Diplomarbeit

Ein Design- Tool für objektorientierte portable Programmierschnittstellen

Vortragender:
Elias Volanakis

Inhalt

1. Entwurfsmuster
2. Wrapper Facade (WF)
   - Aufgaben & Struktur
   - Implementierung
   - Nutzen & Herausforderungen
3. Toolgestützte WF Entwicklung
   - Bestandteile
   - Features
   - Vorteile
4. Demo-Session
5. Zusammenfassung

Entwurfsmuster

allgemein verwendbare Problemlösung
für wiederkehrendes Entwurfsproblem

hier: Entwurfsprobleme objektorientierter Software

Besteht aus:

- Namen
- Problembeschreibung
- Problemlösung
- Analyse der Konsequenzen

Wrapper Facade Entwurfsmuster

Aufgaben:

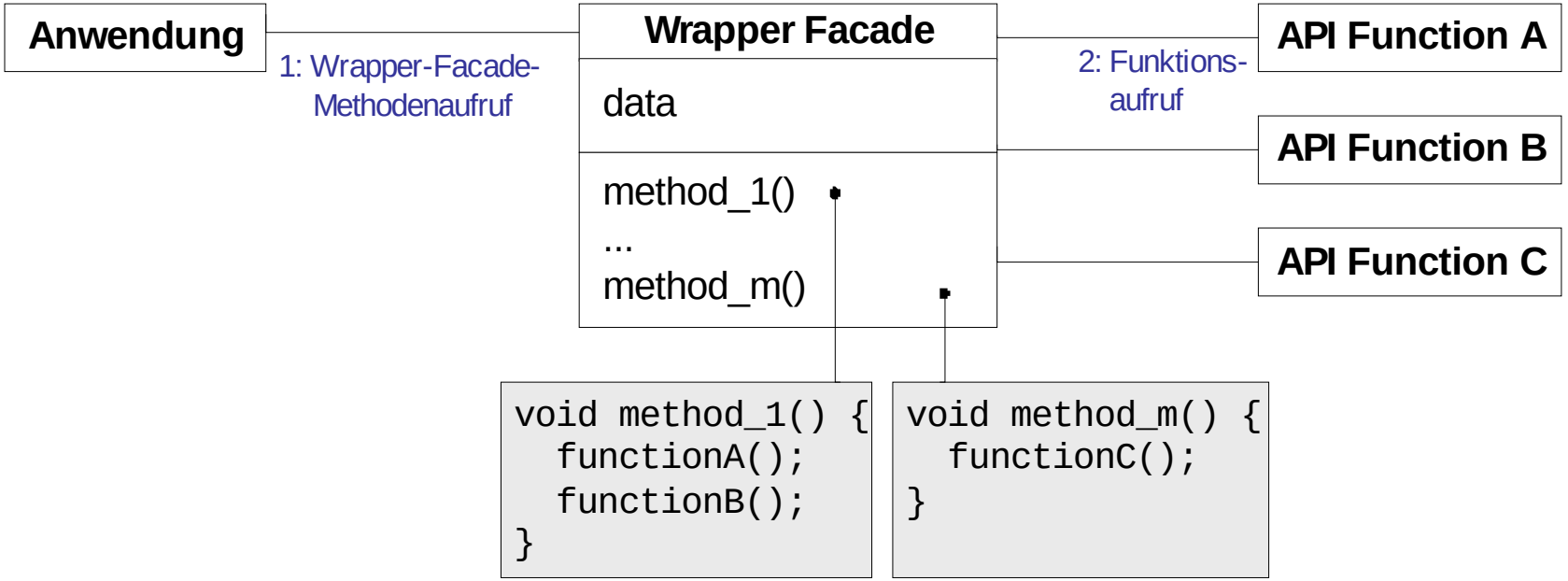
- kapselt systemnahe prozedurale APIs
- bietet stattdessen objektorientierte Schnittstelle mit höherem Abstraktionsgrad

Portabilität?

Zwei Vorraussetzungen:

- Einheitliche Schnittstelle auf allen Systemen
- Austausch systemabhängiger Codestücke

Struktur der Wrapper Facade



Anmerkungen:

1. Bietet einheitliche Schnittstelle
2. Hier unbehandelt: Austausch systemabhängiger Programmteile
3. Keine feste Struktur zwischen den „Teilnehmern“

Schichtenbasierte Betrachtung

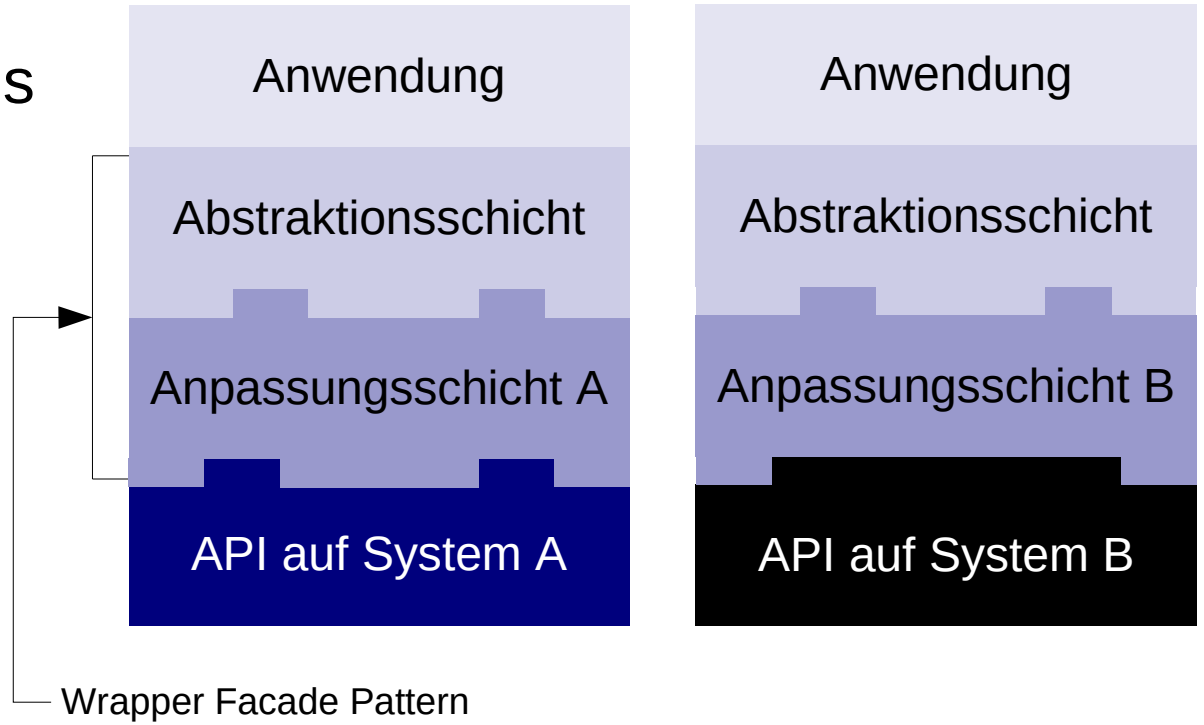
Abstraktionsschicht

- Bietet einheitliche Schnittstelle für APIs
- Ruft Anpassungs-schicht auf

Anpassungsschicht

Ruft System-API auf

- Austausch systemspezifischer Eigenschaften



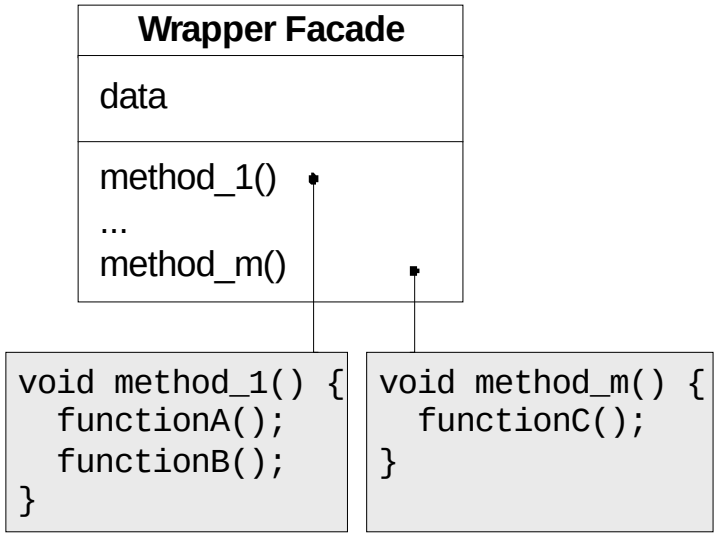
Implementierung der Anpassungsschicht

Fragen zu beantworten:

- Wie implementieren?
- Wie einbinden & austauschen?

Implementierungsverfahren:

- Präprozessorverfahren **`(#ifdef)`**
- Abstract Factory Verfahren
- Template-Verfahren



Relevanz der Wrapper Facade?

- Portabilität ist immer noch ein Thema
  - zwischen Betriebsystemen
  - inkompatible APIs durch Versionssprung
- systemnahe APIs immer noch in C geschrieben
- systemnahe APIs sind umständlich zu benutzten
  - niedriger Abstraktionsgrad
  - fehlende Datenkapselung
- Bindeglied zw. prozeduraler API und OO-Anwendung

Vorteile der Wrapper Facade

- Kompakterer Code & verbesserte Robustheit
  - durch höheren Abstraktionsgrad
  - durch objektorientierte Schnittstelle & Sprache
- Verbesserte Modularität
  - durch Gruppierung von API-Funktionen in Klassen
  - durch Organisation der Klassen
- Verbesserte Portabilität
  - durch Organisation & Implementierungsverfahren

Herausforderungen

- Erstellung guter APIs
  - ist schwierig
  - erfordert iteratives Vorgehen
- Aufwand für Implementierung
  - wird durch iteratives Vorgehen erhöht

Toolgestützte Wrapper Facade Entwicklung

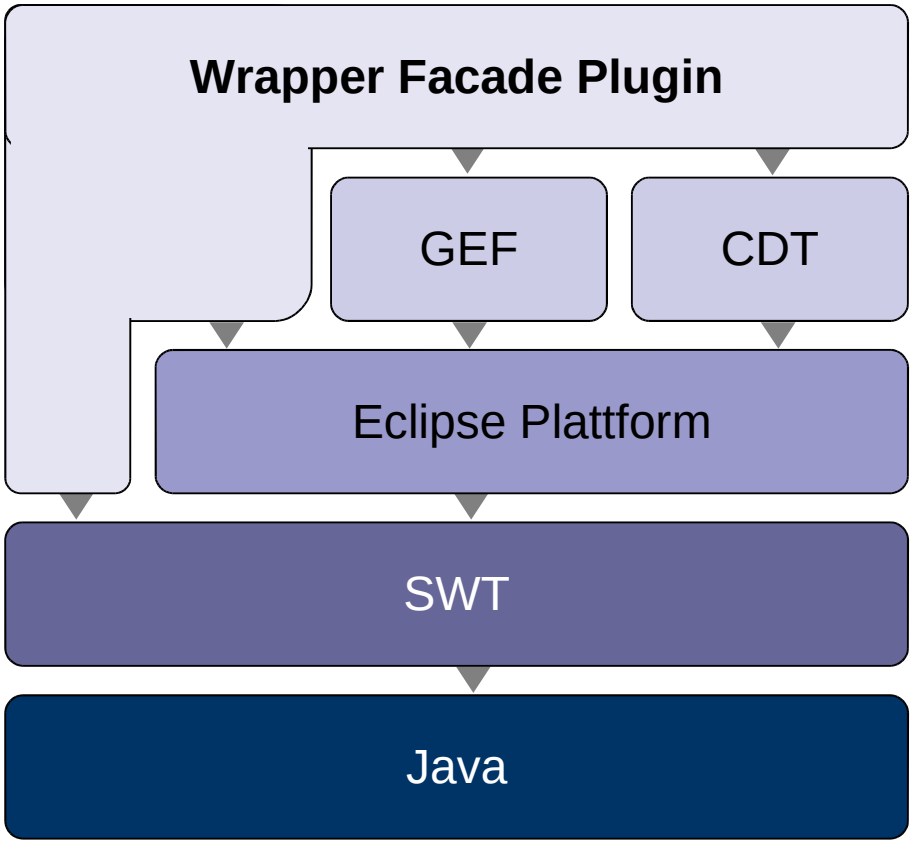
Wrapper Facade Plugin bietet:

- graphische Modellierung von Wrapper Facades
- C++ Codegenerierung
- Integration in Eclipse-IDE

Besteht aus:

1. Wrapper Facade Modell
2. graphischem Editor
3. Codegenerator
4. Import-Tool

Verwendete Technologien

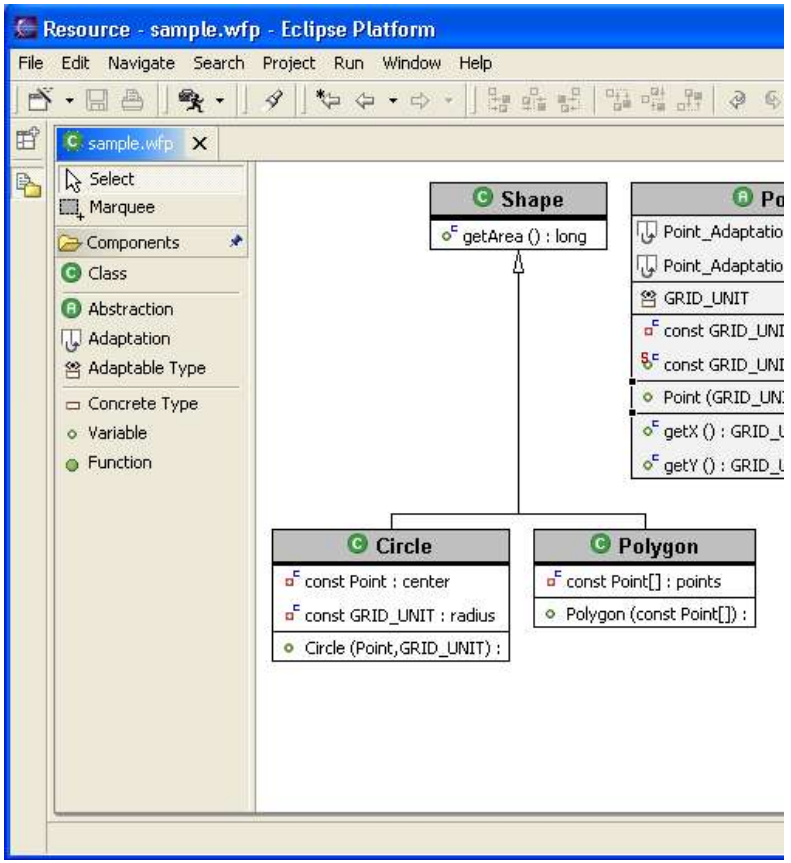


GEF: Graphical Editing Framework
CDT: C/C++ Development Tools
SWT: Standart Widget Toolkit

Features

Graphischer Editor

- zur Modellierung von Wrapper Facades
  - Drag & Drop von Modellelementen
  - autom. Layouterstellung
  - Zoom In / Out
  - Undo / Redo
- sprachspezifisch (C++)
- erkennt Fehler während der Eingabe
- speichert Modelldaten in XML



Features

Codegenerator

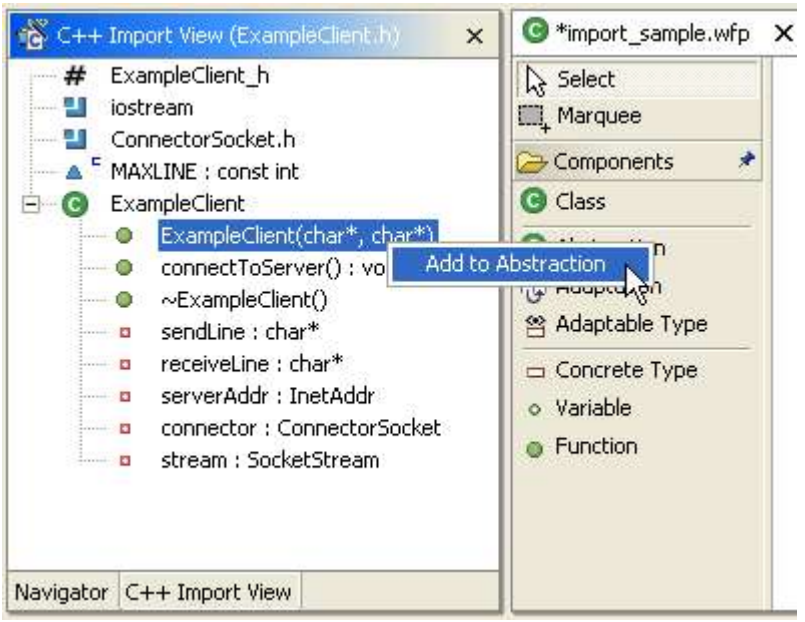
- austauschbar
- wendet das Template-Verfahren an
- erzeugt C++ Klassendefinitionen
- fügt Implementierung hinzu (wenn verfügbar)
- markiert anzupassende Stellen

```
template<typename api_t>
class Simple_Wrapper_Facade {
public:
 typedef typename api_t::X X;
 typedef typename api_t::Y Y;
 static const X xconst;

 inline static Y circle_area(X radius)
 { return api_t::circle_area(radius); }

protected:
private:
};
```

Features

Import-Tool

- stellt beliebige C++ und Header-Dateien strukturiert dar
- importiert Methoden- und Variablendeklarationen in das Modell
- erzeugt fehlende Datentypen während des Imports



Vorteile des Wrapper Facade Plugins

Graphischer Editor:

- modellbasierter Entwurf
- interaktive & iterative Entwicklung
- verbessert Kommunikation & Verständnis
- hilft Fehler zu vermeiden

Codegenerator:

- automatische Codeerzeugung
- austauschbar

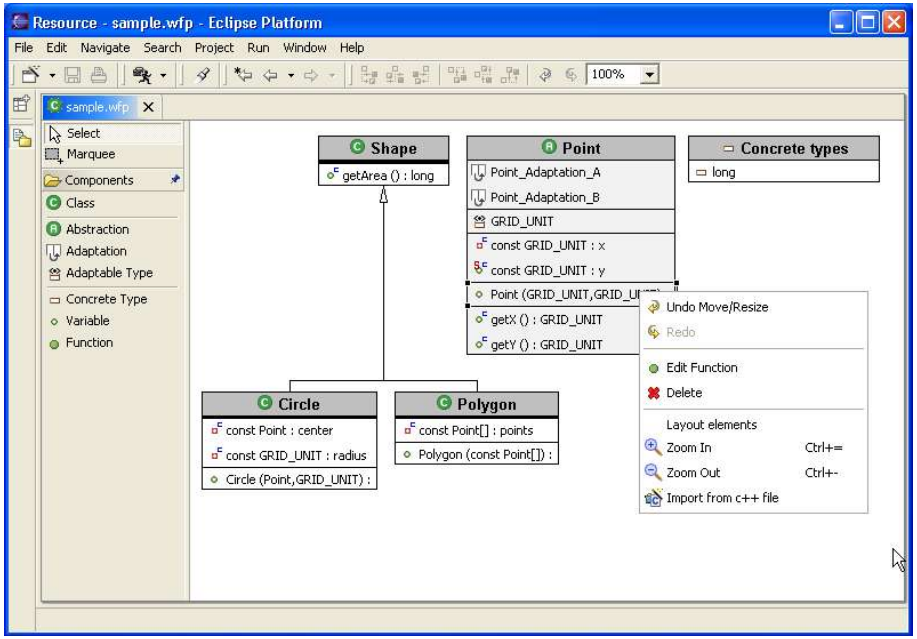
⇒ mehrere Implementierungen aus einem Modell

Import-Tool

- beschleunigt Modellerstellung

Demo-Session

Vorführung des Wrapper Facade Plugins
in der Eclipse-IDE



Zusammenfassung der geleisteten Arbeit

Konzipiert & Implementiert wurden:

ein OO Modell	Beschreibung von WF
ein graphischer Editor	interaktive Modellierung
ein Codegenerator	wandelt Modell in C++ Code
ein Import-Tool	unterstützt Modellerstellung
Use-Cases	Socket Wrapper Facade

Ende

Vielen Dank für Ihre Aufmerksamkeit!

Eure Fragen und Diskussion sind willkommen!

Kontakt: elias@volanakis.de