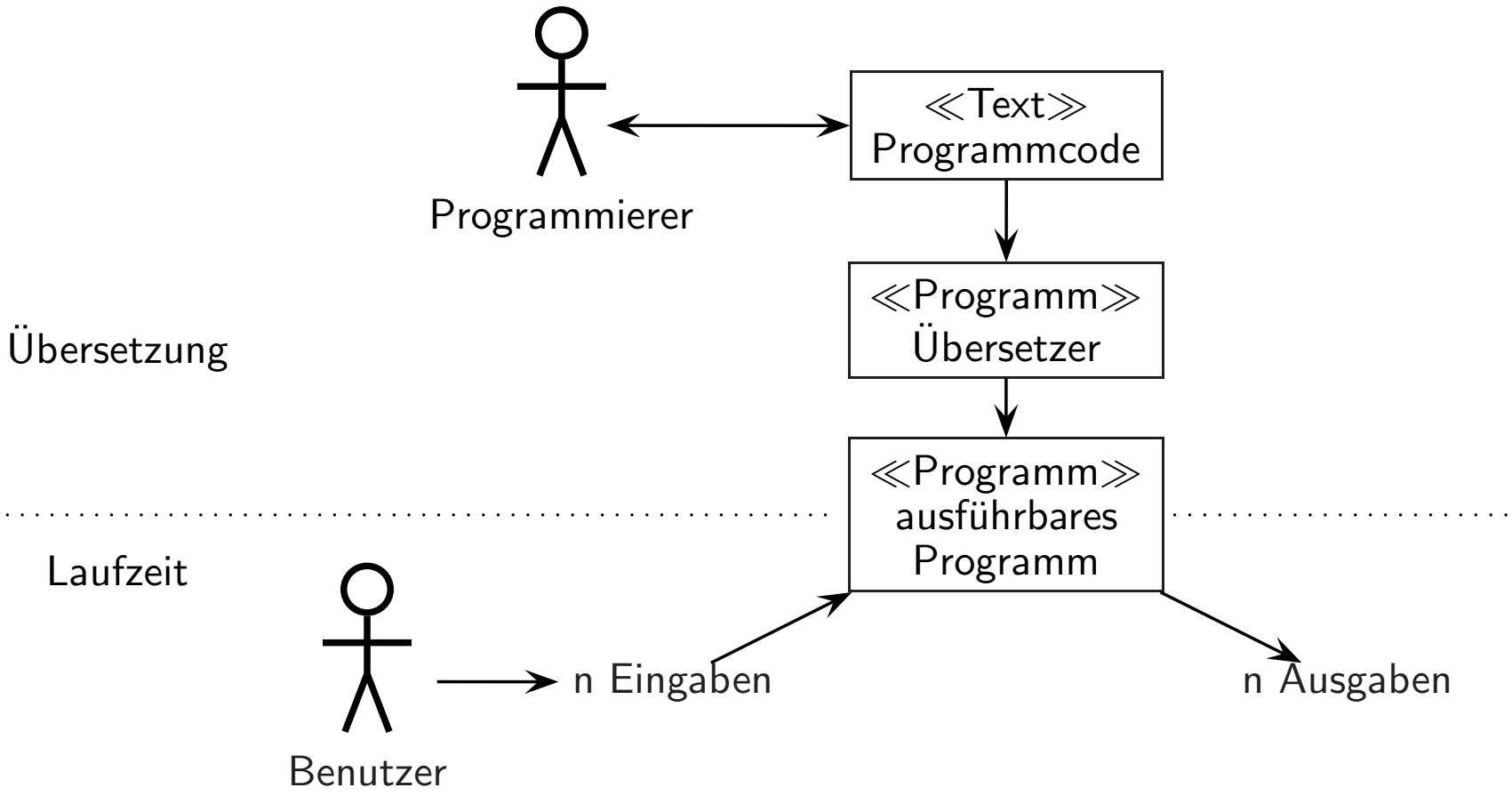
Koreferenz und Anaphorik in einer Programmiersprache: Beispiele, Aussichten und Probleme

Sebastian Lohmeier
sl@monochromata.de

16. Mai 2012

Übersicht

Programmierumgebung



Beispielprogramm

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {









}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Beispielprogramm

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  Run run = new Run();
```

```
  return run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Beispielprogramm

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  Run run = new Run();
  MachineListener
     listener=user.getListener();
  setListener(listener);




  removeListener(listener);
  return run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Beispielprogramm

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  Run run = new Run();
  MachineListener
     listener=user.getListener();
  setListener(listener);
  start(run, clothes.getDescription());
  wash(run);
  spinDry(run);
  finish(run, clothes.getDescription());
  removeListener(listener);
  return run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Beispielausgabe

```
start(run, clothes.getDescription());
wash(run);
spinDry(run);
finish(run, clothes.getDescription());

Run 1: started washing my socks
Run 1: washing
Run 1: spinning dry
Run 1: finished washing my socks
```

Referenz im Beispielprogramm: Variablen

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  Run run = new Run();
  MachineListener
     listener=user.getListener();
  setListener(listener);
  start(run, clothes.getDescription());
  wash(run);
  spinDry(run);
  finish(run, clothes.getDescription());
  removeListener(listener);
  return run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Direkte Anaphern

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  Run run = new Run();
  MachineListener
     listener=user.getListener();
  setListener(.Listener);
  start(run, clothes.getDescription());
  wash(run);
  spinDry(run);
  finish(run, clothes.getDescription());
  removeListener(.Listener);
  return run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Koreferenz

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  Run run = new Run();
  MachineListener
     listener=user.getListener();
  setListener(.Listener);
  start(run, clothes.getDescription());
  wash(run);
  spinDry(run);
  finish(run, clothes.getDescription());
  removeListener(.Listener);
  return run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Indirekte Anaphern

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  Run run = new Run();
  MachineListener
     listener=user.getListener();
  setListener(.Listener);
  start(run, clothes.getDescription());
  wash(run);
  spinDry(run);
  finish(run, clothes.getDescription());
  removeListener(.Listener);
  return run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Indirekte Anaphern

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  Run run = new Run();
  MachineListener
     listener=user.getListener();
  setListener(.Listener);
  start(run, .Description);
  wash(run);
  spinDry(run);
  finish(run, clothes.getDescription());
  removeListener(.Listener);
  return run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Indirekte Anaphern

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  Run run = new Run();
  MachineListener
     listener=user.getListener();
  setListener(.Listener);
  start(run, .Description);
  wash(run);
  spinDry(run);
  finish(run, .Description);
  removeListener(.Listener);
  return run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Weitere direkte Anaphern

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  Run run = new Run();
  MachineListener
     listener=user.getListener();
  setListener(.Listener);
  start(run, .Description);
  wash(run);
  spinDry(run);
  finish(run, .Description);
  removeListener(.Listener);
  return run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Weitere direkte Anaphern

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  new Run();
  MachineListener
     listener=user.getListener();
  setListener(.Listener);
  start(.Run, .Description);
  wash(.Run);
  spinDry(.Run);
  finish(.Run, .Description);
  removeListener(.Listener);
  return .Run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Eine direkte in eine indirekte Anapher umwandeln

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  new Run();
  MachineListener
     listener=user.getListener();
  setListener(.Listener);
  start(.Run, .Description);
  wash(.Run);
  spinDry(.Run);
  finish(.Run, .Description);
  removeListener(.Listener);
  return .Run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Eine direkte in eine indirekte Anapher umwandeln

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  new Run();


  setListener(.Listener);
  start(.Run, .Description);
  wash(.Run);
  spinDry(.Run);
  finish(.Run, .Description);
  removeListener(.Listener);
  return .Run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Zum Vergleich: die initiale Version

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  Run run = new Run();
  MachineListener
     listener=user.getListener();
  setListener(listener);
  start(run, clothes.getDescription());
  wash(run);
  spinDry(run);
  finish(run, clothes.getDescription());
  removeListener(listener);
  return run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Aussichten

- Kürzerer Code durch referentielle Unterspezifikation
- Interessanterer Code durch Kohärenzlücken
- Keine komplizierten „künstlichen“ Sprachfunktionen
- Praktische Anwendung linguistischer Modelle wie Schwarz (2000)

Probleme: Referentielle Ambiguität

```
class WashingMachine {
 Run wash(Clothes c1, Clothes c2, User
    user) {
  new Run();
  setListener(.Listener);
  start(.Run, .Description);
  wash(.Run);
  spinDry(.Run);
  finish(.Run, .Description);
  removeListener(.Listener);
  return .Run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my
     trousers")),
  new Clothes(new Description("my socks")),
  new User());
```

Probleme: Referentielle Ambiguität

```
class WashingMachine {
 Run wash(Clothes c1, Clothes c2, User
    user) {
  new Run();
  setListener(.Listener);
  start(.Run, .Description);
  wash(.Run);
  spinDry(.Run);
  finish(.Run, .Description);
  removeListener(.Listener);
  return .Run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my
     trousers")),
  new Clothes(new Description("my socks")),
  new User());
```

Probleme: Kontrollstrukturen

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  if(clothes.needCleaning()) {
    new Run();
    setListener(.Listener);
    start(.Run, .Description);
    wash(.Run);
    spinDry(.Run);
    finish(.Run, .Description);
    removeListener(.Listener);
  }
  return .Run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Probleme: Kontrollstrukturen

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  if(clothes.needCleaning()) {
    new Run();
    setListener(.Listener);
    start(.Run, .Description);
    wash(.Run);
    spinDry(.Run);
    finish(.Run, .Description);
    removeListener(.Listener);
  }
  return .Run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Probleme: Fokus

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  new Run();
  setListener(.Listener);
  wash(.Run);
  spinDry(.Run);
  if(clothes.stillDirty()) {
    new Run();
    wash(.Run);
    spinDry(.Run);
  }
  removeListener(.Listener);
  return .Run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Probleme: Fokus

```
class WashingMachine {
 Run wash(Clothes clothes, User user) {
  new Run();
  setListener(.Listener);
  wash(.Run);
  spinDry(.Run);
  if(clothes.stillDirty()) {
    new Run();
    wash(.Run);
    spinDry(.Run);
  }
  removeListener(.Listener);
  return .Run;
}}
new WashingMachine().wash(
  new Clothes(new Description("my socks")),
  new User());
```

Grundlegende Probleme

- Lernen durch Lesen bei Unterspezifikation
- Deterministisch vs. probabilitisch
- Natürliche Kategorien: Ross u. Makin (1999) vs. Knöll u. a. (2011)
- Fokus und Aktivierung: Engle u. Oransky (1999)
- Naturalistische Programmierung (Lopes u. a., 2003; Knöll u. Mezini, 2006): Empirische Beurteilung der Natürlichkeit
- Berechenbarkeit / Komplexität / Optimalität
- Isolierbarkeit einzelner Funktionalitäten natürlicher Sprachen

Diskussion

*

Dokumentation, Quellcode, RSS-Feed & E-Mail Newsletter:
http://www.monochromata.de/lop

Diaspora-Handle:
monochromata@pod.geraspora.de

Literatur I

[Engle u. Oransky 1999] Engle, Randall W. ; Oransky, Natalie: Multi-Store versus Dynamic Models of Temporary Storage in Memory. In: **(Sternberg, 1999)**, S. 515–555

[Knöll u. a. 2011] Knöll, Roman ; Gasiunas, Vaidas ; Mezini, Mira: Naturalistic Types. In: *ONWARD '11: Proceedings of the 10th SIGPLAN symposium on New ideas, new paradigms, and reflections on programming and software*, 2011, S. 33–47

[Knöll u. Mezini 2006] Knöll, Roman ; Mezini, Mira: Pegasus: first steps toward a naturalistic programming language. In: *OOPSLA '06: Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications*. New York, NY, USA : ACM, 2006, 542–559

[Lopes u. a. 2003] Lopes, Cristina V. ; Dourish, Paul ; Lorenz, David H. ; Lieberherr, Karl: Beyond AOP: toward naturalistic programming. In: *SIGPLAN Not.* 38 (2003), Nr. 12, 34–43. `http://dx.doi.org/10.1145/966051.966058`

Literatur II

[Ross u. Makin 1999] Ross, Brian H. ; Makin, Valerie S.: Prototype versus Exemplar Models in Cognition. In: **(Sternberg, 1999)**, S. 205–241

[Schwarz 2000] Schwarz, Monika: *Indirekte Anaphern in Texten*. Tübingen : Niemeyer, 2000

[Sternberg 1999] Sternberg, Robert J. (Hrsg.): *The Nature of Cognition*. Cambridge, MA et al. : MIT Press, 1999