

MOD2-SCM: Eine modellgetriebene Produktlinie für Softwarekonfigurations- verwaltungssysteme

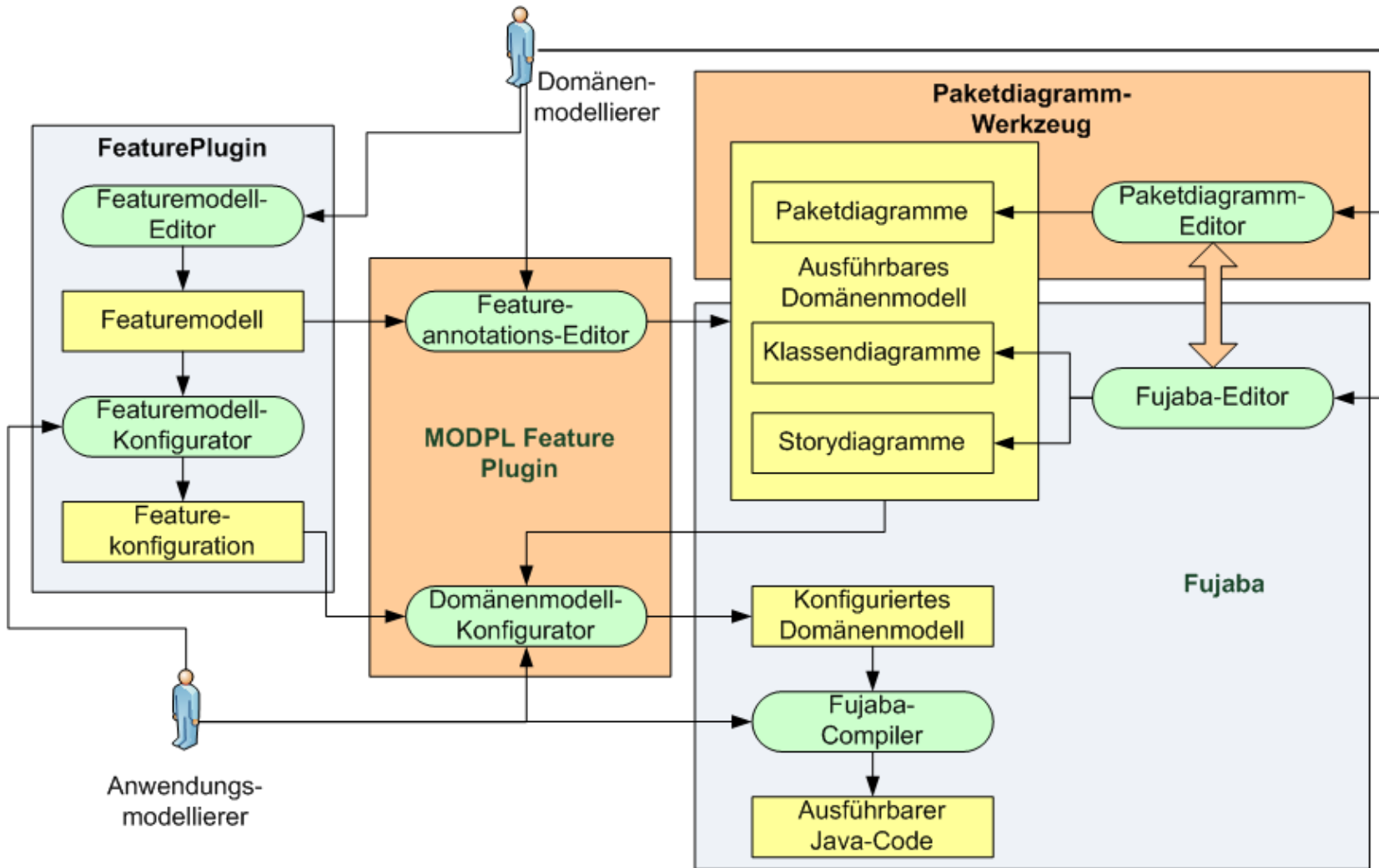
Thomas
Buchmann

Alexander
Dotor

Motivation

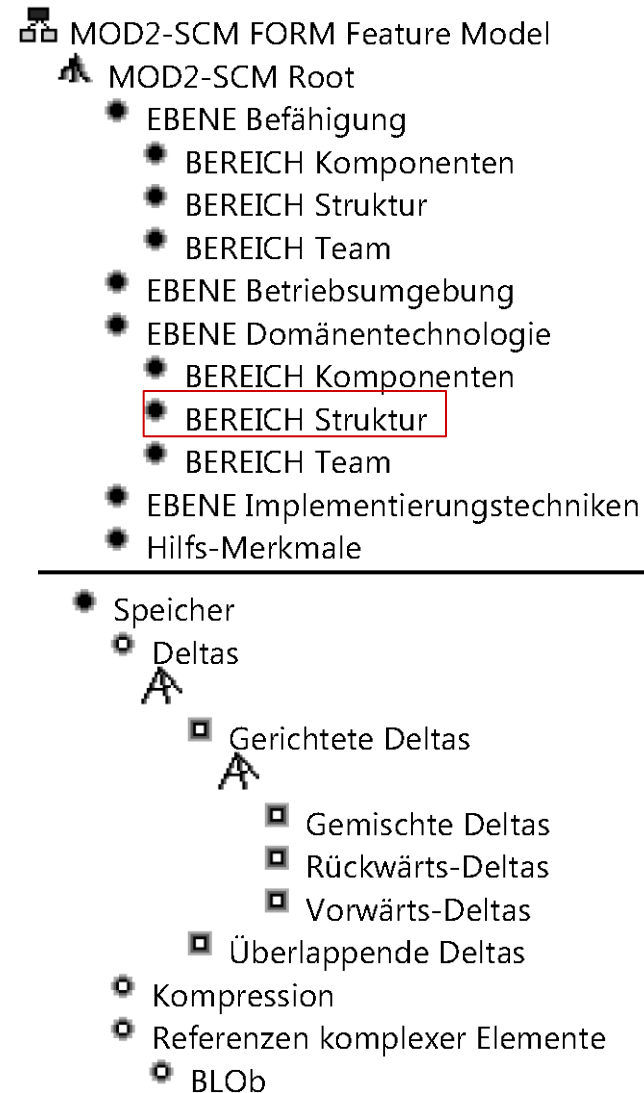
- Mehr als 70 SKMS
- Viele Gemeinsamkeiten
- Nicht OO implementiert
 - Schwer Erweiterbar
 - Kaum Wiederverwendung
- Modellgetriebene Produktlinie für SKMS
 - Wiederverwendung beschleunigt Entwicklung
 - Systematischer Vergleich von Verfahren
 - Explizite Dokumentation durch Modelle

MODPL



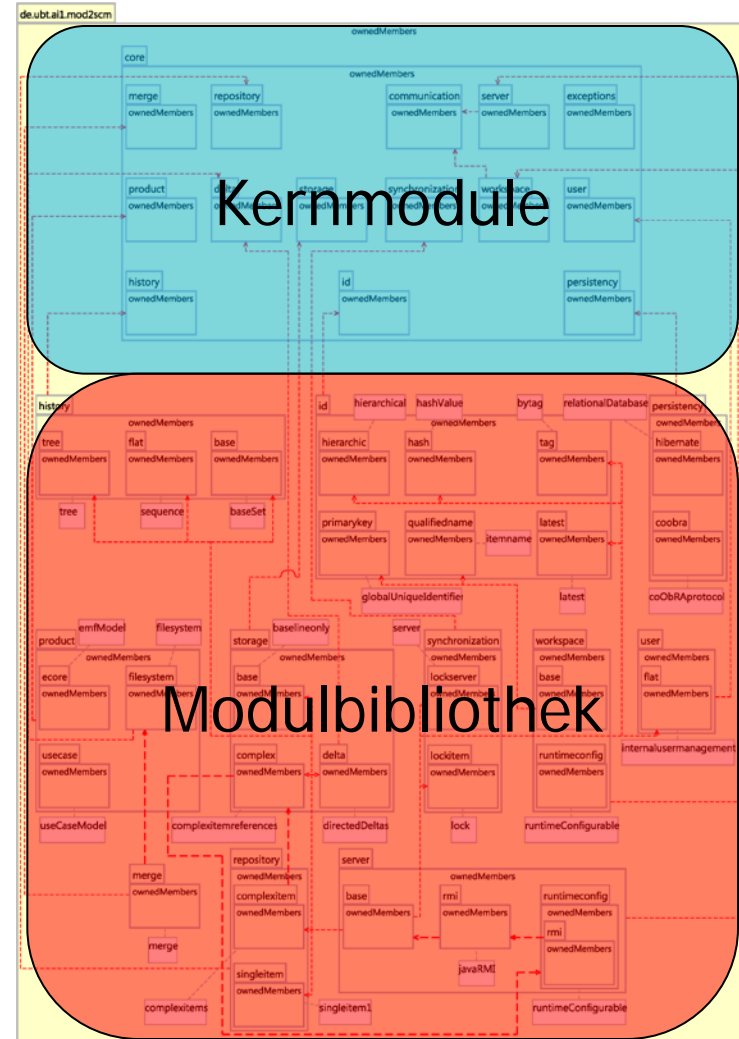
Merkmalsmodell

- Quellen
 - Existierende SKMS inkl. Dokumentation
 - Wissenschaftliche Arbeiten aus der Domäne
- 75 Merkmale
- 34 Einschränkungen

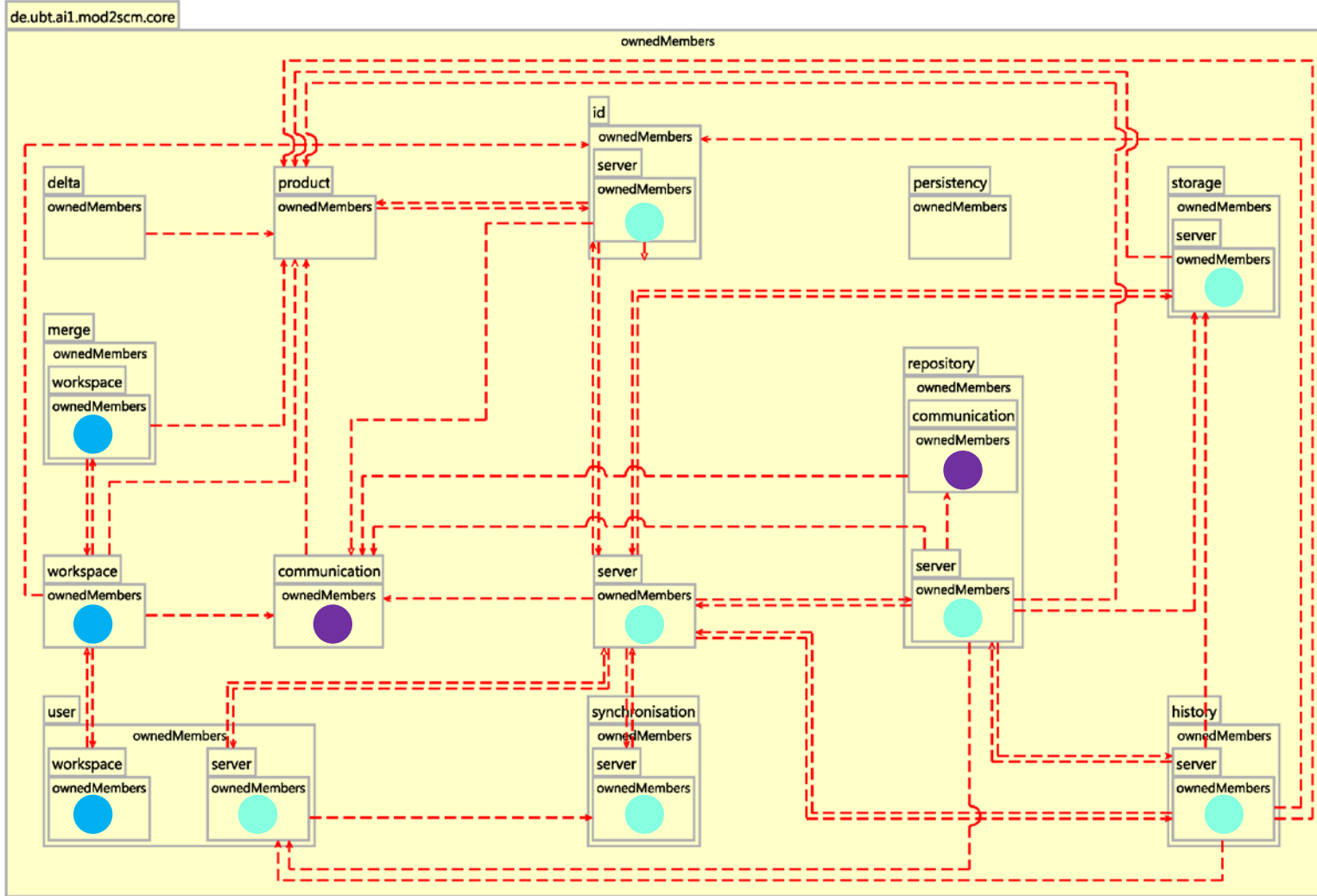


Architektur

- Client-Server-Architektur
- Kern-Bibliotheksmodule-Trennung
- Ziel: Lose Kopplung der Module



Kernmodule



Abhängigkeiten

- Abhängigkeiten
 - Gewollte und ungewollte Abhängigkeiten
- Private Paketimporte
 - B Attributtyp in A
 - B Rückgabebetyp in A
 - B Parametertyp in A
 - B Oberklasse von A
 - B Objekttyp in A

Kopplung im Modell

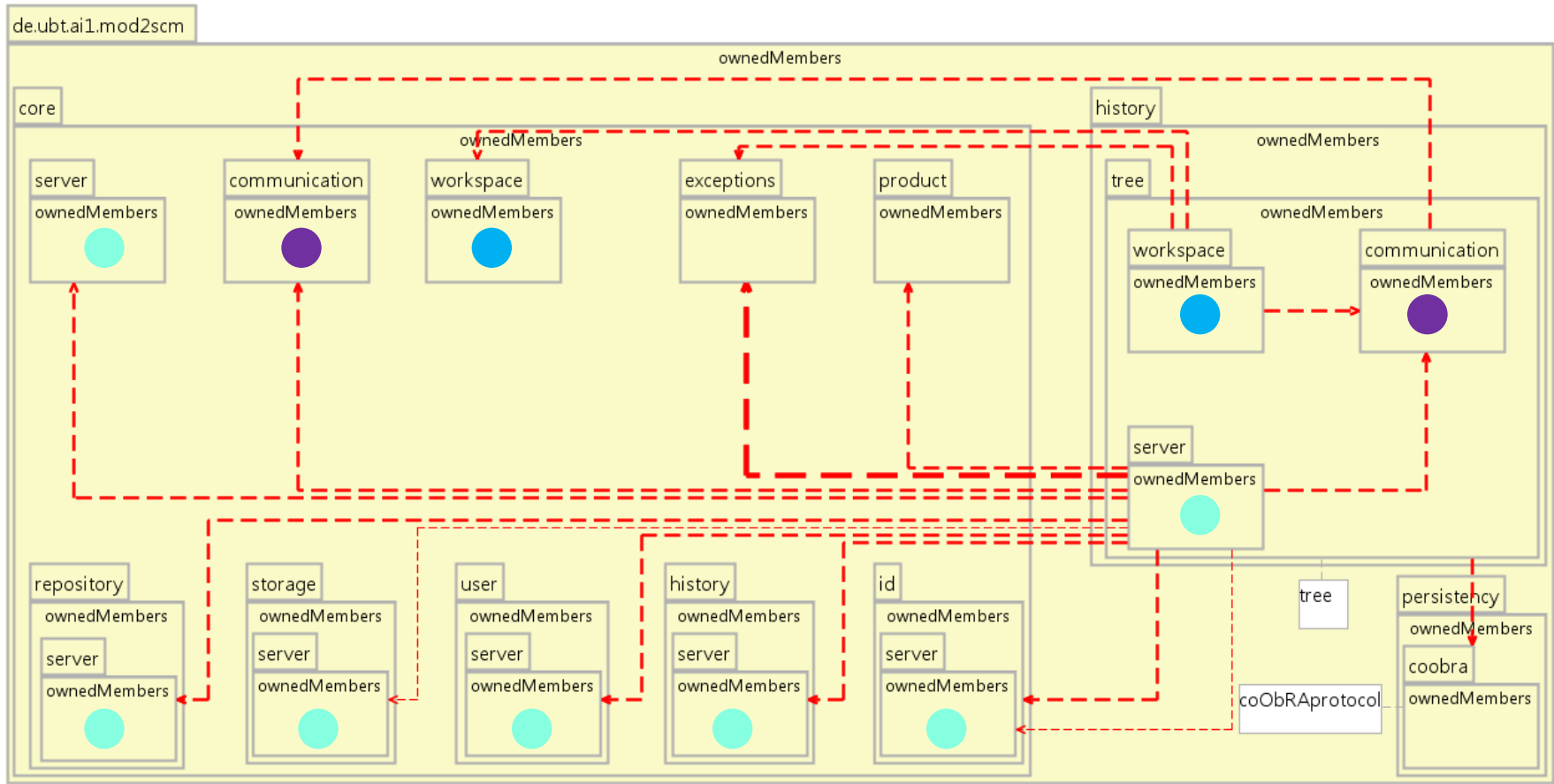
$$\begin{aligned} k(A,B) = & | \text{ B Attributtyp in A } | \\ & + | \text{ B Rückgabebetyp in A } | \\ & + | \text{ B Parametertyp in A } | \\ & + | \text{ B Oberklasse von A } | \\ & + | \text{ B Objekttyp in A } | \end{aligned}$$

Kopplungsstärke
zwischen
den Klassen
A und B

$$k(a,b) = \sum_{x=1}^{|a|} \sum_{y=1}^{|b|} k(A_x, B_y)$$

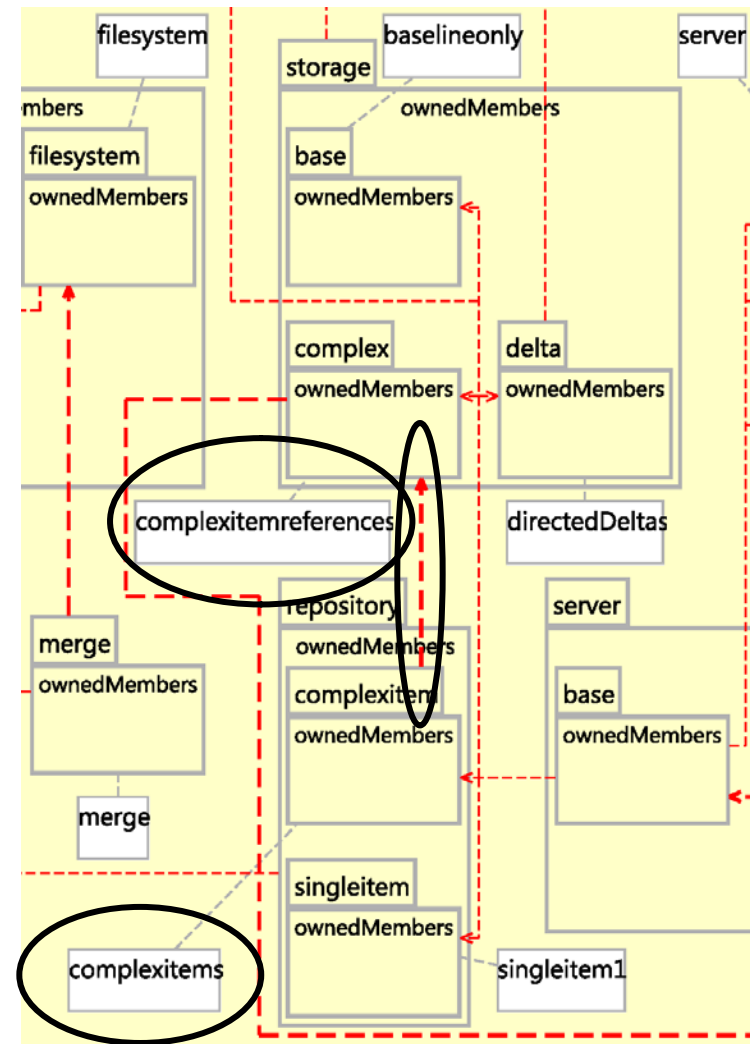
Kopplungsstärke
zwischen
den Paketen
a und b

Beispiel: Modul History



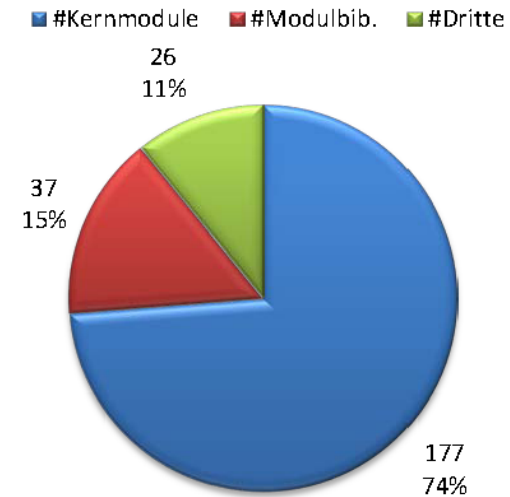
Domänenmodell

- 84 Pakete mit 20 Markierungen (vom Modellierer hinzugefügt)
- 136 Klassen mit 8 Markierungen
- Methoden mit 7 Markierungen und 22 Markierungen an Aktivitäten (Methoden werden durch Story Diagramme modelliert)



Ergebnis

- Analyse/Reduzierung der Abhängigkeiten
- Mehrere Rollen für Domänenentwickler
- Inkrementelle Prozesse
 - Erweiterung Merkmalsmodell
 - Abhängigkeiten in Merkmals-/Domänenmodell



Offene Punkte

- Merkmalsabhängige Architekturen in einem Domänenmodell (Peer-to-Peer)
- Sichten auf Merkmals- und Domänenmodell (Domänen- vs. Produktentwickler)
- Testen aller konfigurierbaren SKMS ?
- Produktlinie für die Benutzeroberfläche

Eclipse Client

The screenshot shows the Eclipse IDE interface. The main editor displays the following Java code:

```
public class HelloWorld {  
  
    public static void main(String[] args) {  
        System.out.println("Hello World! How  
    }  
}
```

The Package Explorer on the right shows the project structure:

- MOD2SCM-Repository
 - de.ubt.ai1.test
 - classpath
 - project
 - settings
 - src
 - HalloWelt.java
 - de.ubt.ai1.helloworld
 - classpath
 - project
 - settings
 - src
 - HelloWorld.java

The Team context menu is open, showing the following options:

- New
- Go Into
- Open in New Window
- Open Type Hierarchy (F4)
- Show In (Alt+Shift+W)
- Copy (Ctrl+C)
- Copy Qualified Name
- Paste (Ctrl+V)
- Delete (Delete)
- Build Path
- Source (Alt+Shift+S)
- Refactor (Alt+Shift+T)
- Import...
- Export...
- Refresh (F5)
- Close Project
- Close Unrelated Projects
- Assign Working Sets...
- Run As
- Debug As
- Team (highlighted)
- Compare With
- Restore from Local History...
- Properties (Alt+Enter)

The Team sub-menu is open, showing the following options:

- Apply Patch...
- Update to Revision / Tag
- Update
- Synchronize with MOD2SCM
- Disconnect
- Connect
- Commit
- Branch
- Merge
- Tag
- Show History
- Unlock
- Lock
- Add to Version Control

The MOD2SCM History view at the bottom shows the following table:

Revision	Project	Author	Comment
M2.1.3	\de.ubt.ai1.hell...	Admin	console-output added
M2.1.2	\de.ubt.ai1.hell...	Admin	main-method
M2.1.1	\de.ubt.ai1.hell...	Admin	HelloWorld Project started

Vielen Dank für ihre
Aufmerksamkeit