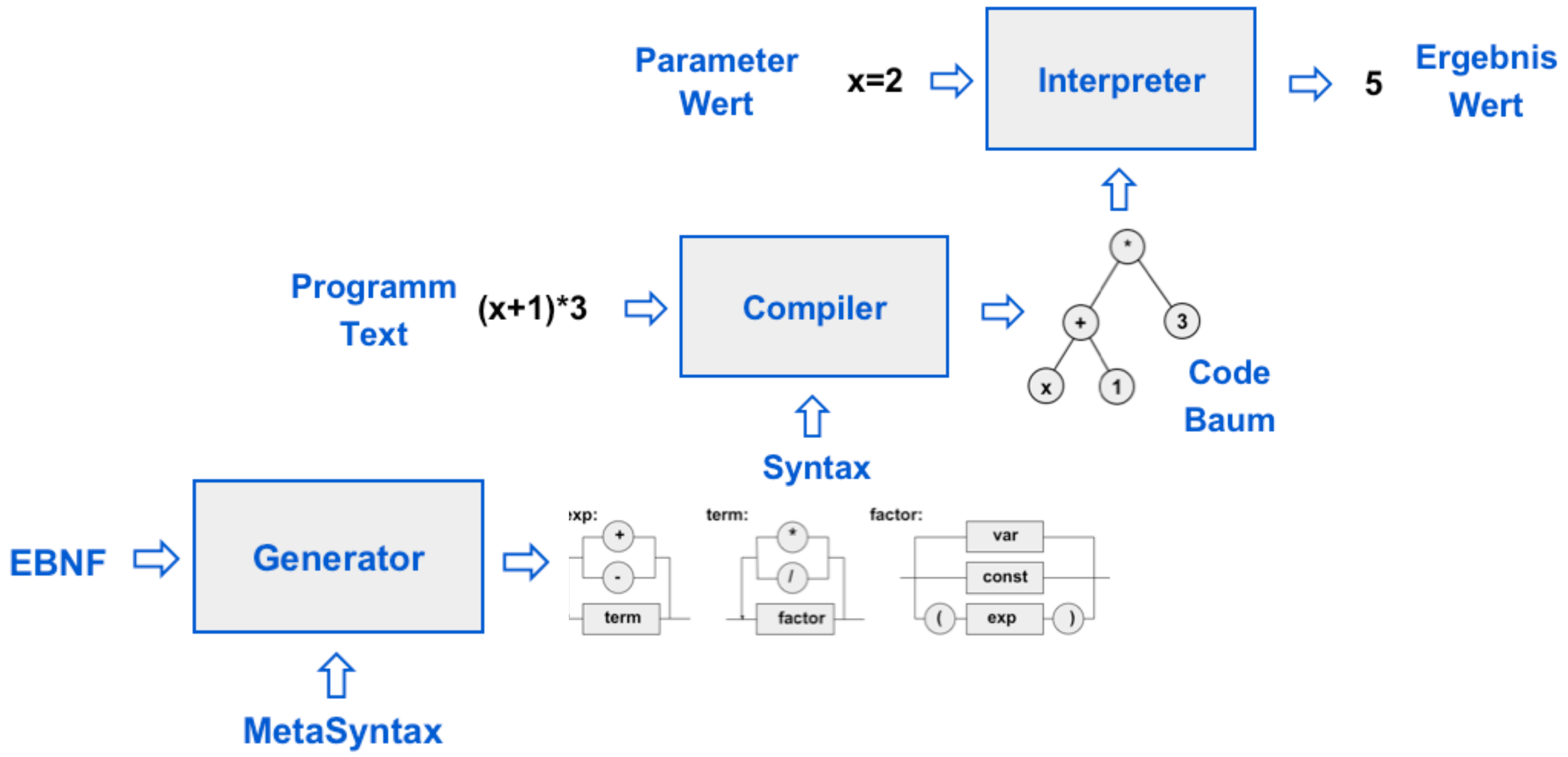
Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Generator



Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Generator, Compiler und Interpreter



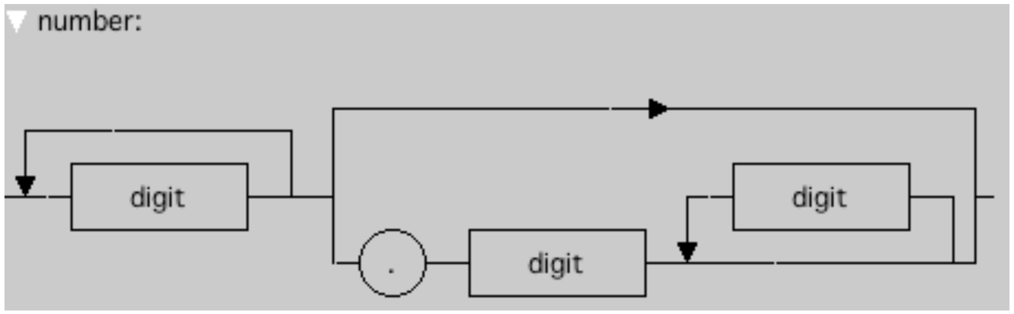
Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Automaten (1)

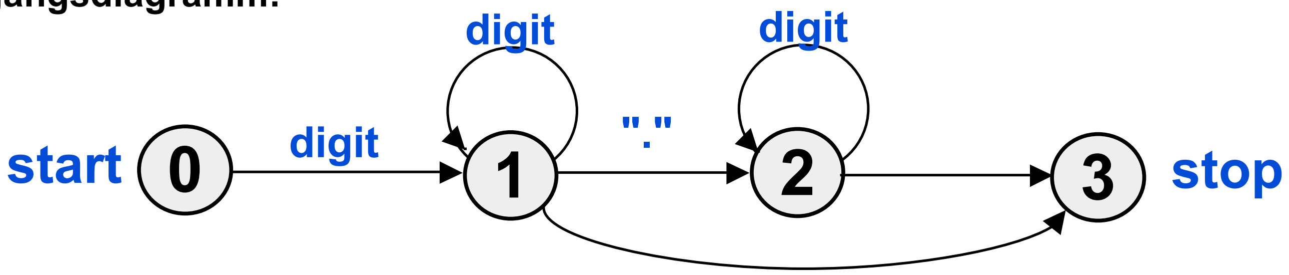
Beispiel: Berechnung des Wertes einer Dezimalzahl mittels eines Automaten

EBNF: number = digit {digit} ["."{digit}]

Syntaxdiagramm:



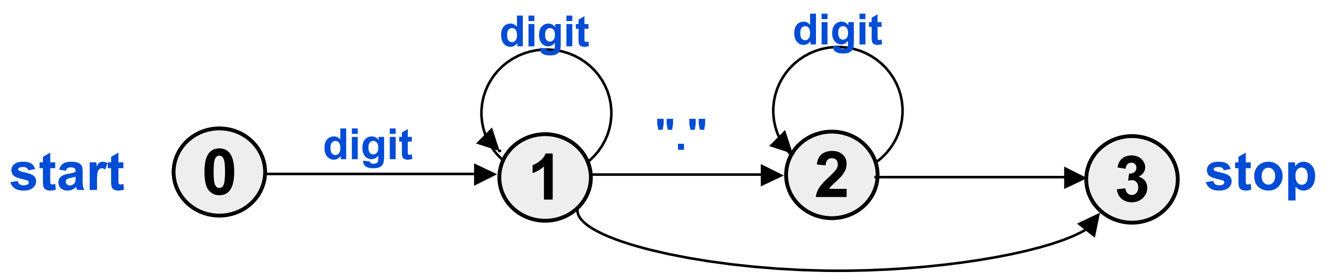
Übergangsdiagramm:



Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Automaten (2)

Übergangsdiagramm:


$$\begin{tikzcd}
\text{start} & 0 \arrow[r, "\text{digit}"] & 1 \arrow[loop above, "\text{digit}"] \arrow[r, "\text{"."}"] \arrow[rr, bend right] & 2 \arrow[loop above, "\text{digit}"] \arrow[r] & 3 & \text{stop}
\end{tikzcd}$$

Zustandstabelle s:

	digit	"."	else
0	1	error	error
1	1	2	3
2	2	error	3

Aktionstabelle a:

	digit	"."	else
0	1	5	5
1	2	0	4
2	3	5	4

Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Automaten (3)

```
void action(int i) throws Exception {
   switch (i) {
         case 1: w = digit; n = 1; break;
         case 2: w = 10*w+digit; break;
         case 3: w = 10*w+digit; n = 10*n; break;
         case 4: result = w/n; break;
         case 5: throw new Exception("bad number");
   }
}
```

Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Automaten (4)

```
final int start = 0;
final int stop = 3;
final int error = 4;
int state = start;
int symbol;
do {
    symbol = nextSymbol();
    action(a[state][symbol]);
    state = s[state][symbol];
} while (state < stop);
action(a[state][symbol]);
```

Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Shift-Reduce Syntaxanalyse (1)

Operator-Precedence:

	var	+	*	empty
var		•>	•>	•>
+	<•	•>	<•	•>
*	<•	•>	•>	•>
empty	<•	<•	<•	

Precedence-Funktionen:

	f	g
var	4	5
+	2	1
*	4	3
empty	0	0

x <• y ⇔ f(x) < g(y)
x •> y ⇔ f(x) > g(y)

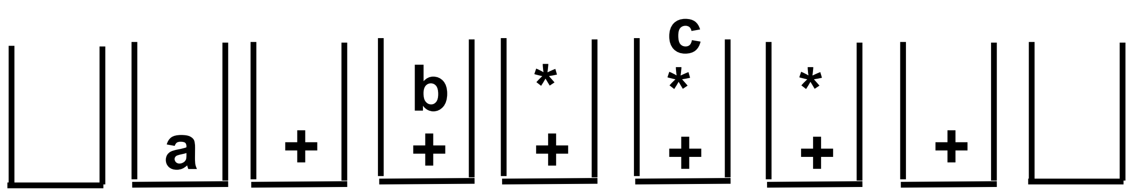
Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Shift-Reduce Syntaxanalyse (2)

```
final String empty = "";
LexicalScanner lexer = new LexicalScanner(text);
String token = lexer.nextToken();
Stack s = new Stack();
s.push(empty);
while (!(s.top().equals(empty)&token.equals(empty))) {
    if (f(s.top())<=g(token)) { // shift
        s.push(token);
        token = lexer.nextToken();
    } else { // reduce
        do {op = s.pop(); action(op);}
        while (f(s.top())<=g(token));
    }
}
```

Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

Shift-Reduce Syntaxanalyse (3)
Beispiel: text = "a+b*c"



Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

InfixAnalyzer (1)

```
class InfixAnalyzer {    // recursive descent syntax analysis:
    public Operand exp() throws Warning {
        Operand op = term();
        while (true) {
            if (nextTokenEquals("-")) op = Operand.genDiff(op,term());
            else if (nextTokenEquals("+")) op = Operand.genSum(op,term());
            else break;
        }
        return op;
    }
```

Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

InfixAnalyzer (2)

```
public Operand term() throws Warning {
   Operand op = factor();
   while (true) {
      if (nextTokenEquals("/")) op = Operand.genQuot(op,factor());
      else if (nextTokenEquals("*")) op = Operand.genProd(op,factor());
      else break;
   }
   return op;
}
```

Helmut Schauer
Educational Engineering Lab
Department for Information Technology
University of Zurich

InfixAnalyzer (3)

```
    public Operand factor() throws Warning {
        Operand op = Operand.UNDEF;
        if (token.equals("")) throw new Warning("factor is missing!");
        if (isNumber()) { // factor = constant
            try {op = Operand.genConst(Integer.parseInt(token));}
            catch(NumberFormatException e) {
                throw new Warning(token+" is not a number!");
            }
            token = lexer.nextToken();
        } else if (isVariable()) { // factor = variable
            op = Operand.genVar(token);
            token = lexer.nextToken();
        } else if (nextTokenEquals("(")) { // factor = ( exp )
            op = exp();
            if (!nextTokenEquals(")")) throw new Warning(") is missing!");
        } else throw new Warning("factor is missing!");
        return op;
    }
}
```