

Clone-based Reengineering für Java auf der Eclipse-Plattform

Geplante Diplomarbeit
Simon Giesecke

Klone

- allgemein: Redundanz in einem (Software-)System
 - ⇒ duplizierte Code-Fragmente (Code Clones)
 - ⇒ duplizierte Konzepte (Concept Clones)
- Unerwünscht, insb. bei agilen Entwicklungsmethoden
 - ⇒ XP: OAOO (Do it once and only once.)
 - ⇒ PP: DRY (Don't repeat yourself.)

Beispiel (1)

```
String getValue(String searchKey) {  
    Enumeration enum =  
        properties.keys();  
    while (enum.hasMoreElements()) {  
        String currentKey =  
            (String)enum.nextElement();  
  
        if (currentKey.startsWith  
            (searchKey))  
            return (String)  
                properties.get(currentKey);  
    }  
    return null;  
}
```

```
String getValueIgnoreCase(String key) {  
    Enumeration keyEnum =  
        properties.keys();  
    while (keyEnum.hasMoreElements()) {  
        String currentKey =  
            (String)keyEnum.nextElement();  
  
        if (currentKey.equalsIgnoreCase  
            (key))  
            return (String)  
                properties.get(currentKey);  
    }  
    throw new IllegalArgumentException("Key " +  
        key + " not found");  
}
```

Beispiel (2)

```
public String getValue(String searchKey) {
    return getValueGeneric(searchKey,
        new Condition() {
            boolean check(String a, String b) {
                return a.startsWith(b);
            }
        });
}
```

```
public String getValueIgnoreCase(String key) {
    return getValueGeneric(key,
        new Condition() {
            boolean check(String a, String b) {
                return a.equalsIgnoreCase(b);
            }
        });
}
```

```
private interface Condition {
    boolean check(String a, String b);
}

private String getValueGeneric(String searchKey, Condition condition) {
    Enumeration enum = properties.keys();
    while (enum.hasMoreElements()) {
        String currentKey = (String) enum.nextElement();

        if (condition.check(currentKey, searchKey))
            return (String) properties.get(currentKey);
    }
    return null;
}
```

Klonentstehung

- Copy & Paste (& Modify!)
- routinemäßige Implementierung
 - ⇒ z.B. Durchlaufen von Listen
- Mängel im Design, falsche Kapselung
 - ⇒ bedingte Konstrukte statt Polymorphie
- Zusammenführung mehrerer Systeme

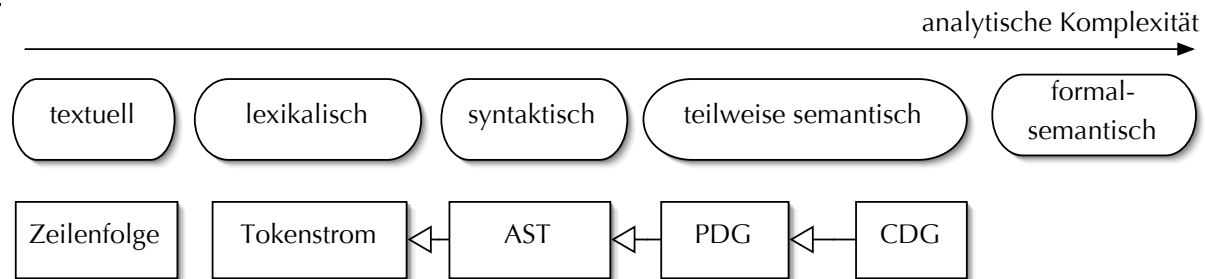
Probleme durch Klone

- erschweren Änderungen
 - ⇒ einige Clones werden bei Änderungen übersehen
- zerstreuen Anwendungslogik über das System
 - ⇒ erschweren Verständnis
- vergrößern das System

Clone-based Reengineering

- umfasst Identifikation und Entfernung von Klonen
- Aktivität
 - ⇒ beim klassischen Reengineering
 - ⇒ bei agilen Entwicklungsprozessen (XP)
- System- vs. Entwickler-orientierter Ansatz

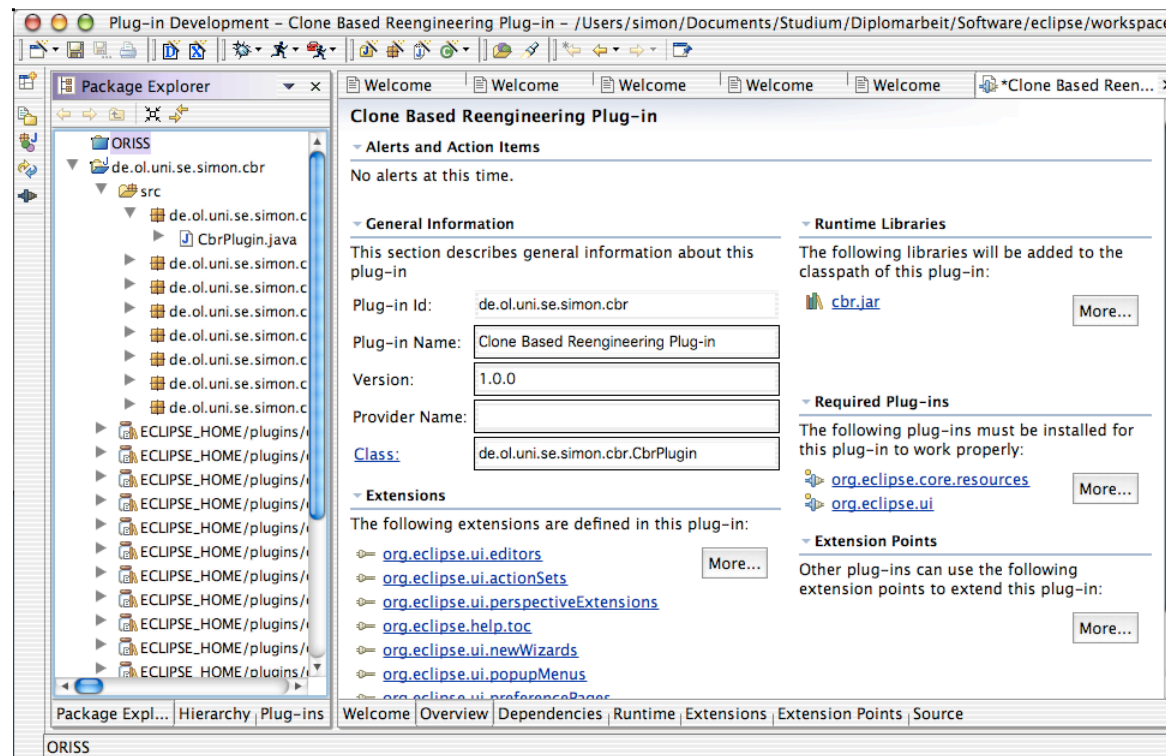
Werkzeuge: Klonerkennung



Ziele der Arbeit

- CBR-Werkzeug für die Eclipse-Plattform
- besondere Berücksichtigung der Einbindung in den Entwicklungsprozess
 - ⇒ Entwickler-orientierter Ansatz
 - ⇒ inkrementelle Duplikationserkennung
- weitgehende Unabhängigkeit von bestimmten Algorithmen zur Duplikationserkennung und -entfernung

Eclipse PDE



Kontakt

E-Mail: simon.giesecke@acm.org

Web: <http://www.informatik.uni-oldenburg.de/~matrix/da/>