

Konzepte und Nutzen Aspektorientierter Programmierung (AOP)



Christoph Knabe

TFH Berlin

24.11.2009

Gliederung

- Vortrag Böhm
- Anwendungsgebiete
- Demonstrationen
- AOP-Plattformauswahl
- Fazit

Anwendungsgebiete statisch

□ Statische Prüfungen mittels

```
declare warning|error
```

□ Schichtverletzungen

```
declare error
```

```
: call(* fb6.*.db.*.*(..)) && within(fb6.*.ui.*)  
: "Do not call the db-layer directly";
```

□ Constant/Mutable-Konventionen

```
declare warning: set(@Constant * *.*.)
```

```
: "Do not change @Constant field";
```

□ Ausnahmepropagierung

checked → unchecked exception

Anwendungsgebiete dynamisch

- Logging, Profiling
- @Nullable-Prüfungen
 - Nur als @Nullable markierte Referenzen dürfen `null` erhalten!
- Ausnahmemeldung
 - Swing, Struts, JSF usw. haben verschiedene Techniken für zentrale Ausnahmemelder.
 - Einfacher und gleicher mit Aspekten
- Autorisierung
 - Methodenaufrufeinschränkung je nach Benutzergruppe
- Persistenzschicht
 - „dirty“-Flag mittels `set-Pointcut`
 - „lazy loading“ mittels `get-Pointcut`

Welche AOP-Plattform?

- AspectJ 5
- Spring 2.0 mit Spring AOP
- AspectWerkz
- JBoss AOP
- u.v.a

AspectJ 5

- **+ Original:** Hier wurden die AOP-Konzepte entwickelt
- **+ Statische Prüfungen** durch eigene Kompilation
- **+ lesbar:** Aspekte leichter lesbar
- **+ IDE:** Gute Integration in Eclipse
 - z.B. Markierung aller beeinflussten Stellen
- **- Compiler:** Eigener Compiler in bestimmten Umgebungen nicht durchsetzbar

AspectWerkz

- **+ Standard-Compiler**
- **+ Original** der compilerfreien AOP-Implementierungen
- **- Laufzeit**: Alle Prüfungen erst zur Laufzeit
- **- XML**: Schwer lesbare XML-Syntax

Spring AOP

- **+ Standard-Compiler**
- **- Laufzeit:** Prüfungen erst zur Laufzeit
- **- XML:** Schwer lesbare XML-Syntax
- **+ Annotationen:** Mittel lesbare Syntax
- **+ AspectJ-Kopie:** Pointcut-Muster gleich wie in AspectJ

Fazit

- Geschäftslogik in Java, technische Instrumentierung mittels Aspekten
- Höchst mächtige Technik
 - Log4j, Application Server, OO-DBs ... werden arbeitslos.
- Leicht Wechselwirkungen der Aspekte
 - Vorsichtig einsetzen!
- Noch keine etablierten Aspect-Patterns
- Aspekten gehört die Zukunft