Das JWAM-Framework und Komponenten

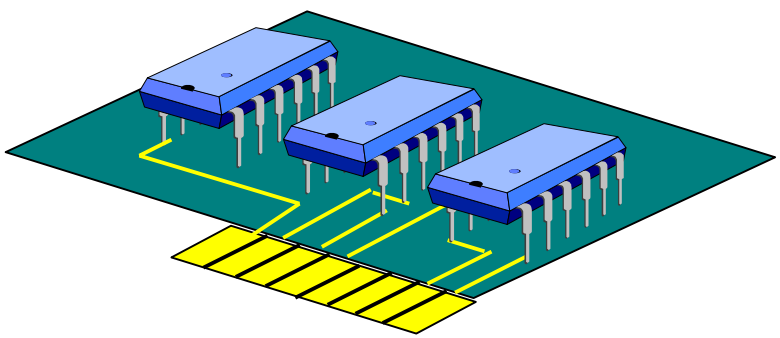
Eine konzeptionelle Bestandsaufnahme

- Kontext
- Der Begriff Komponente
- Aktuelle Bedeutung
- Komponenten und Frameworks
- Komponenten in JWAM
  - Ausstattungskomponenten
  - Einschubkomponenten
- Ausblick

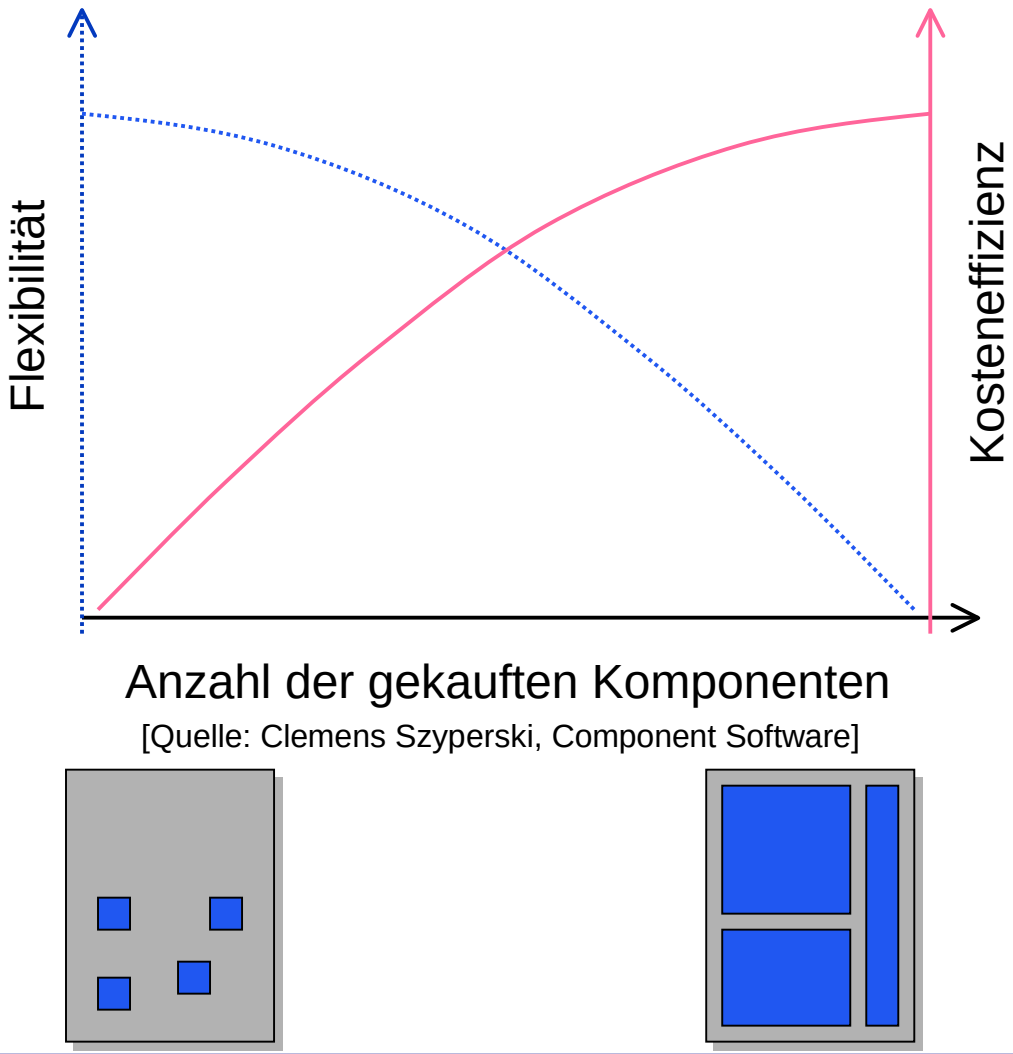
Kontext — Wunschvorstellungen 1

Die Hoffnung:

- Komponentensoftware kann analog zu Hardware aus individuell verfügbaren Komponenten (Software-ICs) komponiert werden.
- Komponenten können additiv hinzugefügt werden.
- Je nach Bedarf können einfache Komponenten durch komplexe ersetzt werden.
- Weiterentwicklung und Verbesserungen finden auf Ebene von Komponenten statt.



Kontext — Wunschvorstellungen 2

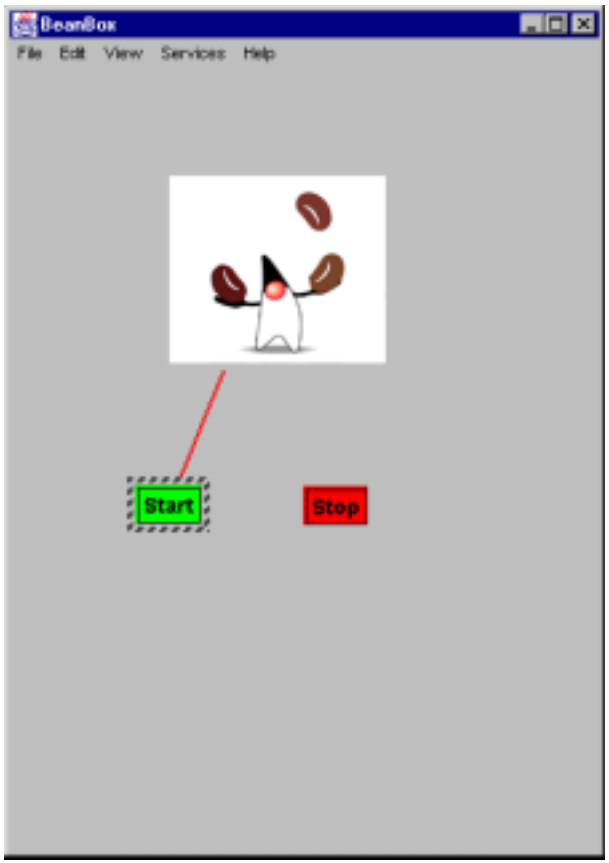


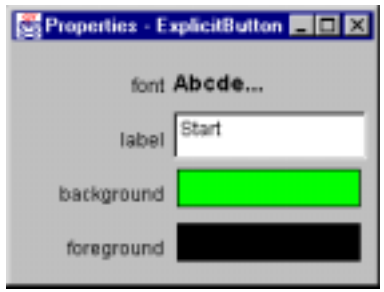
[Quelle: Clemens Szyperski, Component Software]

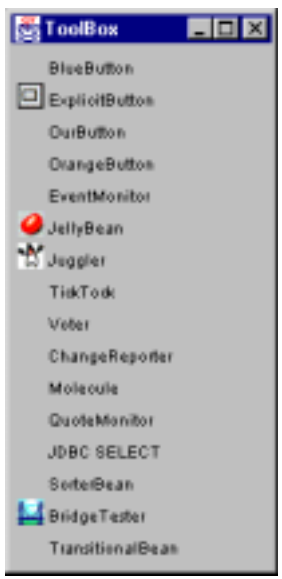
- Das Maß, in dem gekaufte Komponenten eingesetzt werden können, ist frei wählbar
- gekaufte Komponenten lassen sich problemlos mit eigenen Programmstrukturen kombinieren
- Eigenentwicklungen können problemlos zu Komponenten gemacht und auf dem Komponentenmarkt angeboten werden

Kontext — Wunschvorstellungen 3

- Geschulte Anwender sollen Standard-komponenten auswählen und in speziellen Editoren graphisch zu fertigen Systemen zusammen







Der Begriff Komponente — Grundeigenschaften

> "A component is a unit of composition with contractually specified interfaces and explicit context dependencies only. Components can be deployed independently and are subject to composition by third parties."
>
> *Workshop on Component-Oriented Programming, ECOOP 1996*

- macht explizit:
  - für Wiederverwendung
  - explizite Schnittstelle(n) darüber was sie anbieten und was sie benötigen
- schließt damit aus:
  - alltagssprachliche Komponenten im Sinne von „Bestandteil"
  - Dokumentation

Der Begriff Komponente — Spezialisierungen

- Entwurfskomponenten
  - ➔ Liegen im Quelltext vor

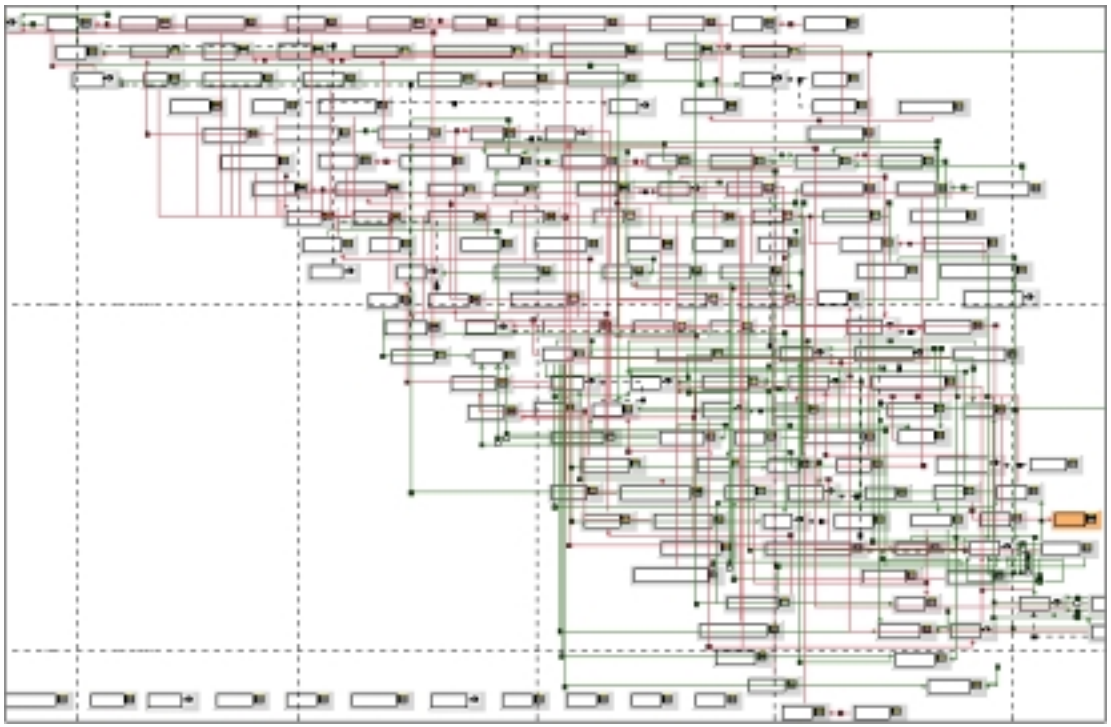
> Komponenten sind in Binärform vorliegende Lösungen softwaretechnischer Probleme, die so zusammengefügt werden können, dass sie ein lauffähiges Softwaresystem bilden. Sie können unabhängig voneinander hergestellt, erworben und eingesetzt werden. Dazu besitzen Komponenten eine feste Schnittstelle und enthalten potentiell auch eigene Ressourcen, auf die sie bei der Ausführung zurückgreifen.
>
> *Clemens Szyperski, 1998*

- Implementationskomponenten
  - ➔ liegen in Binärform vor
  - ➔ werden statisch ins Kompilat eingebunden
- Laufzeitkomponenten
  - ➔ ebenfalls binär
  - ➔ werden zur Laufzeit eingebunden

Aktuelle Bedeutung — Kleinkomponenten

Implementatiations-komponenten für GUI-Bausteine erfolgreich

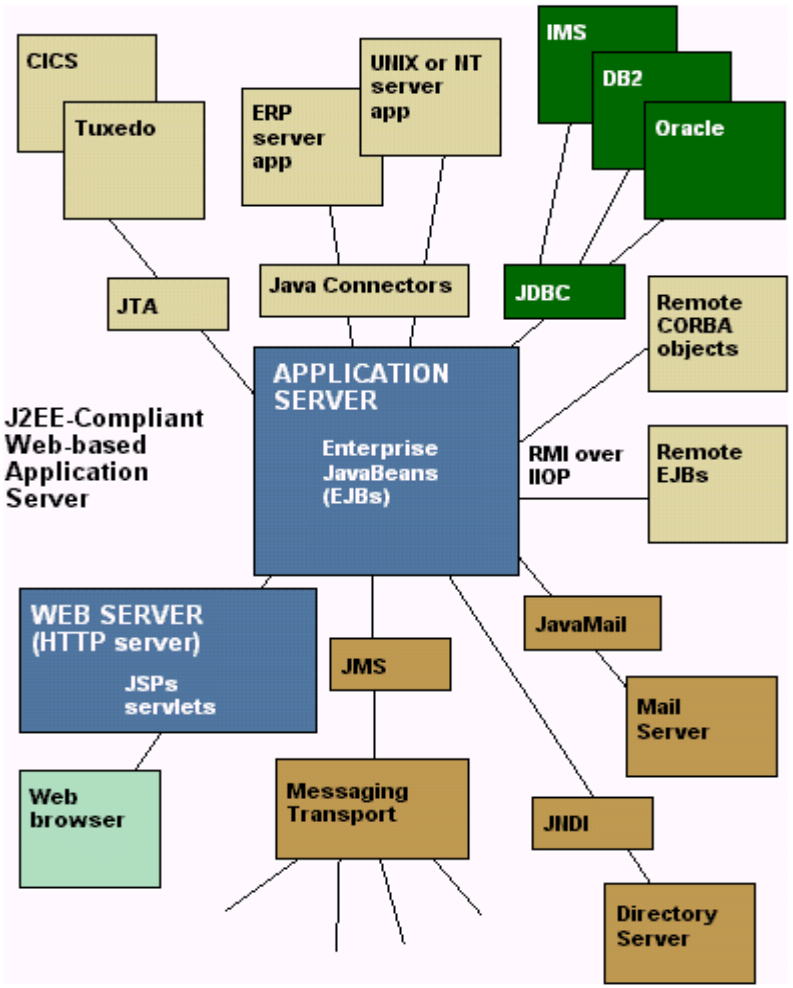
- ➔ werden in graphischer Umgebung zusammengestellt
- ➔ rein technische Verdrahtung
- ➔ wird für große Systeme unübersichtlich (statt dessen besser Glue Code)
- ➔ JavaBeans
- ➔ VBX, OCX, ActiveX



Aktuelle Bedeutung — Einsatzmöglichkeiten

Laufzeitkomponenten für Auslieferung von ERP-Systemen erfolgreich

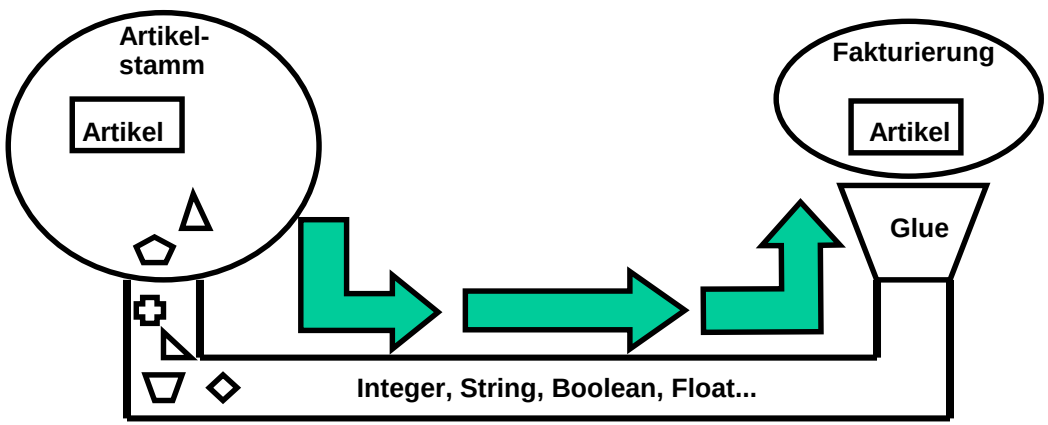
- ➔ Ziel ist nicht Markt, sondern plattformun-abhängige Auslieferung und / oder Verteilung
- ➔ fachlich komplett bis auf GUI (JSP, Servlets)
- ➔ Enterprise JavaBeans
- ➔ SAP R/3, Datenbanken



Aktuelle Bedeutung — Probleme

„Mittelgroße“, fachliche Komponenten

- ➔ haben fachliche Schnittstellen, die sowohl von Groß- als auch Kleinkomponenten ausgespart bleiben
- ➔ auf einheitlichem Softwarebus geht fachliche Semantik verloren, wenn Komponenten separat entwickelt werden (viel Glue Code)
- ➔ Komplexe Abhängigkeitsgeflechte zwischen Komponenten
- ➔ zahlreiche unterschiedliche Konventionen (z.B. Sprache, Verantwortlichkeit für Objektverwaltung, Referenzsemantik, etc.)



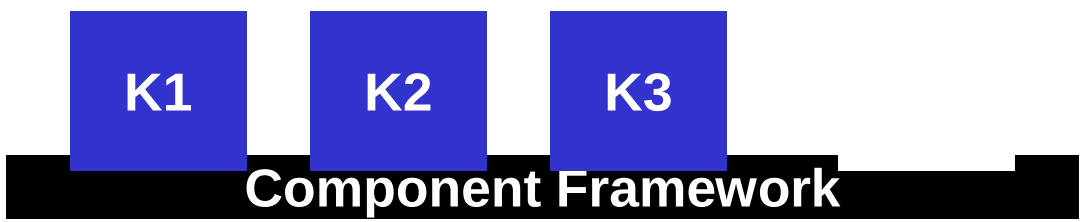
Aktuelle Bedeutung — Probleme

Erfolgreich sind sehr große und sehr kleine Komponenten

- ➔ kontextunabhängig
- ➔ rein technische „Verdrahtung“
- ➔ ERP-Systeme vs. Oberflächenelemente wie Knöpfe, Listen etc.

Problematisch sind mittelgroße Komponenten

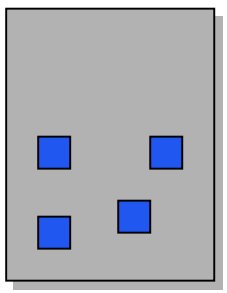
- ➔ erfordern fachliche, kontextabhängige Kommunikation
- ➔ setzt fachliche Einbettung voraus, die nicht ad hoc hergestellt werden kann, sondern von vorneherein geplant werden
- ➔ Mittel: *Component Frameworks*
- ➔ Szyperski fordert sie, stellt selbst aber nur sehr technische Beispiele vor (*OpenDoc* und *BlackBox*)



Frameworks und Komponenten

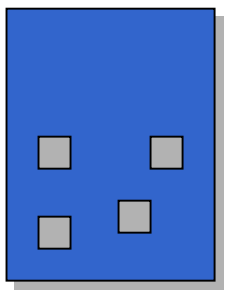
Komponenten

- ➔ Wiederverwendung von Teillösungen
- ➔ Zusätzlicher Glue Code zur Verbindung
- ➔ Probleme durch mangelnde Kommunikation



Frameworks

- ➔ Wiederverwendung von Architektur
- ➔ Framework *ist* der Glue Code, der dessen Teile zusammenhält
- ➔ *fatware*-Problematik



Gemeinsamkeiten

- ➔ Wiederverwendung über Vererbung (White Box) oder Komposition (Black Box)

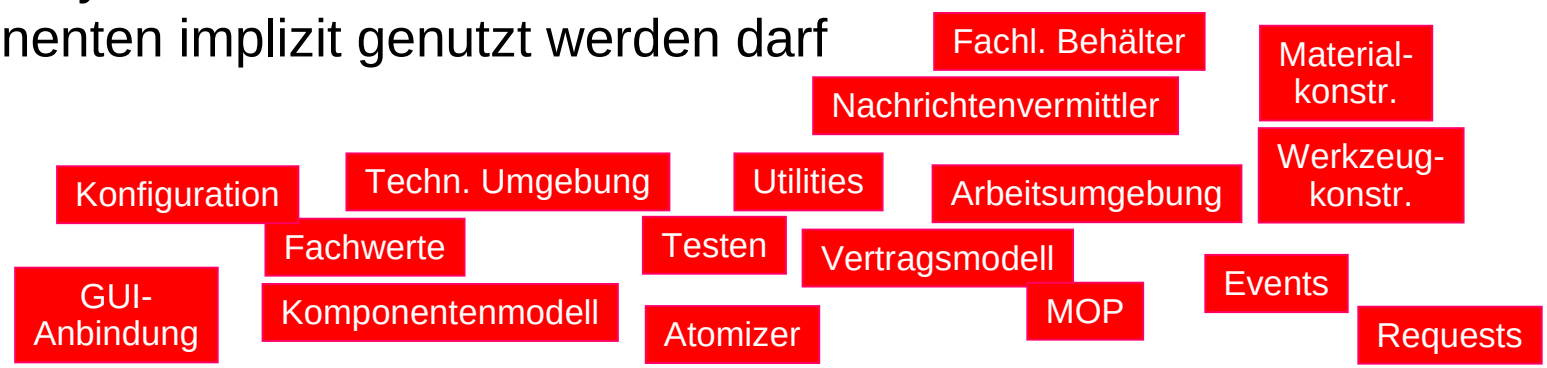
Kombi-Ansatz

- ➔ Framework als fachliche Grundlage für Komponenten
- ➔ Komponenten als Lösung gegen *fatware* bei Frameworks

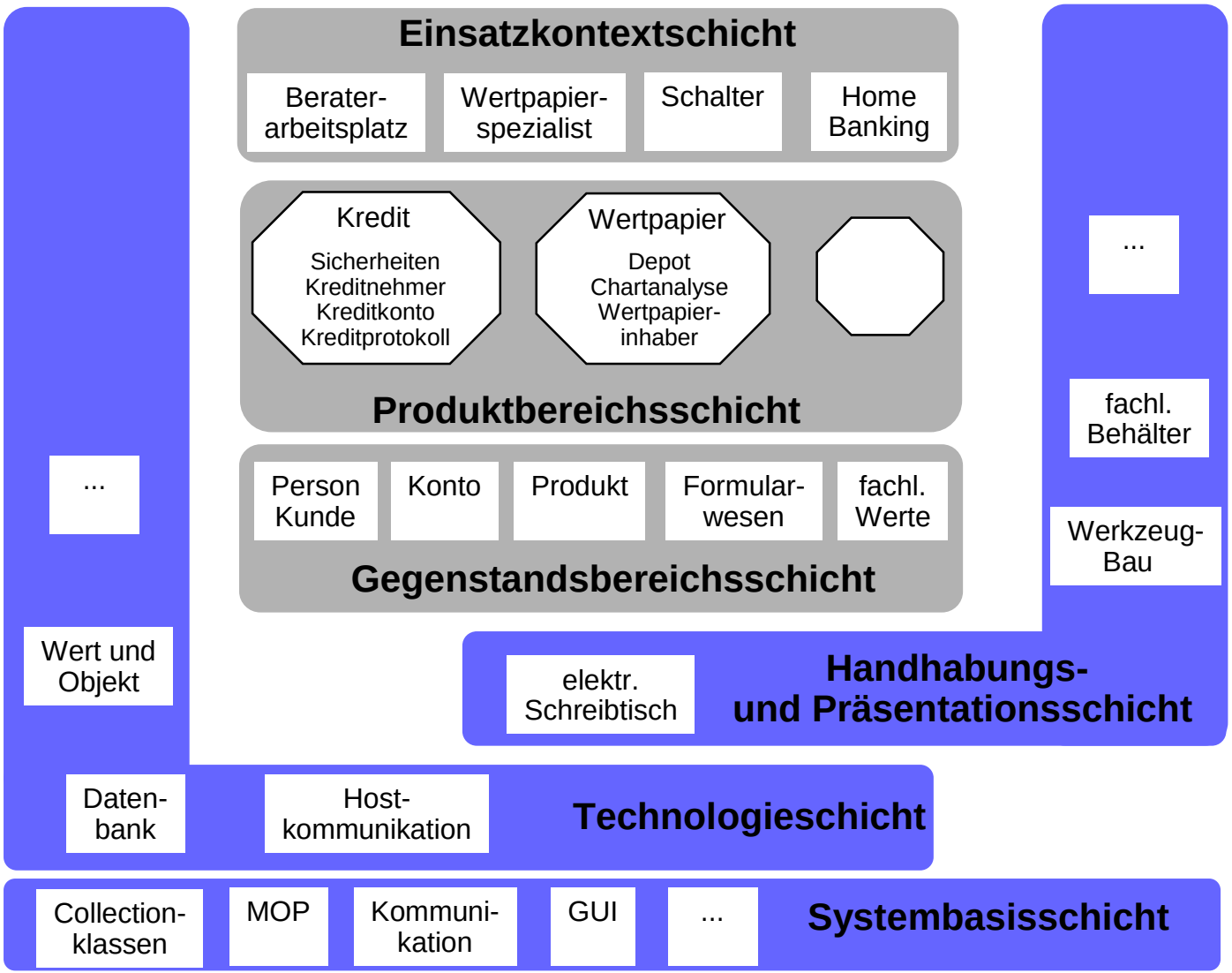
JWAM — Konzepte

- Methodische Basis ist der Werkzeug- und Material-Ansatz (WAM)
- In Java implementiert und in zahlreichen Projekten in unterschiedlichen Anwendungsgebieten eingesetzt
- Weiterentwicklungen und Erkenntnisse aus Projekten fließen in JWAM ein und bestimmen pragmatisch dessen Entwicklungsrichtung

- JWAM gibt als Framework generelle Schnittstelle für Komponenten vor, damit Erweiterbarkeit gewährleistet ist
- stellt die Systembasis dar, deren Funktionalität von allen Komponenten implizit genutzt werden darf



JWAM — Struktur



Komponenten in JWAM — Ausstattungskomponenten

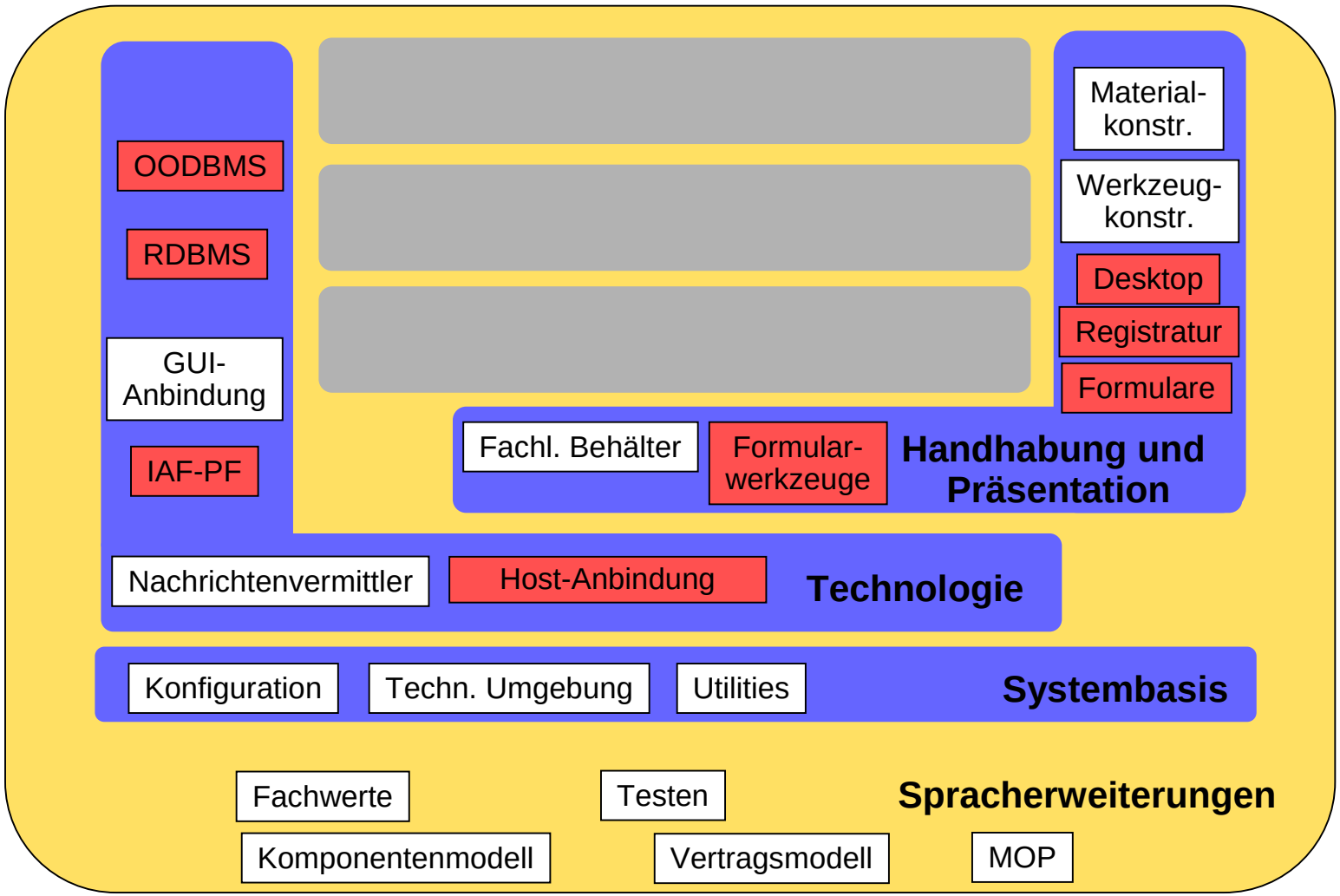
JWAM-Kern soll schlank gehalten werden

- ➔ nicht in jedem Projekt benötigte Funktionalität ist in ***Ausstattungskomponenten*** angesiedelt

Ausstattungskomponenten

- ➔ kapseln fachlich motivierte Funktionalität (Desktop, Formularwesen, etc.)
- ➔ sind Entwurfskomponenten
- ➔ werden als White Boxes von Entwicklern verwendet
- ➔ kommunizieren nur über den Kern miteinander
- ➔ machen explizit, von welchen anderen Ausstattungskomponenten sie abhängen

Komponenten in JWAM — Ausstattungskomponenten



Komponenten in JWAM — Einschubkomponenten

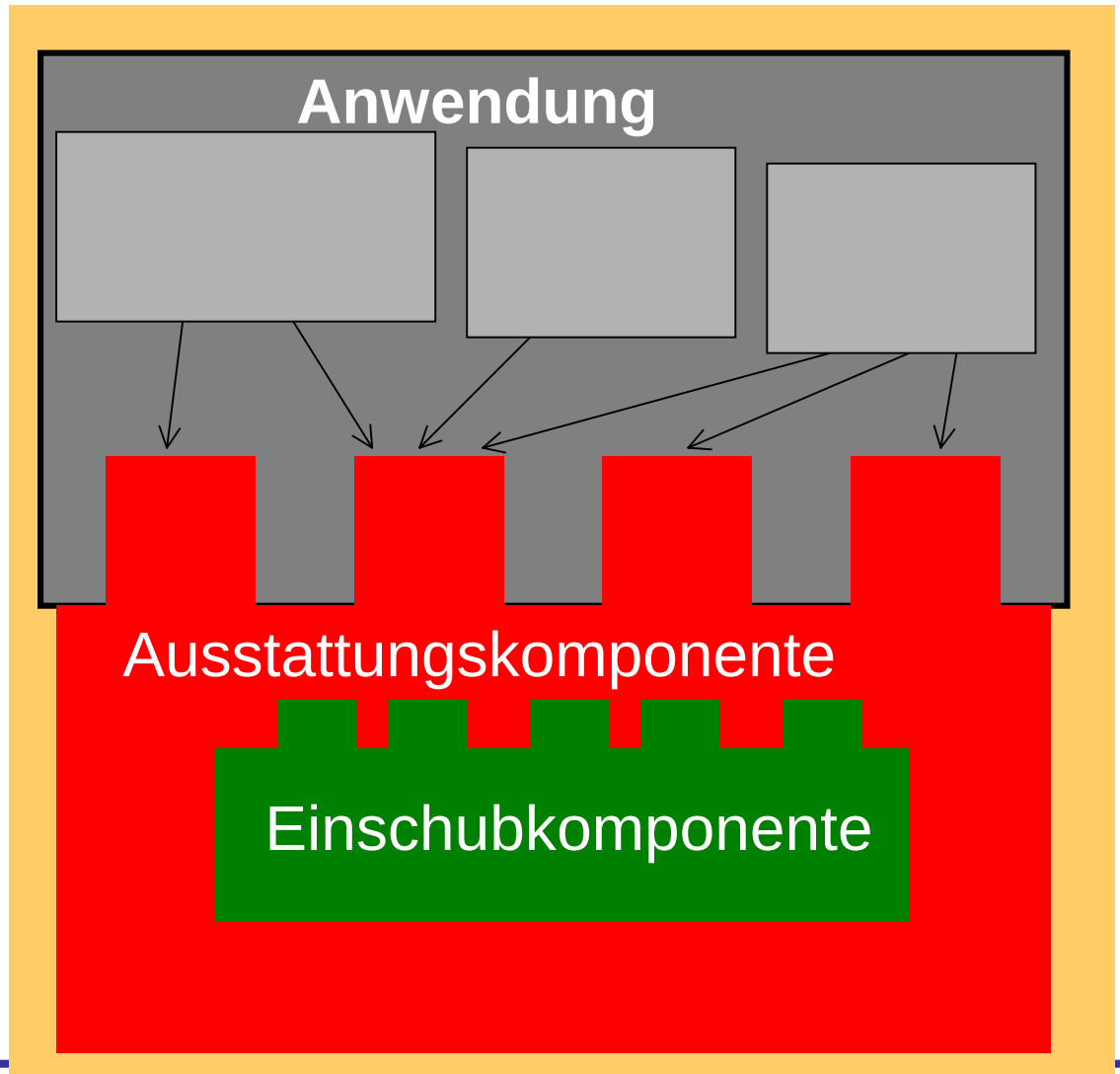
Ausstattungskomponenten sollen nicht auf eine bestimmte Technologie festgelegt sein (Persistenzmechanismus, Nachrichtenübermittlung, etc.)

- ➔ Technologiespezifische Teile der Ausstattungskomponente sind in einer ***Einschubkomponenten*** gekapselt

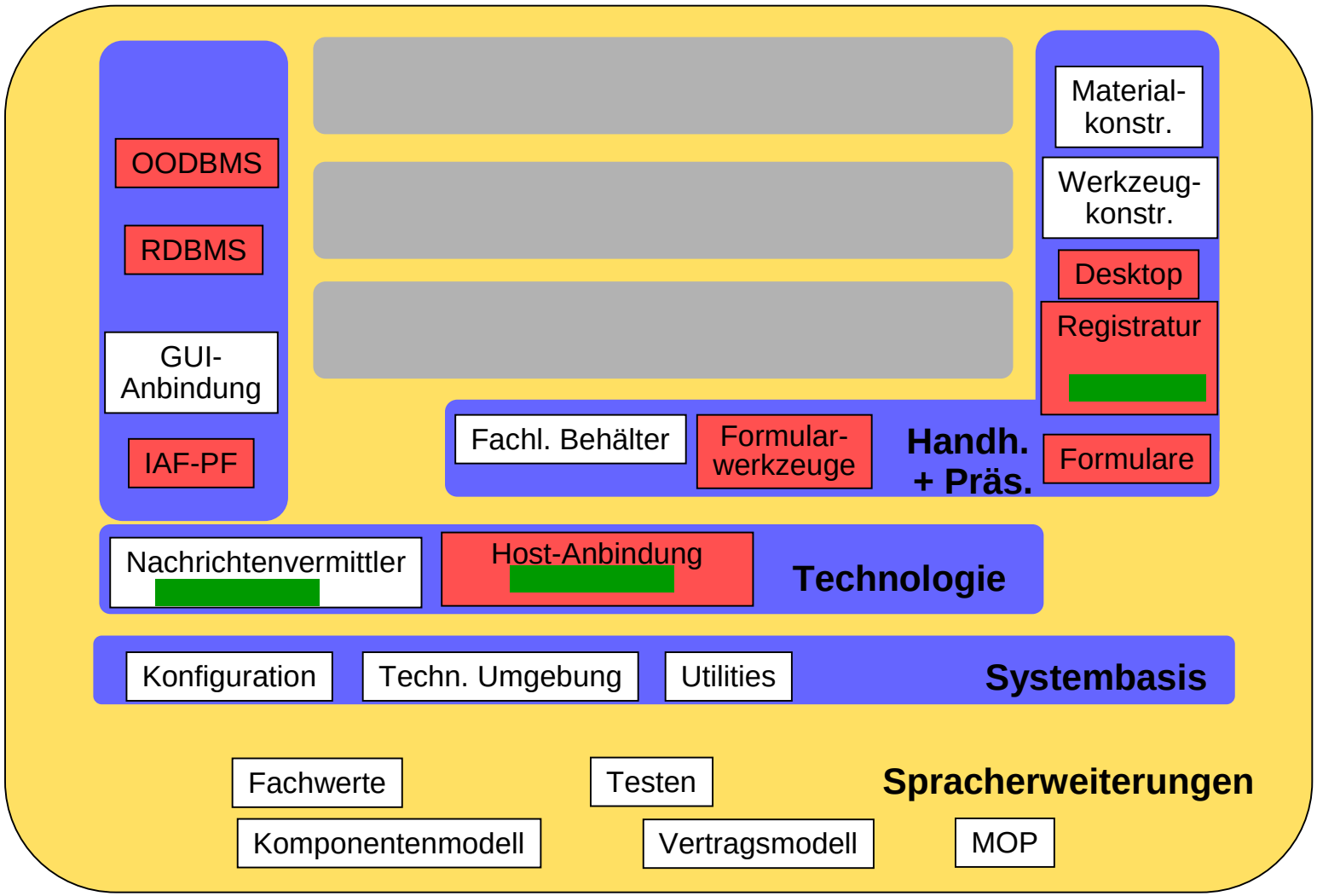
Einschubkomponenten

- ➔ sind Laufzeitkomponenten
- ➔ werden als Black Boxes von Anwendern oder Entwicklern verwendet
- ➔ kommunizieren nur mit Ausstattungskomponente und werden nicht von anderen Systemteilen her angesprochen
- ➔ unterstützen Skalierbarkeit über den Entwicklungszyklus

Komponenten in JWAM — Einschubkomponenten



Fazit


OODBMS
RDBMS
GUI-Anbindung
IAF-PF
Material-konstr.
Werkzeug-konstr.
Desktop
Registratur
Formulare
Fachl. Behälter
Formular-werkzeuge
Handh. + Präs.
Nachrichtenvermittler
Host-Anbindung
Technologie
Konfiguration
Techn. Umgebung
Utilities
Systembasis
Fachwerte
Testen
Spracherweiterungen
Komponentenmodell
Vertragsmodell
MOP

Fazit

JWAM-Komponenten-Konzept

- ➔ nicht top-down spezifiziert, sondern pragmatisch gewachsen
- ➔ Kombination von Framework- und Komponenten-Ansätzen (FW: Infrastruktur und K: Vermeidung von fatware)

Vorteile

- ➔ schlanker und besser verständlicher Framework-Kern
- ➔ fachlich höhere Schnittstellen auf Grundlage des Kerns
- ➔ flexible Skalierung mit alternativen Ausbaumöglichkeiten, da Komponenten nur über Konzepte des Kerns verbunden sind
- ➔ Trennung fachlicher und technologischer Dimensionen