Anpassung

- Adapter
- Facade
- Bridge

Adapter

Quelle:www.bilder.preistaktik.de

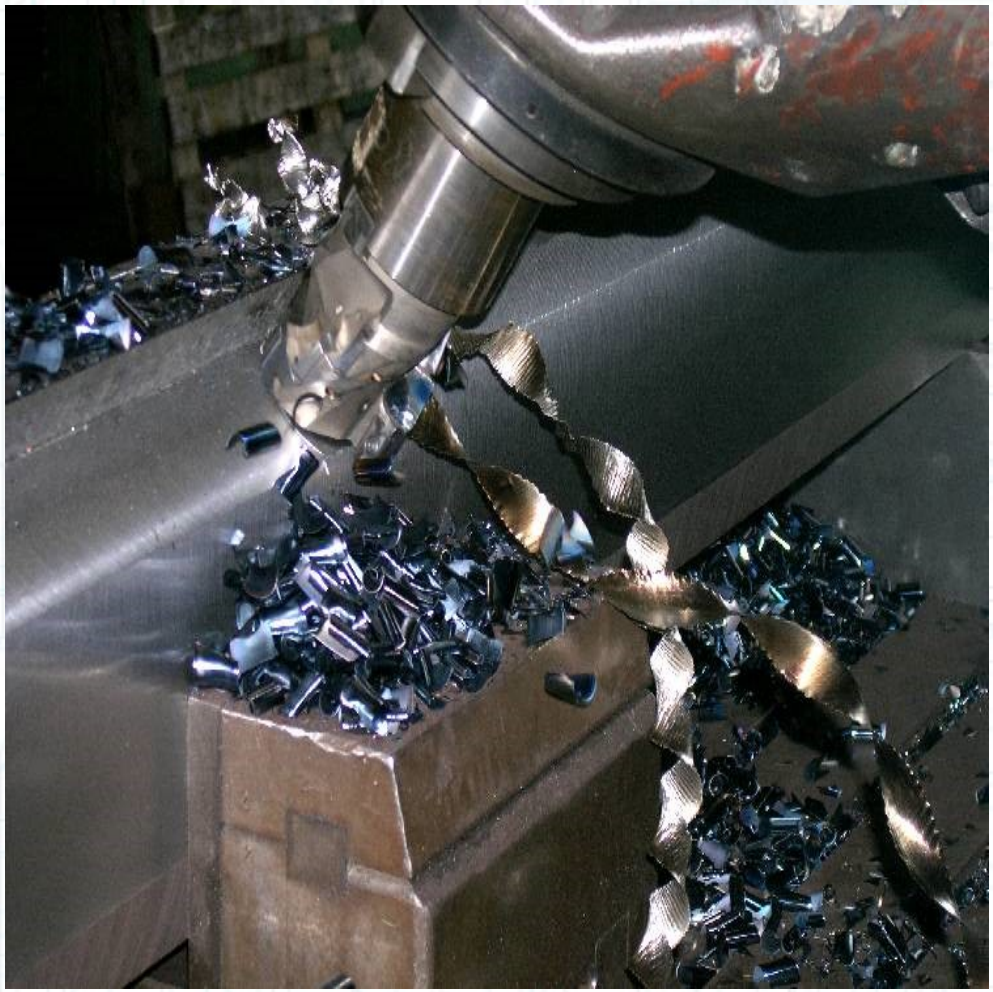
Adapter - Problemstellung

- 2 Klassen können aufgrund inkompatibler Schnittstellen nicht interagieren.
- **Änderung der Schnittstellen nicht möglich!**

Adapter - Intuitive Lösung

- Verbindungsstück einfügen
- oder solange mit Zange und Hammer hantieren bis es passt.

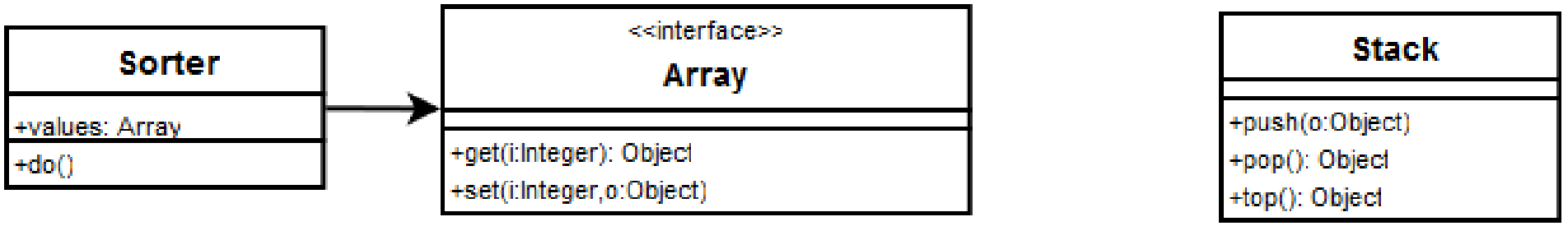
Adapter vs Aufbohren



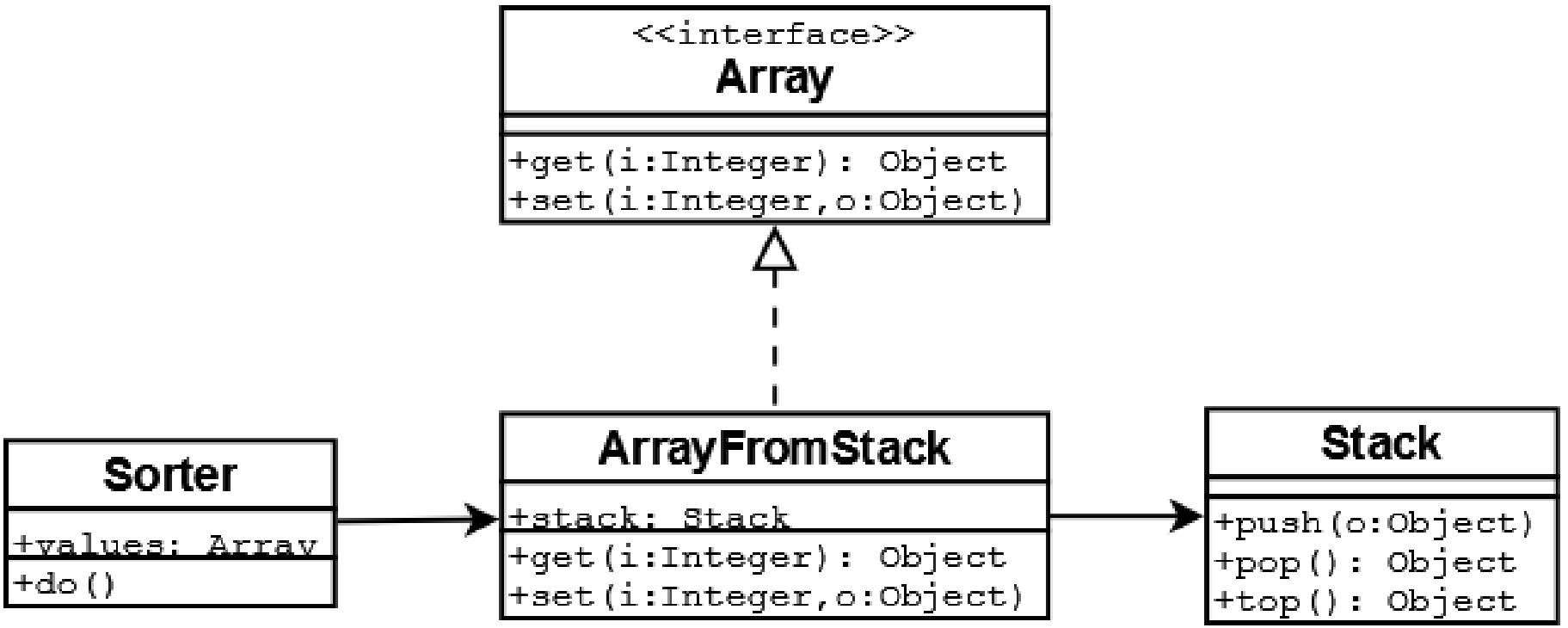
Quelle: www.mts-shop.at

Quelle: www.groner-gmbh.de

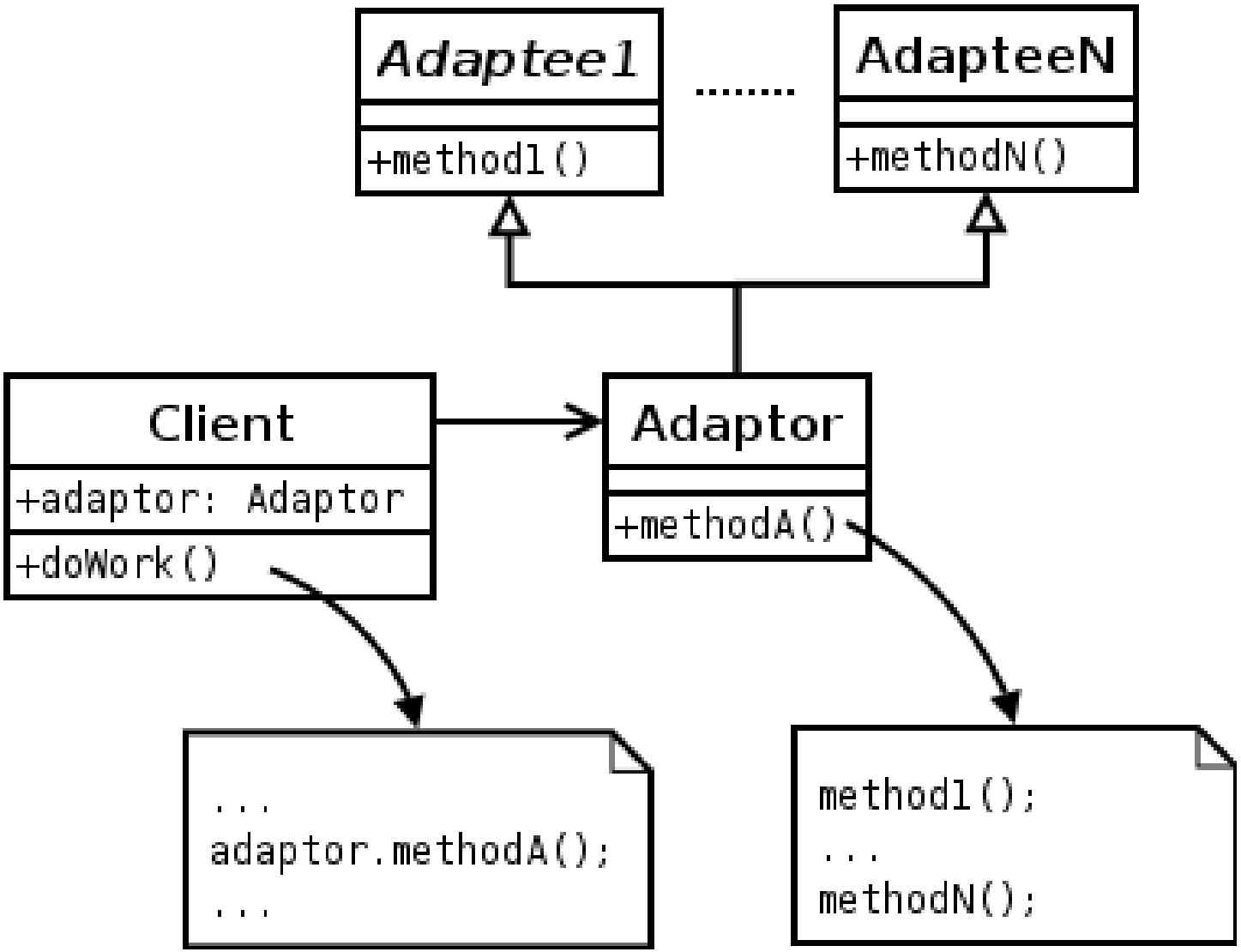
Adapter - Beispiel



Adapter - Beispiel

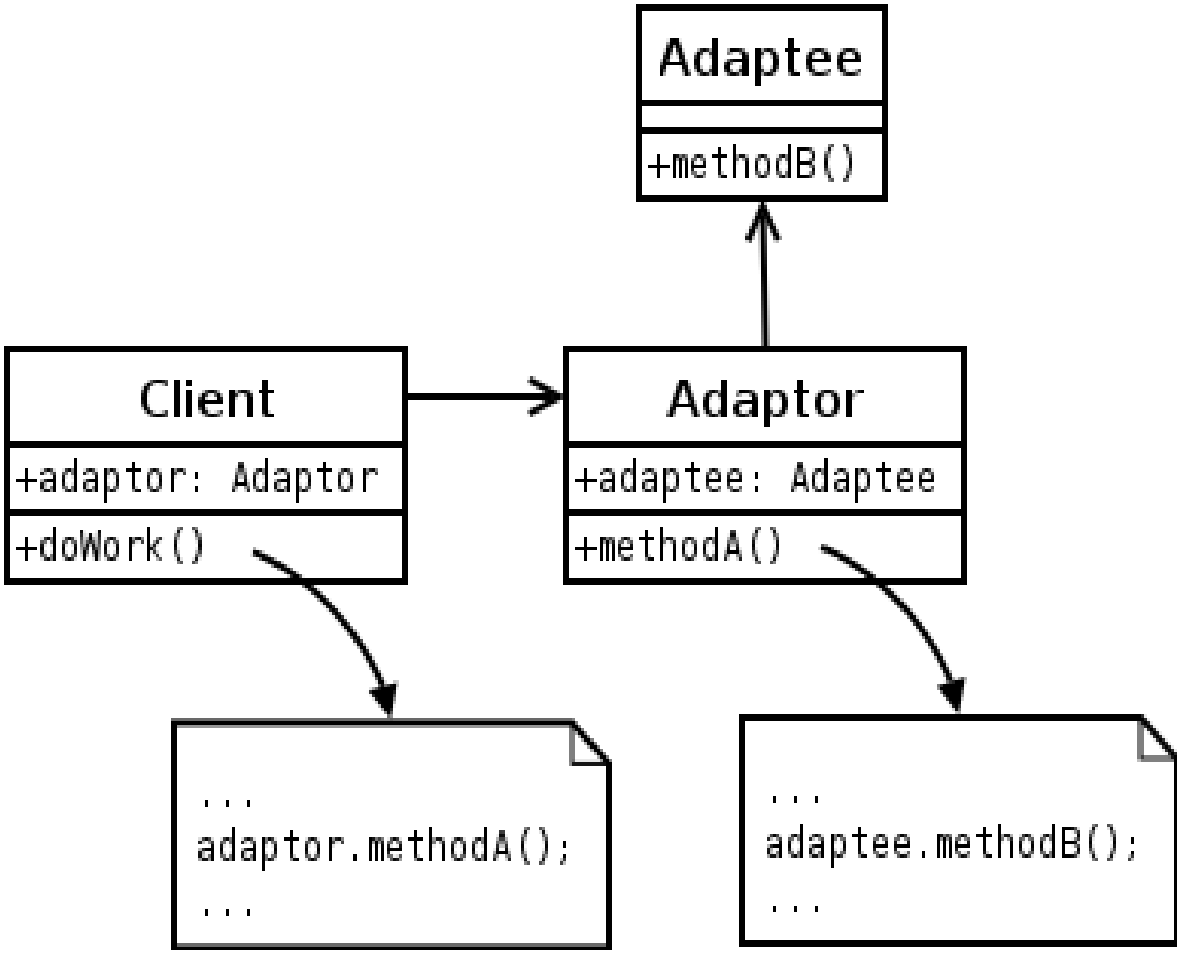


Klassenadapter



Quelle: Wikipedia.org
GilHamilton

Objektadapter



Quelle: Wikipedia.org
GilHamilton

Adapter - Vergleich

Klassenadapter

- Mehrfachvererbung
- Von Vorteil wenn die Ausgangs-klasse bereits einen Teil der Zielschnittstelle implementiert

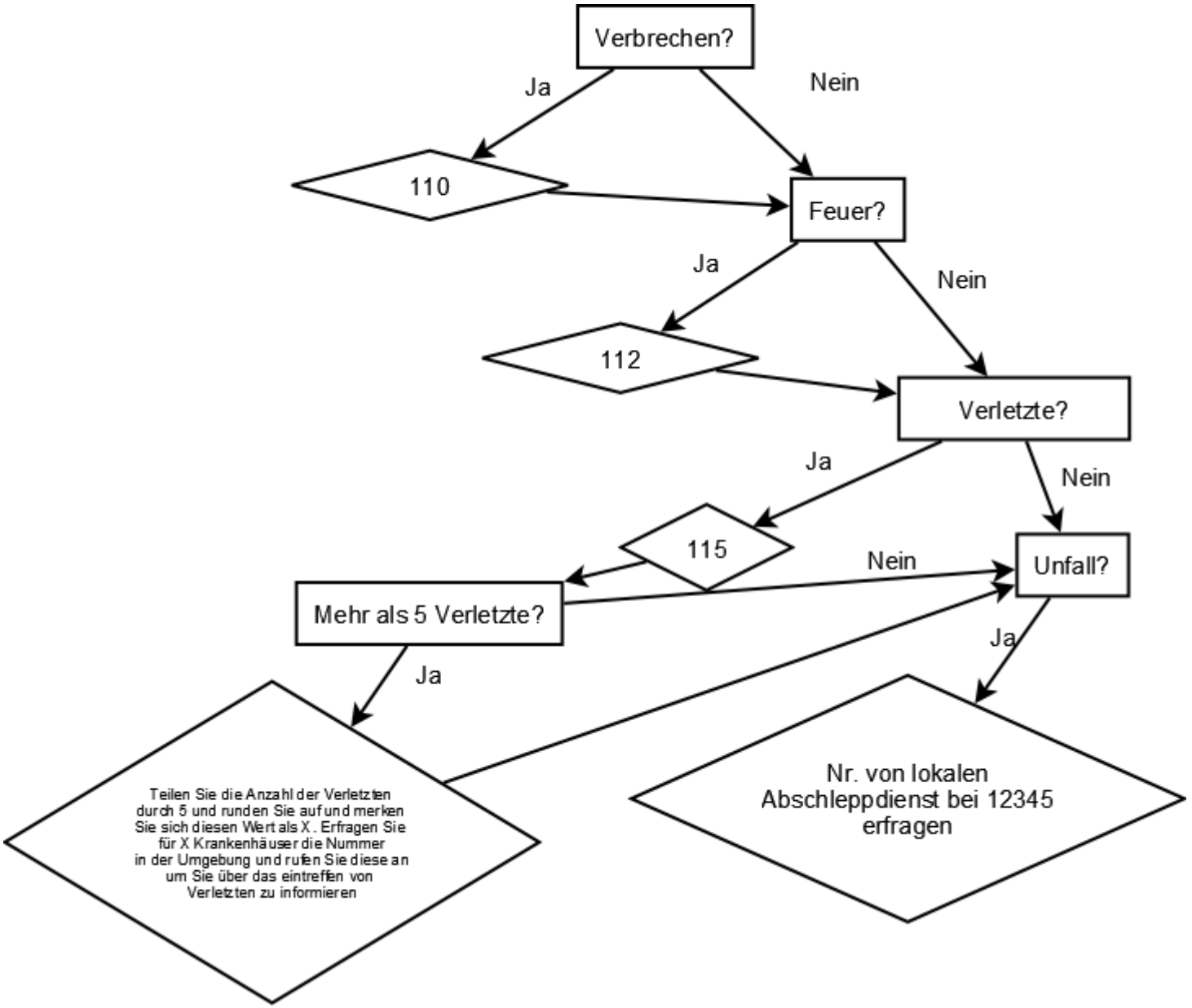
Objektadapter

- Referenzierung
- Transformation wird durch Aufrufe von Methoden auf den referenzierten Objekten realisiert

Adapter - Vorteile

- Ermöglicht die Wiederverwendung bisher inkompatibler Klassen.
- Ausgangsklassen werden dabei nicht verändert.
- Adapter können auch deutlich nach Entwicklungsabschluss der beteiligten Klassen erstellt werden.

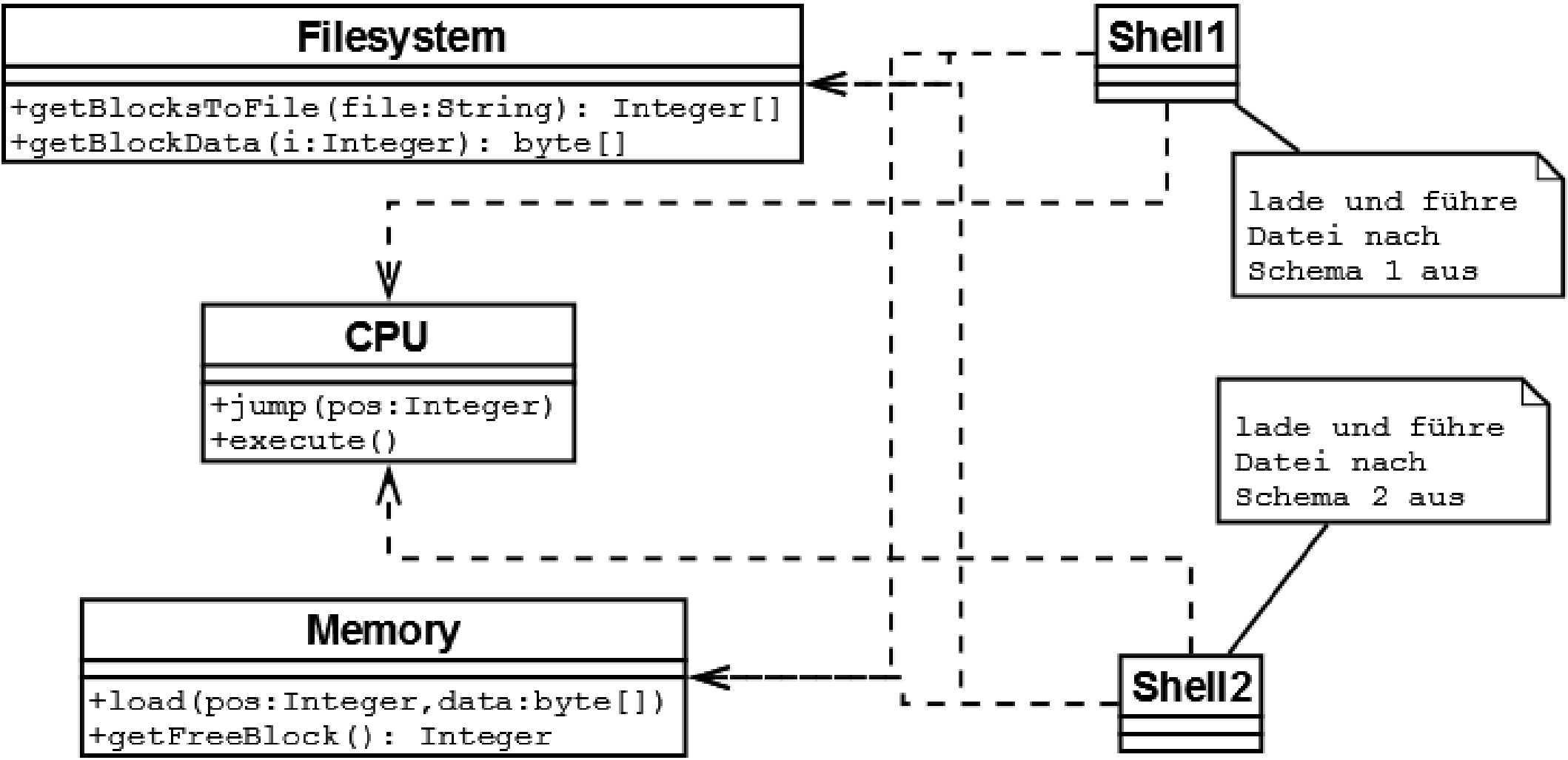
Facade - Notrufsystem



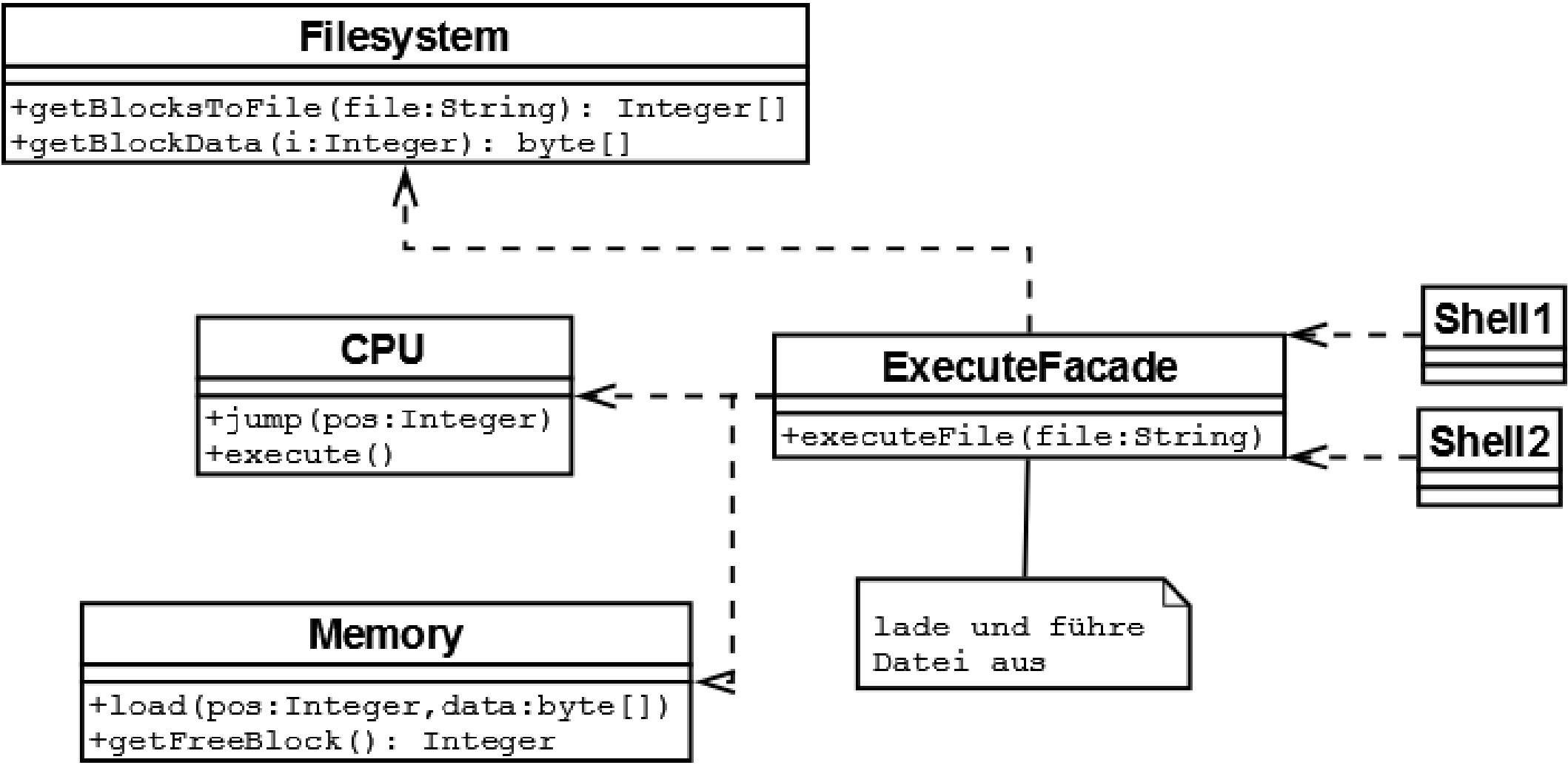
Facade

- Problem: viele Funktionen mit nur schwer überschaubarem Zugriff
- Lösung: Zwischenschicht um bestimmte (immer wiederkehrende) Abläufe zu vereinfachen
- Siehe Notrufleitzentrale - 110

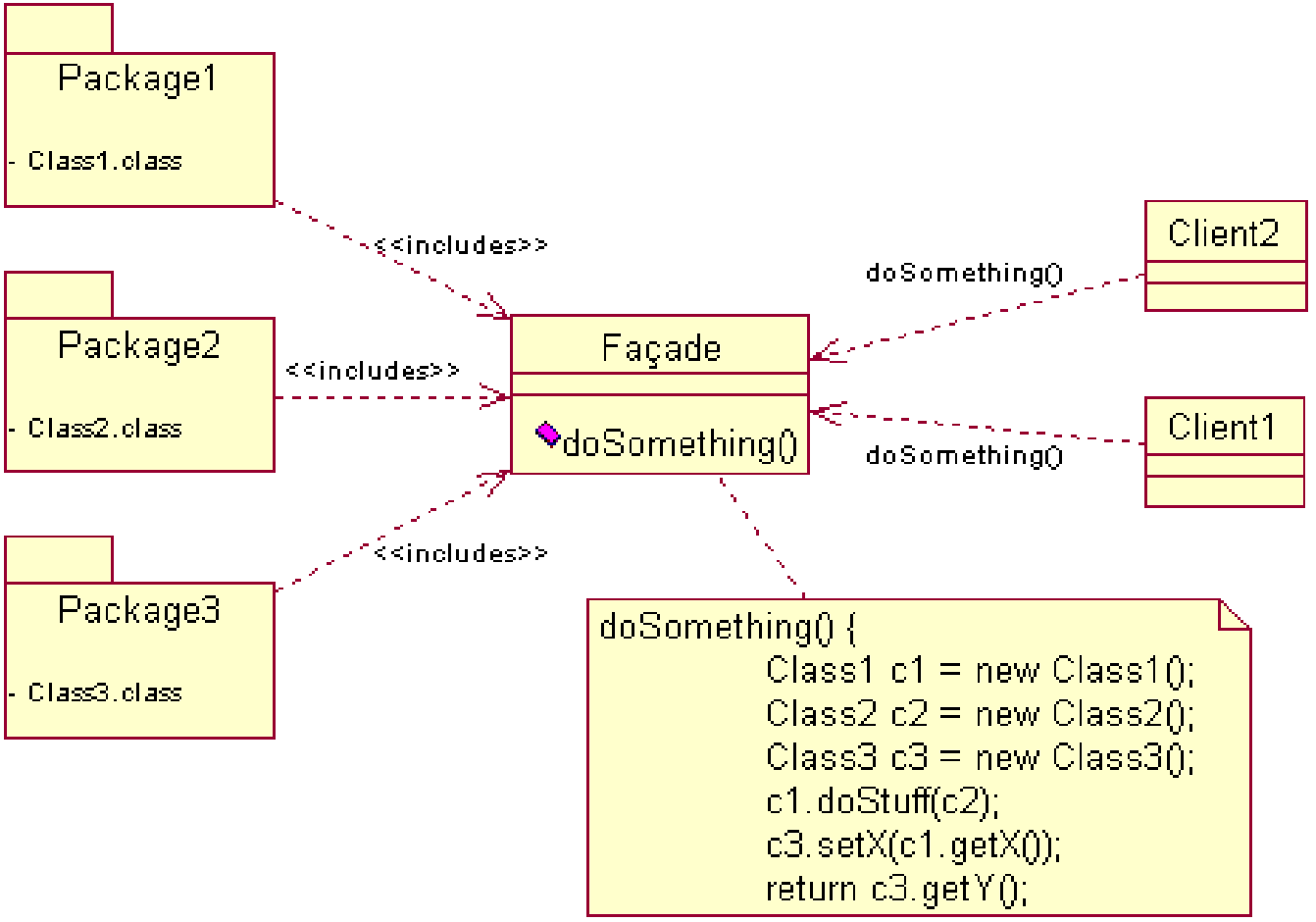
Facade - Beispiel



Facade - Beispiel



Facade



wikipedia.org - Alcides Carlos de Moraes Neto

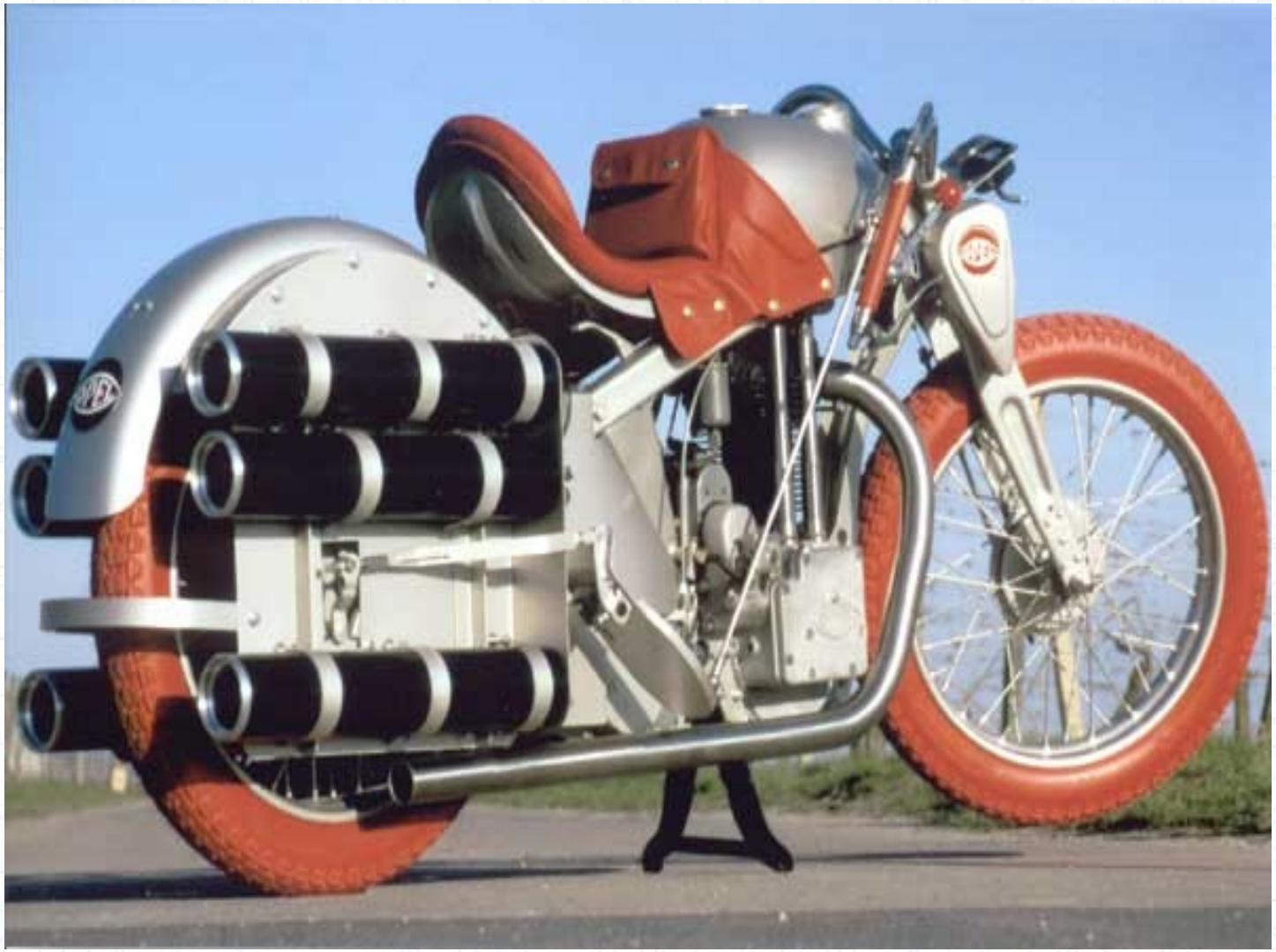
Facade - Vorteile

- Vereinfachung der Struktur
- Rückgriff auf getestete und sichere Methoden, um gängige Aufgaben zu lösen
- Möglichkeiten werden nicht eingeschränkt

Vergleich Adapter/Facade

- Beide fügen eine Zwischenschicht ein
- Adapter verbindet 2 unabhängige Klassen/ Interfaces, nachdem die beteiligten Klassen realisiert wurden
- Facade stellt eine allgemeine Schnittstelle für eine Vielzahl von Klienten und inneren Funktionen zur Verfügung und muss zumindest vor Realisierung der Klienten geplant sein

Bridge



Quelle: http://www.classic-motorrad.de

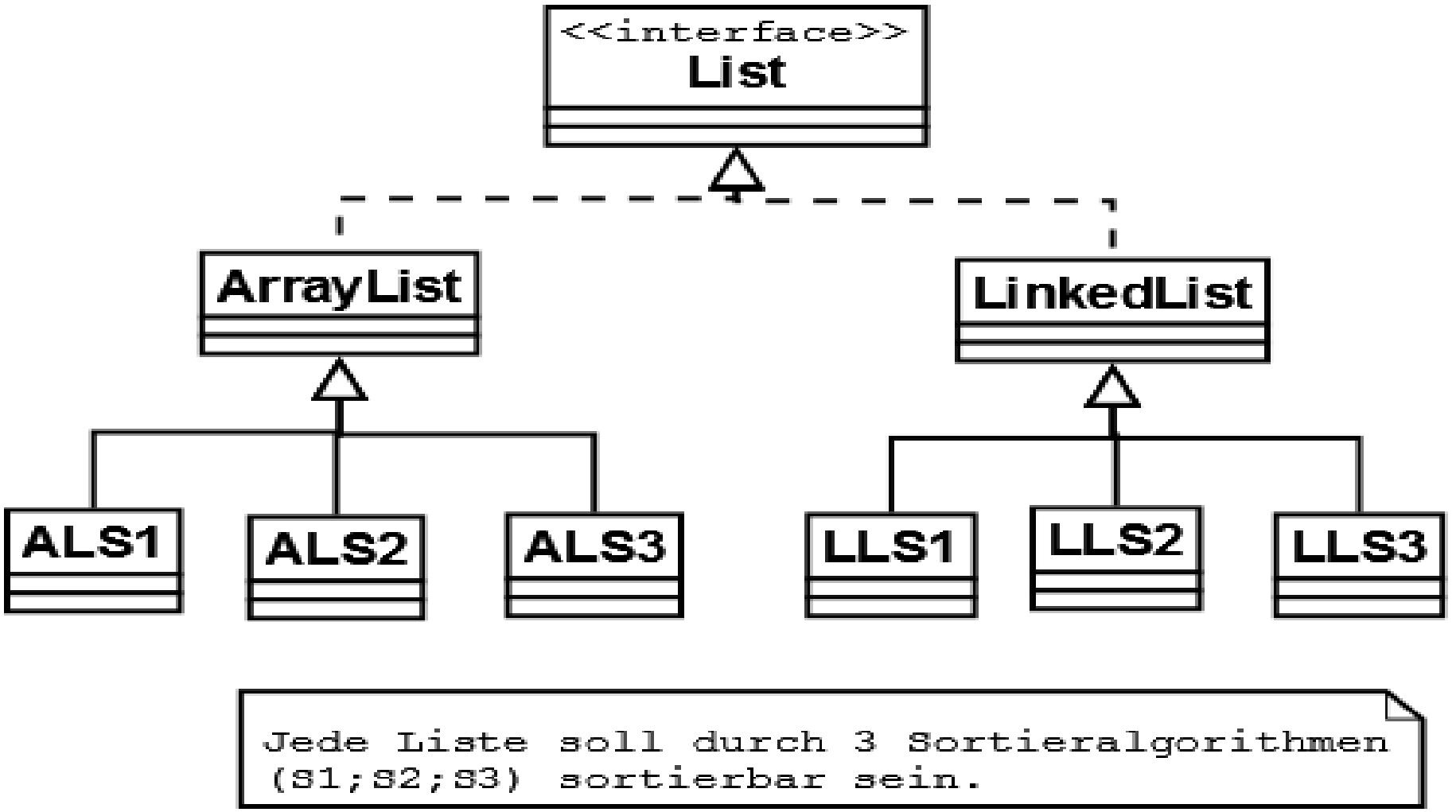
Bridge

- Normalfall: Implementierung erbt von Abstraktion
- Implementierung kann also nicht alleine existieren
- Trennung von Implementierung und Abstraktion kann aber gewollt sein, um die Anzahl der benötigten Klassen zu reduzieren

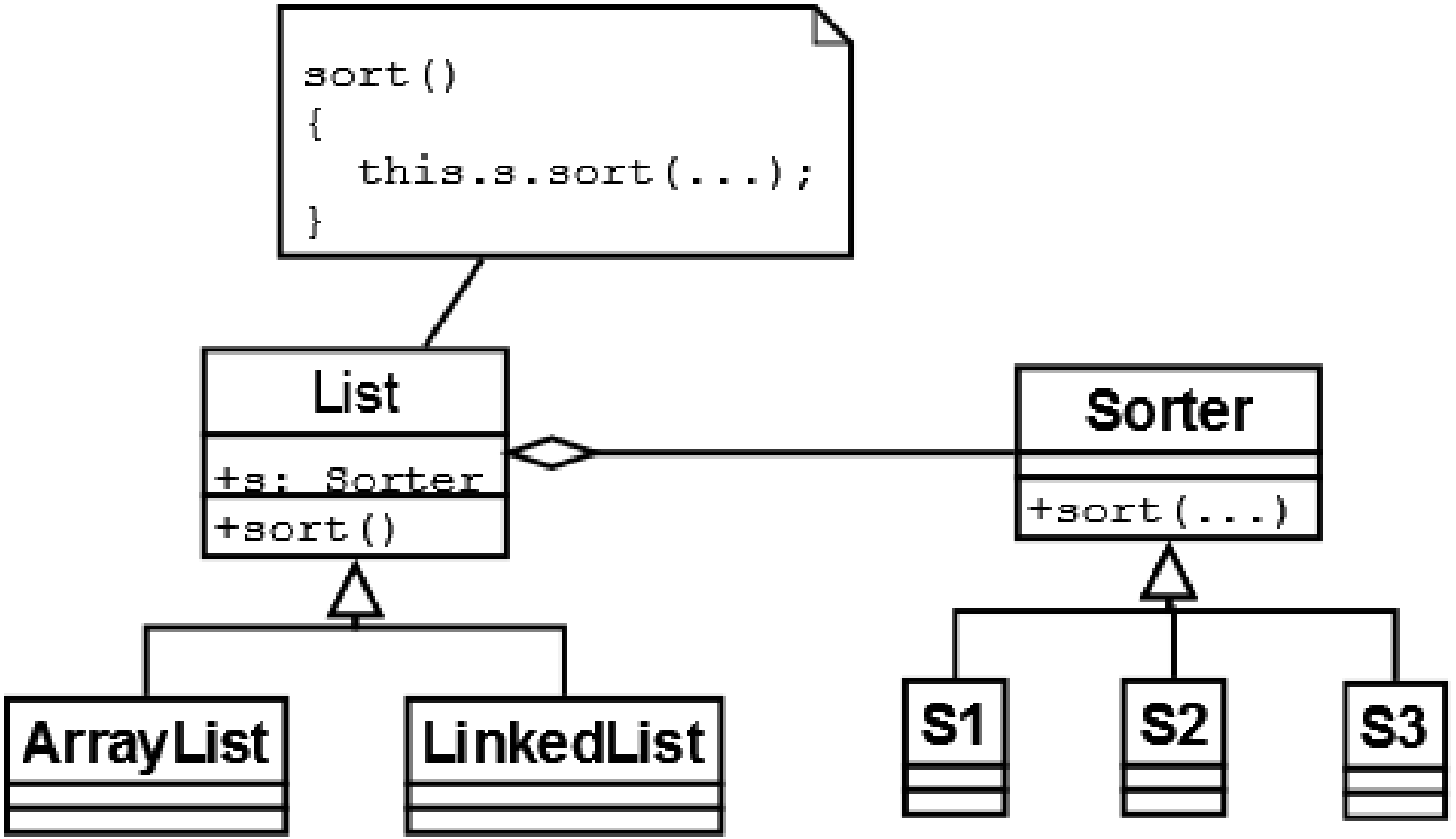
Bridge

- Der Teil des Problems, der unabhängig sein soll, wird in eine eigenständige Implementierung ausgelagert
- In der Abstraktion wird eine Implementierung referenziert

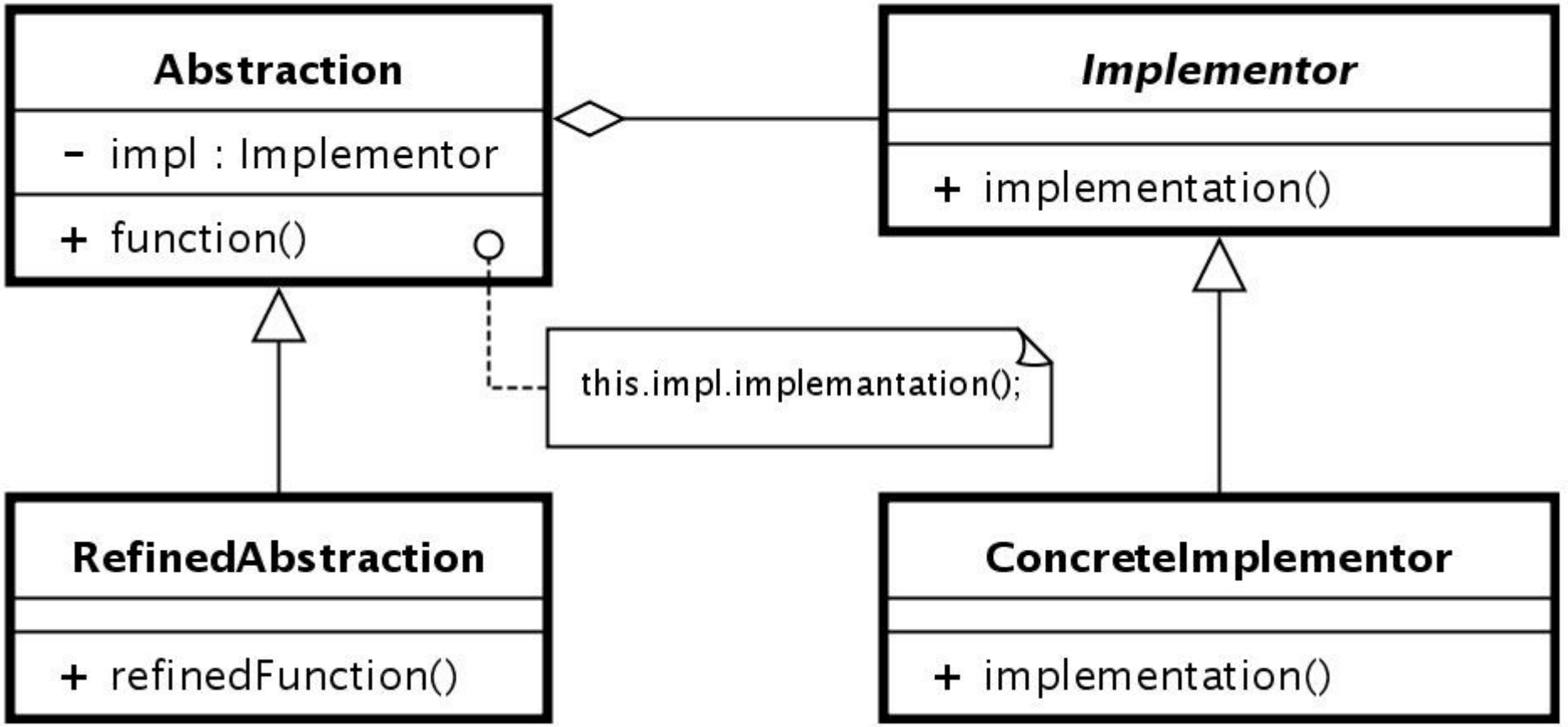
Bridge - Beispiel



Bridge - Beispiel



Bridge



Quelle: Wikipedia.org - Trashtoy

Bridge - Vorteile

- Weniger Klassen => übersichtlicheres Design
- Bessere Erweiterbarkeit, auch unabhängig von dem jeweils anderem Abstraktionsstrang
- Austausch der Implementierung zur Laufzeit möglich

Wann, Was?

- Bridge – muss bereits während der Designphase geplant werden
- Adapter – eigenständiges Projekt und unabhängig von der Entwicklung des Restsystems
- Facade – 3 Teilprojekte
    1. Innere Klassen oder Facade
    2. Facade oder Innere Klassen
    3. Klienten