Simula

Die Entstehung der Objektorientierung

Felix Franck 02I/M

Gliederung(1)

- Einführung
  - Situation
- Simula I
  - Grundlagen
  - Syntax
  - Beispiel

Gliederung(2)

- Simula I Simula 67
  - Class Prefixing
- Simula 67
  - Beispiel
- Schlussfolgerungen
  - Softwareentwicklung
  - Vor-/Nachteile
- Kulturelle Einflüsse

Einführung

The simulation application language

1962 - 67 am Norwegian Computer Center (NCC) entwickelt

O.J. Dahl, B. Myhrhaug, K. Nygaard

3 Versionen

- Simula 0
- Simula I
- Simula 67

1987 von der Schwedischen Standardisierungsgruppe festgelegt

Einführung - Situation

- Assembler
- Zeilennummern und „Goto“
- Fortran
- Algol60

Grundlagen Simula(1)
Simulation **La**nguage

Weiterentwicklung von Algol60

- Blockstruktur
  - Prozeduren sind spezialisierte Blöcke
- Laufzeiteffizienter Compiler
- Programmsicherheit
- Europäischer Patriotismus

Grundlagen Simula(2)

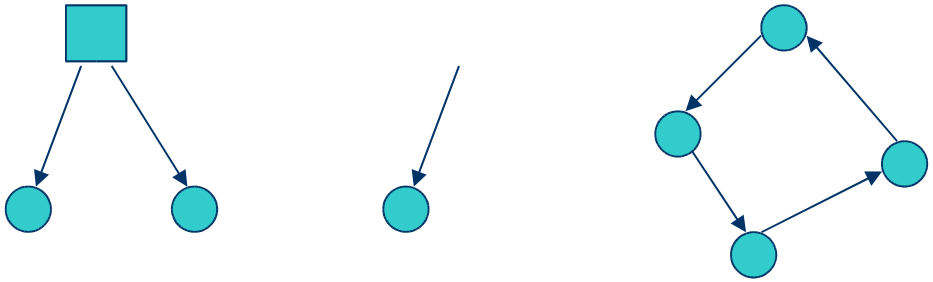
Sprache für Simulationszwecke

- Nebenläufigkeit
- Aufbrechen des LIFO
- frei zugreifbare Listenstruktur
- Sicherheit
  - Striktes Typprüfen
  - Preinitialisierung von Variablen
  - Pointeroperationen beschränken
  - Automatische Deallokation

Prozesse

Anlegen explizit

Löschen implizit



Grundlagen Simula I(3)

Änderungen

- Call by value
- Call by reference
- Variablenbehandlung

Erweiterungen zu Algol60

- **„activity“**
- **„explicit process pointers“**
- **„inspect“** Konstrukt
- Neues Laufzeitsystem

abwärtskompatibilität

Simula Syntax(1)
Datentypen und Operatoren

```
boolean, integer, real, character
+, -, *, /, **
<, <=, >, >=, =, <>
and, or, not, imp, eqv
```

automatische Typumwandlung

Simula Syntax(2) Kontrollanweisungen

- `Begin .. end`
- `for .. step .. until`
- bedingten arithmetischen Ausdruck
  - `i: = if x = 1 then 5 else 6`
- Call by value
- Co-Routine

Simula Syntax(3)
1. Beispiel

```
begin
  real x1,y;

  real procedure t(n,x); integer n; real x;
    t := if n > 1 then 2*x*t(n-1,x)-t(n-2,x)
                  else (if n = 1 then x else 1);

  for x1 := -1 step 0.1 until 1 do
  begin
    y := t(5,x1);
    outfix(x1,2,6); outfix(y,3,10); outimage;
  end;
```

Programmbeispiel Simula I

```
SIMULA begin activity Car;
   begin real X0; T0; V ;
         real procedure X; X := X0+V (time–T0);
         procedure newV (Vnew); real Vnew;
                   begin X0 :=X; T0 := time; V := Vnew end;
                  Car behaviour: . . . . . ; hold(<travel time>);
                           . . . . . .
   end of Car;
   activityPolice;
         begin . . . . . ; inspect <process> when Car do
                  if X <is within city> and V >50 then
                  begin newV (50); <give ne> end; . . . . . .
         end of Police;
         main program: <initialise>; hold(<simulation period>)
end of simulation model;
```

Simula I Simula 67(1)
Simple **u**niversal **la**nguage

Sprache auf Simulation beschränkt

„**inspect**“ –war recht unhandlich

Algol-Compiler für UNIVAC 1107

„activity/process“ Konzept abwandeln

Tony Hoare

- **„record classes“ und „subclasses“**
- Punktnotation

„class prefixing“

Simula I Simula 67(2)
Class Prefixing

Klassen können hierarchisch verbunden werden

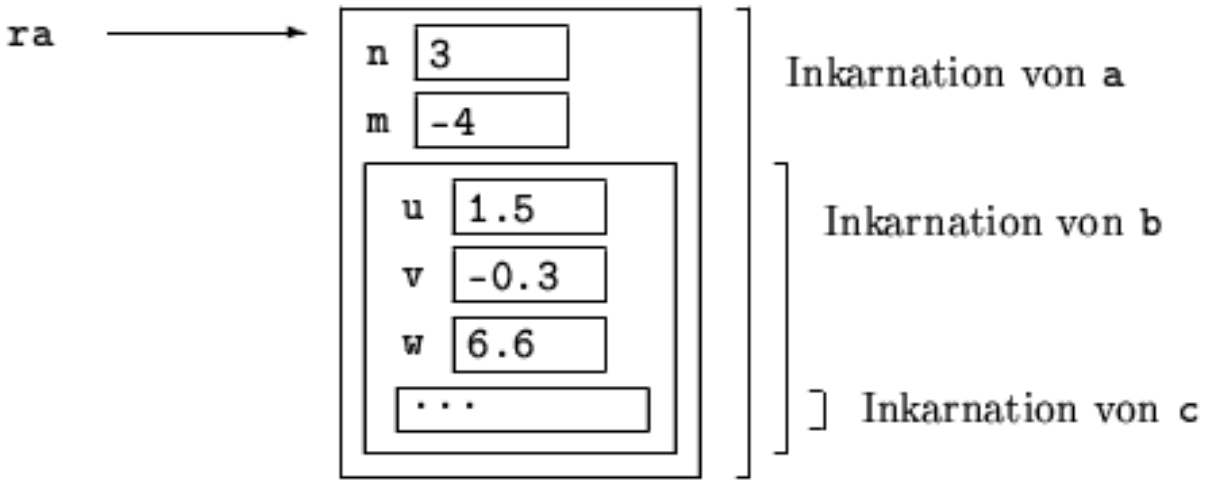
„prefix part“ und **„main part“**

„prefix part“ gehört der übergeordneten Klasse an

Aus **„main part“** hat man vollen Zugriff auf den **„prefix part“** umgekehrt nur durch **„remote accessing“**

Simula I Simula 67(3)
Class prefixing(2)

```
Ref(C) ra
ra :- new C(3, -4, 1.5, -0.3, 6.6)
```



Simula I Simula 67(4)

Wahl der Begriffe: Klasse und Objekte

- **„prefixed class“**
- Objekt ist Instanz einer Klasse
- Variablen lokal zum Äußersten Block werden Attribute genannt

Programmbeispiel Simula 67 (1)

```
class SIMULATION;
begin class process;
   begin real EventTime; NextEvent; . . . . . end;
   ref(process) current;
   comment current points to the currently operating process.
        It is the head of the \time list" of scheduled ones,
        sorted with respect to nondecreasing EventTimes;
   real procedure time; time := current.EventTime;
   procedure hold(deltaT ); real deltaT ;
   begin current.EventTime := time+deltaT ;
        if time  current.NextEvent.EventTime then
        begin ref(process)P; P :-current; current :-P.NextEvent;
                <move P to the right position in the time
   list>;
        resume(current) end end of hold;
   . . . . . .
end of SIMULATION;
```

Programmbeispiel Simula 67 (2)

```
SIMULATION begin
   process class Car;
   begin real X0; T0; V ;
         real procedure X; X := X0+V (time–T0);
         procedure newV (V new); real V new;
           begin X0 :=X; T0 := time; V := V new end;
         Car behaviour: . . . . ; hold(<travel time>); . . . .
   end of Car;
   process Police;
   begin . . . . . ; inspect <process> when Car do
         if X <is within city> and V >50 then
           begin newV (50); <give ne> end; . . . . . .
   end of Police;
   main program: <initialise>; hold(<simulation period>)
end of simulation model;
```

Simula 67 (3)

small-Typen, long-Typen

While, switch

when, is, in, qua

otherwise

this

Nested Classes

Schlussfolgerungen(1)

Daten und **ihre** Methoden Objekt

Bottom-Up

- Verberbung

Top Down

- Virtual Procedures

Simulationsdaten Objekte

Schlussfolgerungen(2)

2 Sichten

- *Von Innen*
  - lokale Variable,
  - eine Invariante,
  - Operationen auf den Variablen
- *Von Außen*
  - *Abstrakt*
  - *Funktionen zum Erstellen des Objekts und*
  - *Funktionen, die über Remoting ansprechbar sind*

Schlussfolgerungen(3) Vor-Nachteile(1)

Vorteile

- Struktur der Codes = gedankliche Struktur
- Fehlerbehandlung
- Nebenläufigkeit
- Sicherheit
- Simulation

Schlussfolgerungen(4) Vor-Nachteile(1)

Nachteile

- Laufzeit
- Übersetzungszeit
- Sicherheit
- OOP
  - Multiple inheritance
  - Interfaces

Auswirkungen

- Keine starke Verbreitung
  - Hoher Preis für Compiler
- Verwendung
- Einfluss auf Weiterentwicklung
  - Abstrakter Typen
  - SmallTalk, C++, …
  - 20 Jahre später – Sprache Beta

Interessante Links

http://heim.ifi.uio.no/~olejohan/birth-of-oo.pdf

ftp://ftp.ifi.uio.no/pub/cim/

- Simula nach C Compiler

Quellen

Günther Lamprecht
„Simula – Eine Einführung“
- Dez. 1999

O.J. Dahl
„ sd&m Software Pioniere“
- Dez. 1999

Bjarne Stroustrup
„The Design and Evolution of C++
- 1994

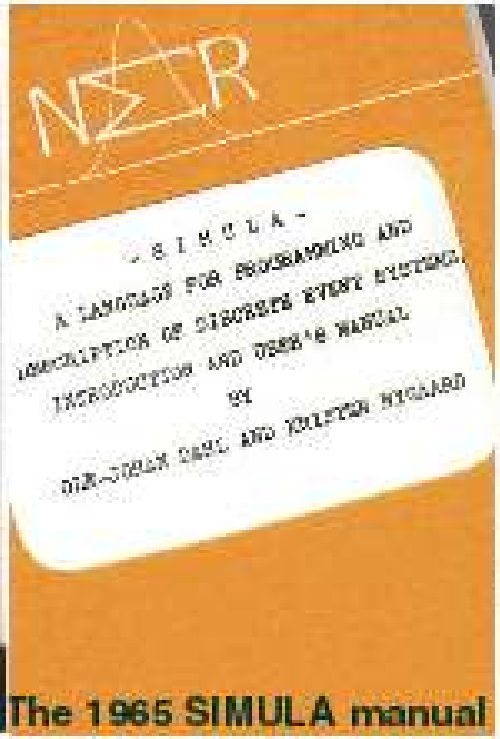
http://heim.ifi.uio.no

Danke

für ihre Aufmerksamkeit!

Fragen?

Ole-Johan Dahl (1978)



The 1965 SIMULA manual
The first object-oriented report
(1965)

Kristen Nygaard (1978)