

Tipps zur Integration von XML und Web Services

von Torben Deumert

Tipps zur Integration von XML

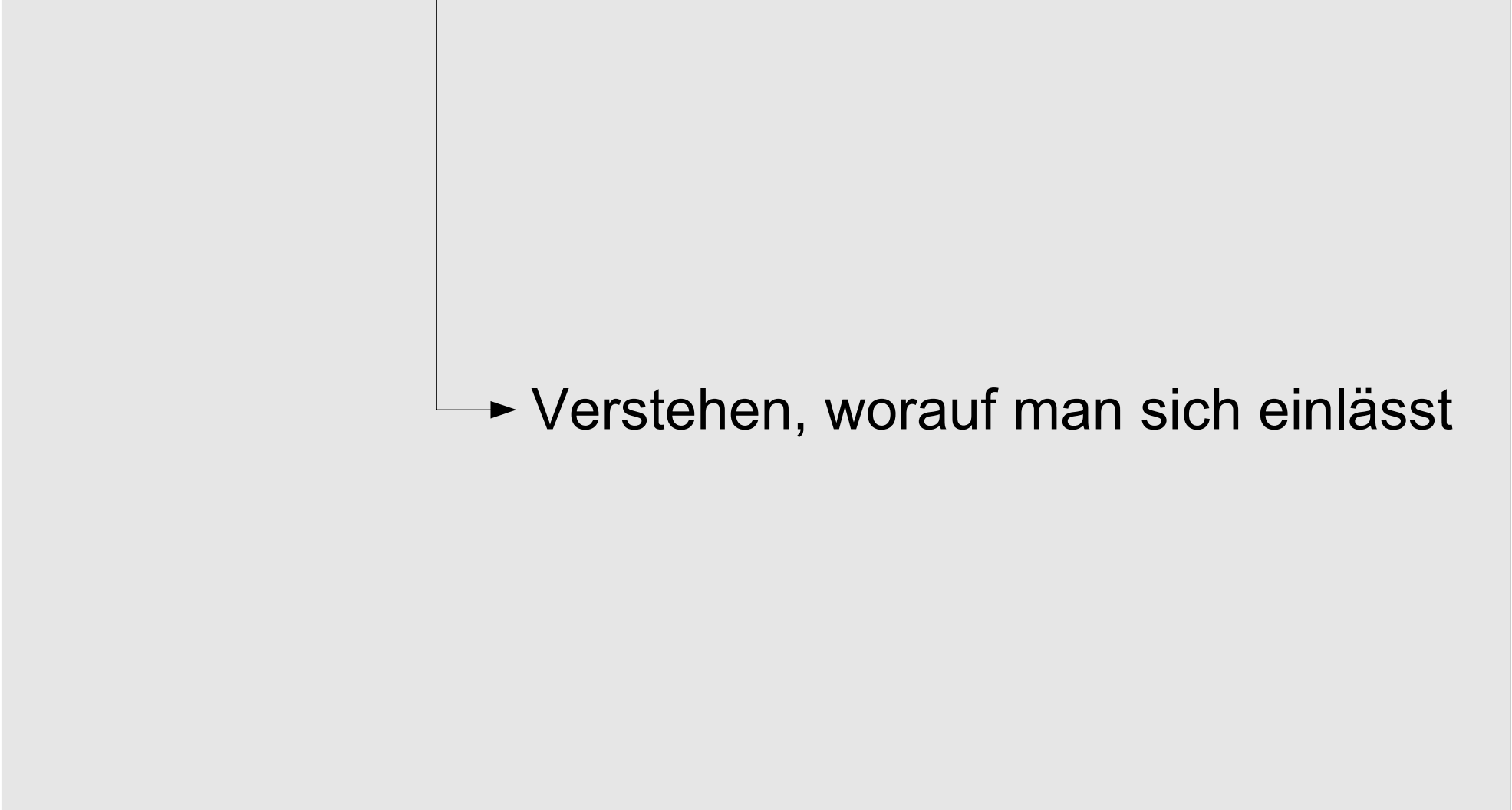
- Verstehen, worauf man sich einlässt
- Standards erstellen und einbeziehen
- XML zur Standardisierung des Datenzugriffs
- Bewertung von Tools vor der Integration
- Entwickeln eines Systems zur Wissens-Verbreitung

Tipps zur Integration von Web Services

- Wissen, WANN man Web Services verwenden sollte
- Wissen, WIE man Web Services verwenden sollte
- Wissen, WANN man Web Services NICHT verwenden sollte
- Fortschreiten mit einer Übergangs-Architektur
- Altlasten zu seinem Vorteil nutzen
- Um ein vernünftiges Sicherheits-Modell herumbauen

Tipps zur Integration von

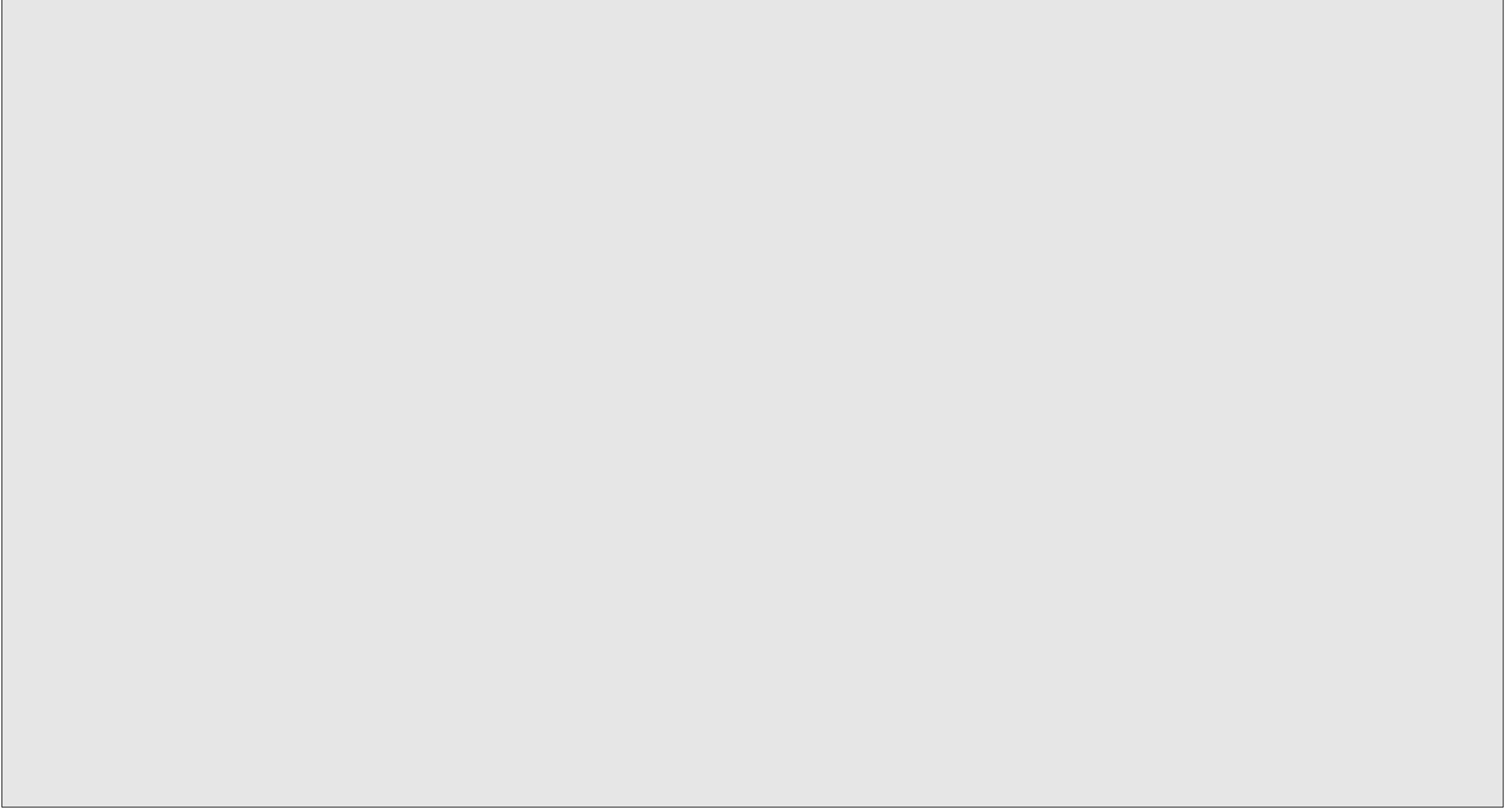
XML



→ Verstehen, worauf man sich einlässt



→ Verstehen, worauf man sich einlässt





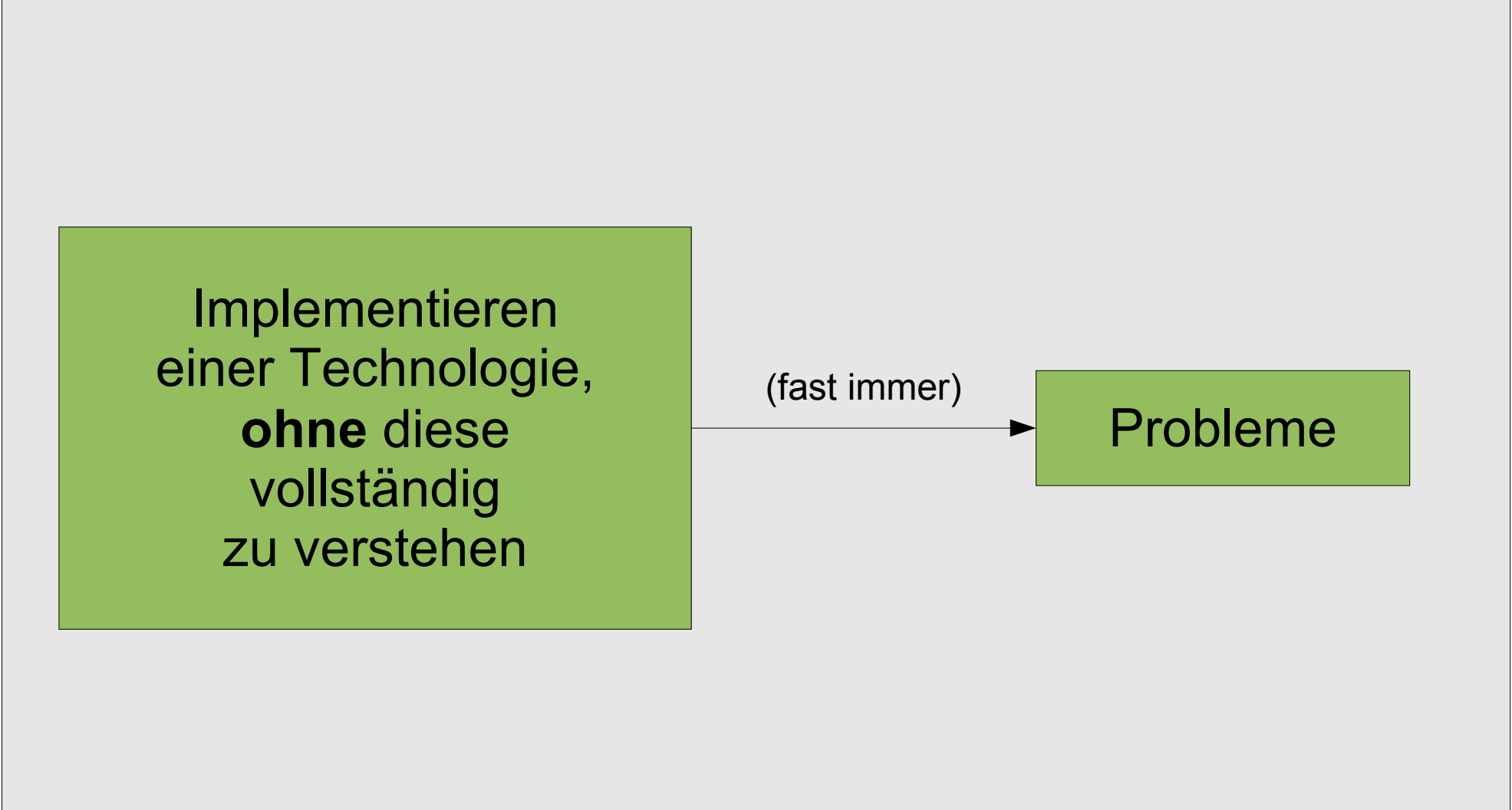
→ Verstehen, worauf man sich einlässt

Tipp:

Bestimme Deinen Kenntnisstand relevanter XML-Technologien vor der Planung eines Projekts.



→ Verstehen, worauf man sich einlässt





→ Verstehen, worauf man sich einlässt

Es ist wichtig, dass man versteht...

- ... wie und wo XML in die Applikation passt?
- ... welche Probleme gelöst werden, wenn XML integriert ist?



→ Verstehen, worauf man sich einlässt

Eine kleine Übung:

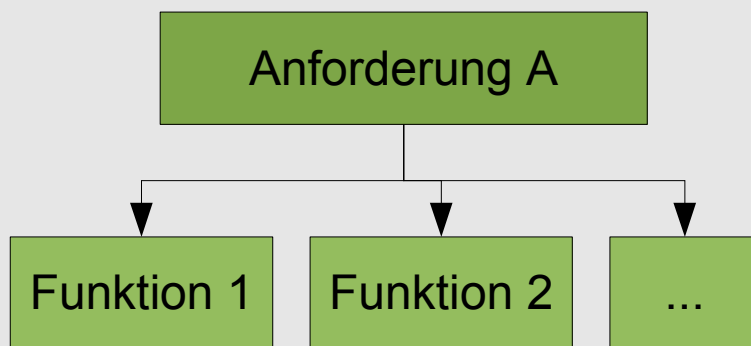
1. Beschreiben, wie XML dabei helfen kann, Projekt-Aufgaben zu erfüllen

→ Verstehen, worauf man sich einlässt

Eine kleine Übung:

1. Beschreiben, wie XML dabei helfen kann, Projekt-Aufgaben zu erfüllen

- a) Abbilden von Anforderungen auf Funktionen

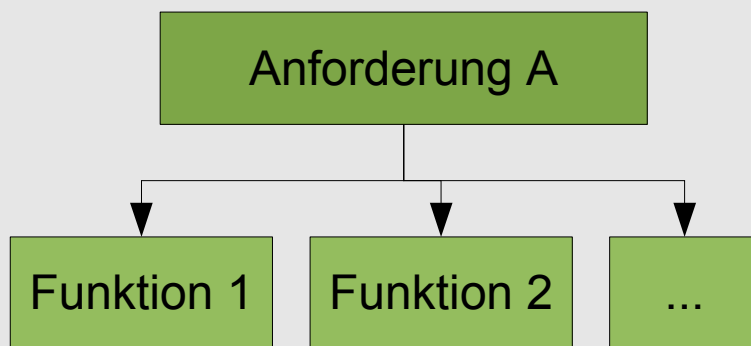


► Verstehen, worauf man sich einlässt

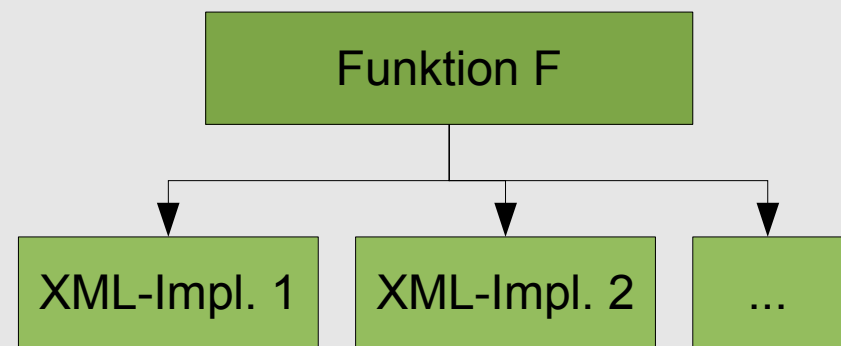
Eine kleine Übung:

1. Beschreiben, wie XML dabei helfen kann, Projekt-Aufgaben zu erfüllen

a) Abbilden von Anforderungen auf Funktionen



b) Abbilden von Funktionen auf XML-Implementationen



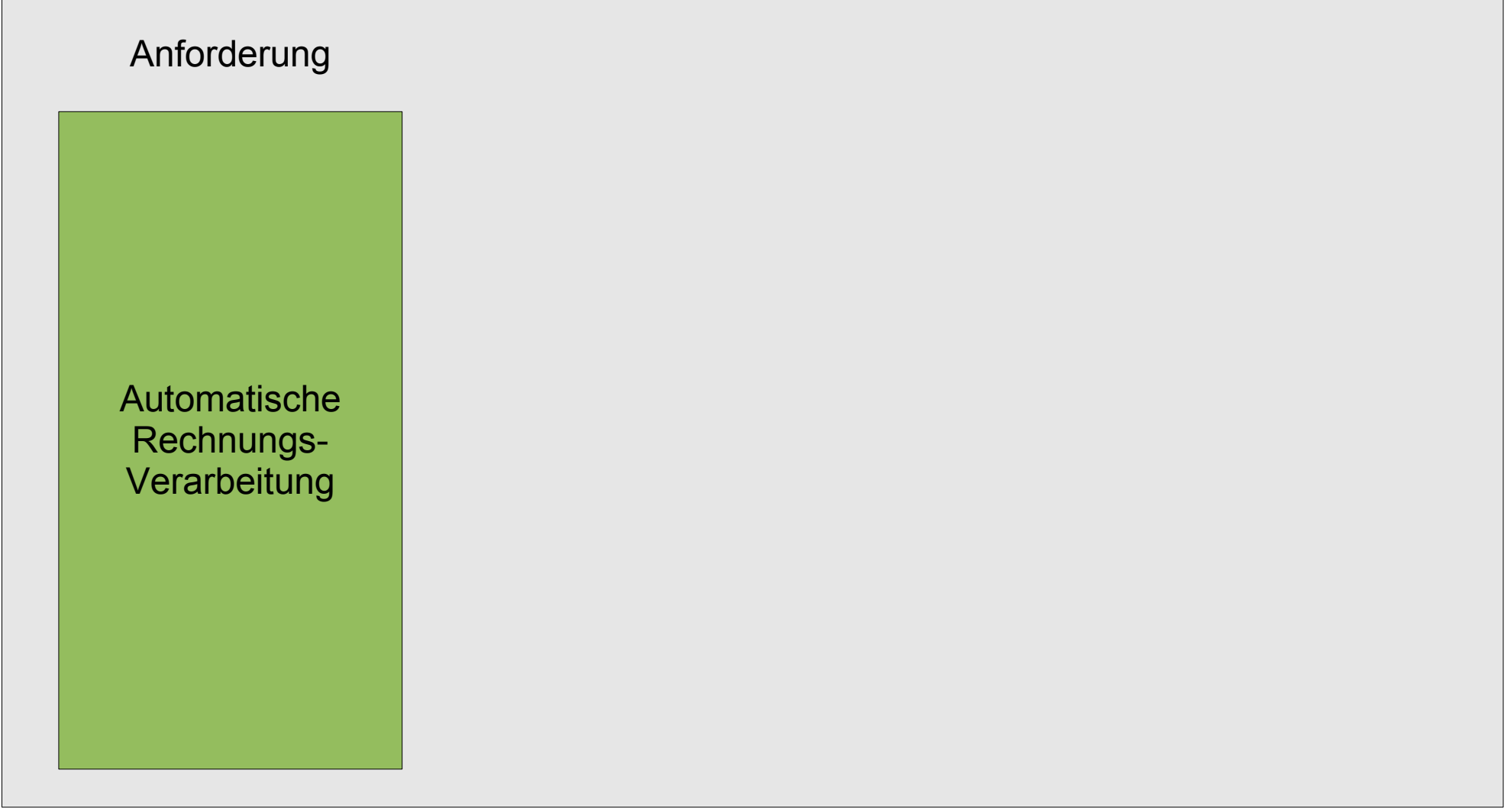


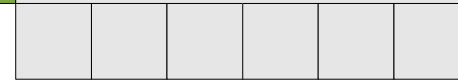
→ Verstehen, worauf man sich einlässt

Ein Beispiel ...

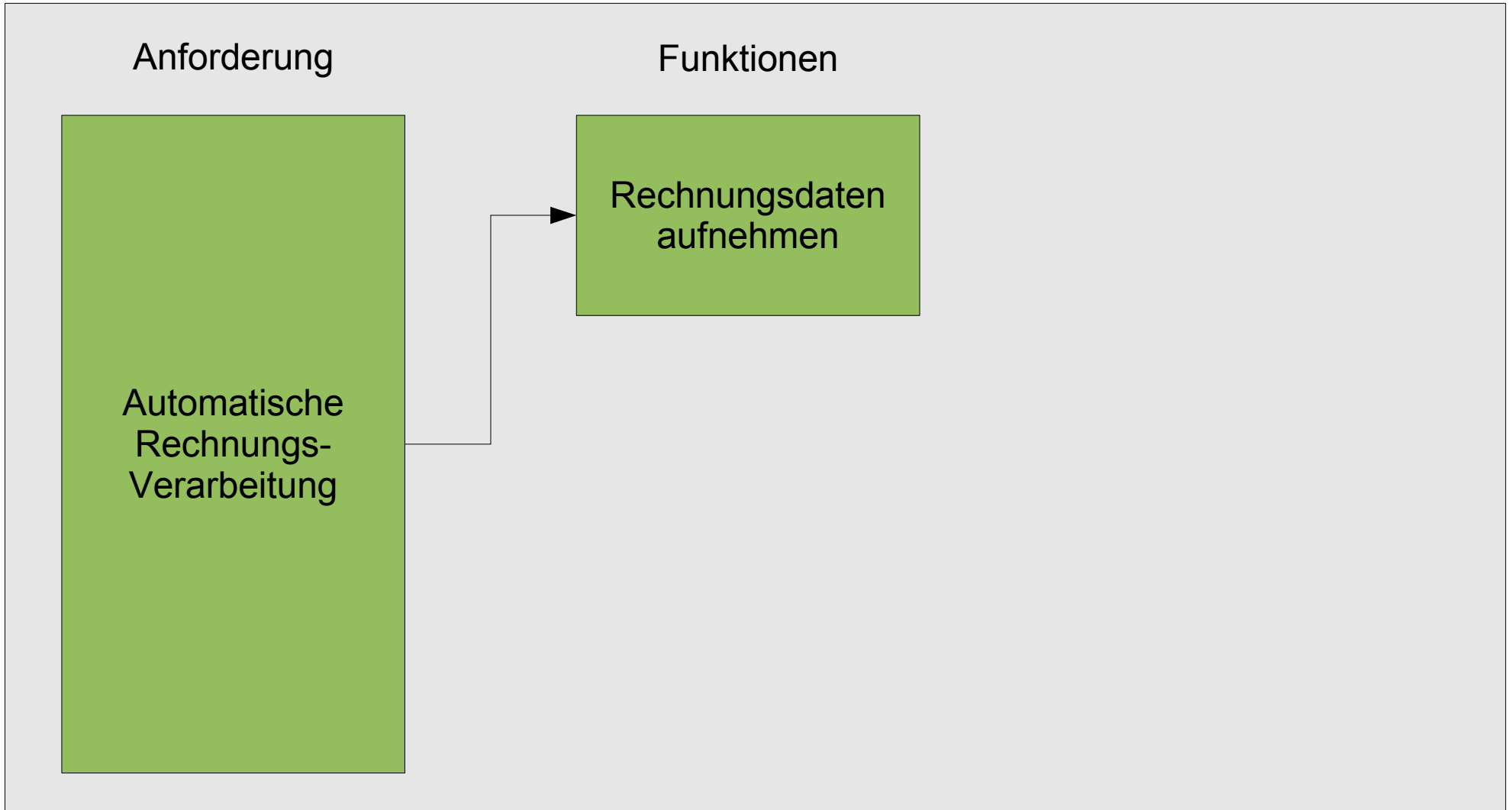


→ Verstehen, worauf man sich einlässt



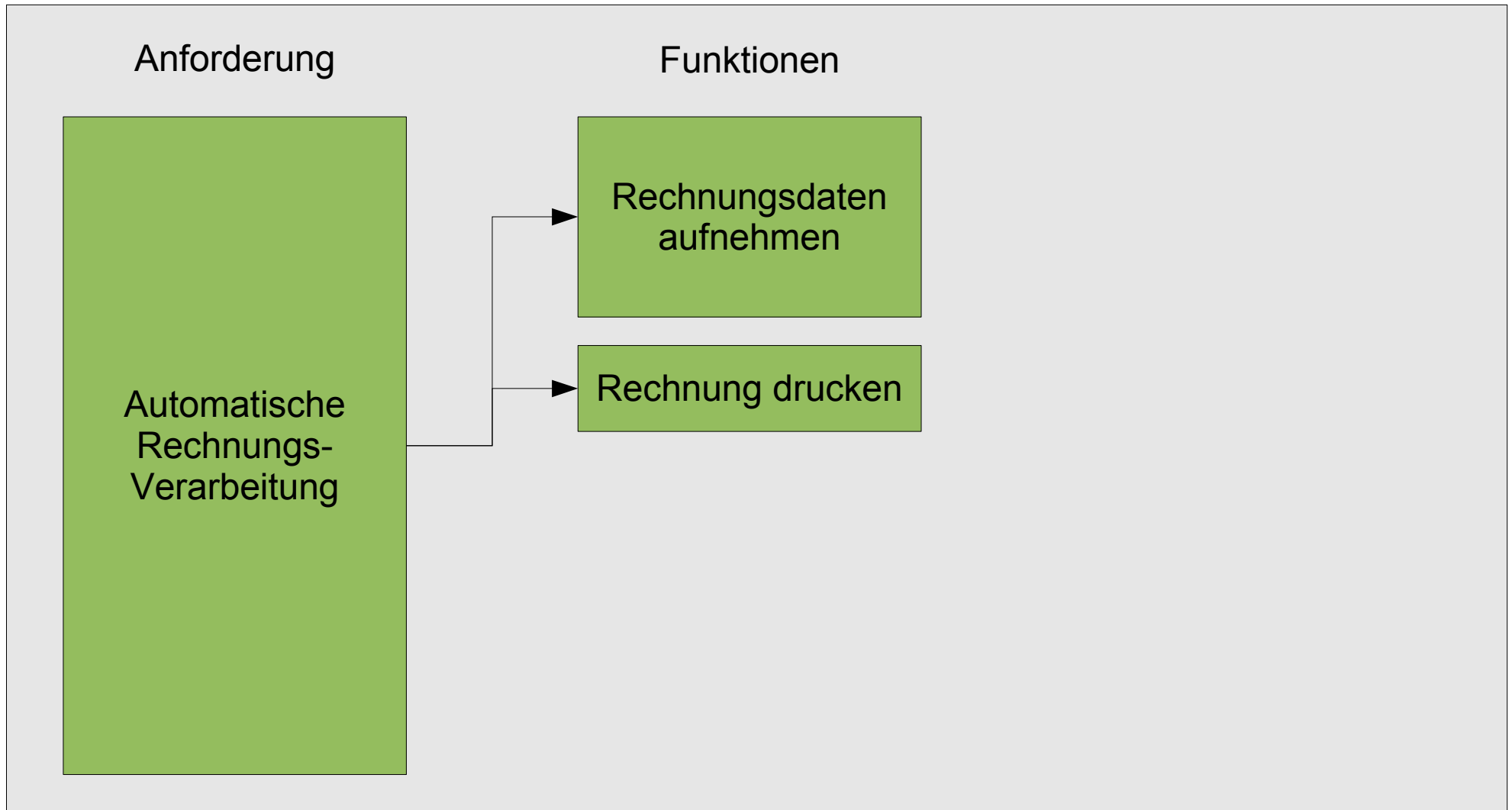


➔ Verstehen, worauf man sich einlässt

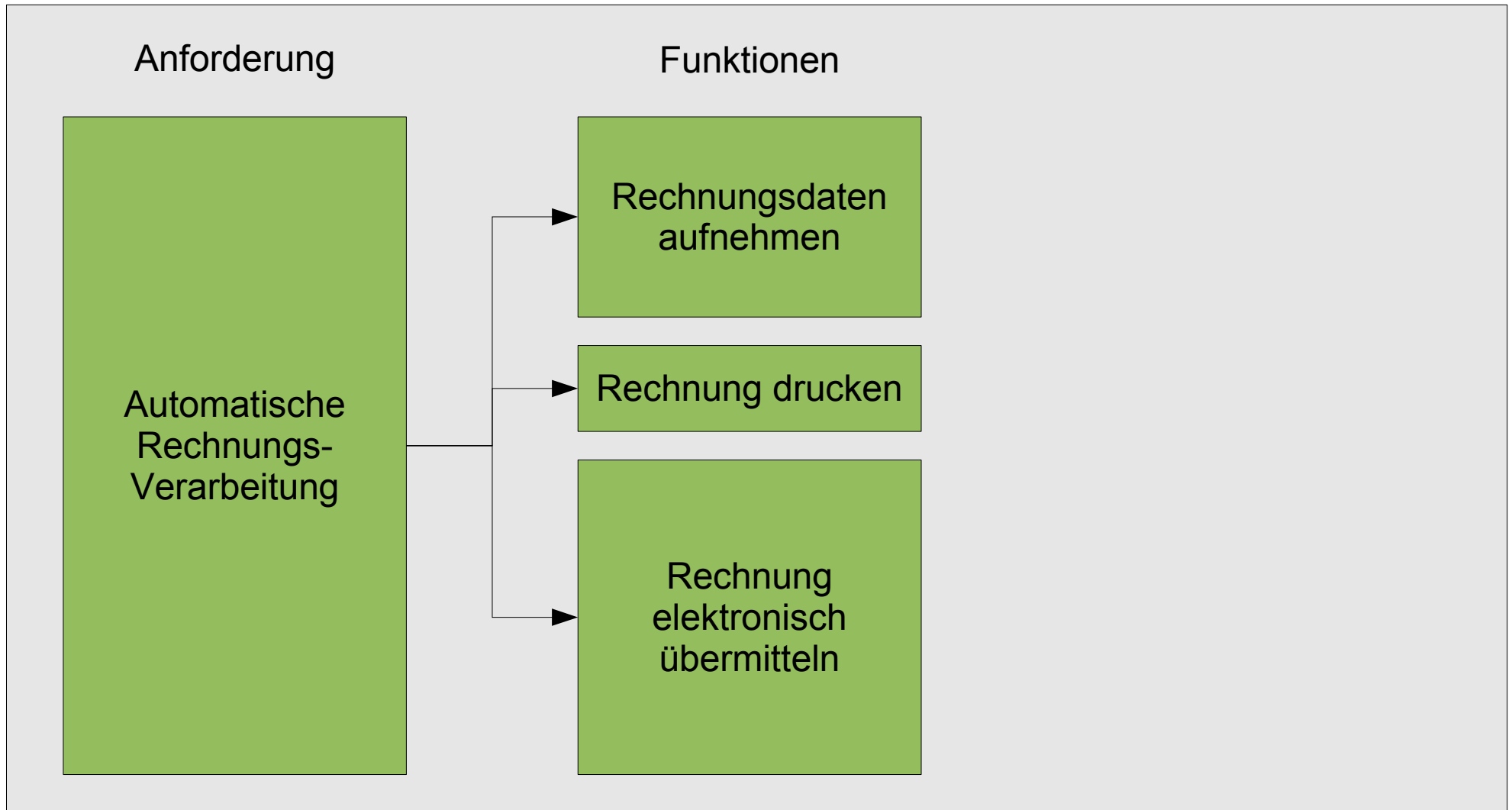




➔ Verstehen, worauf man sich einlässt

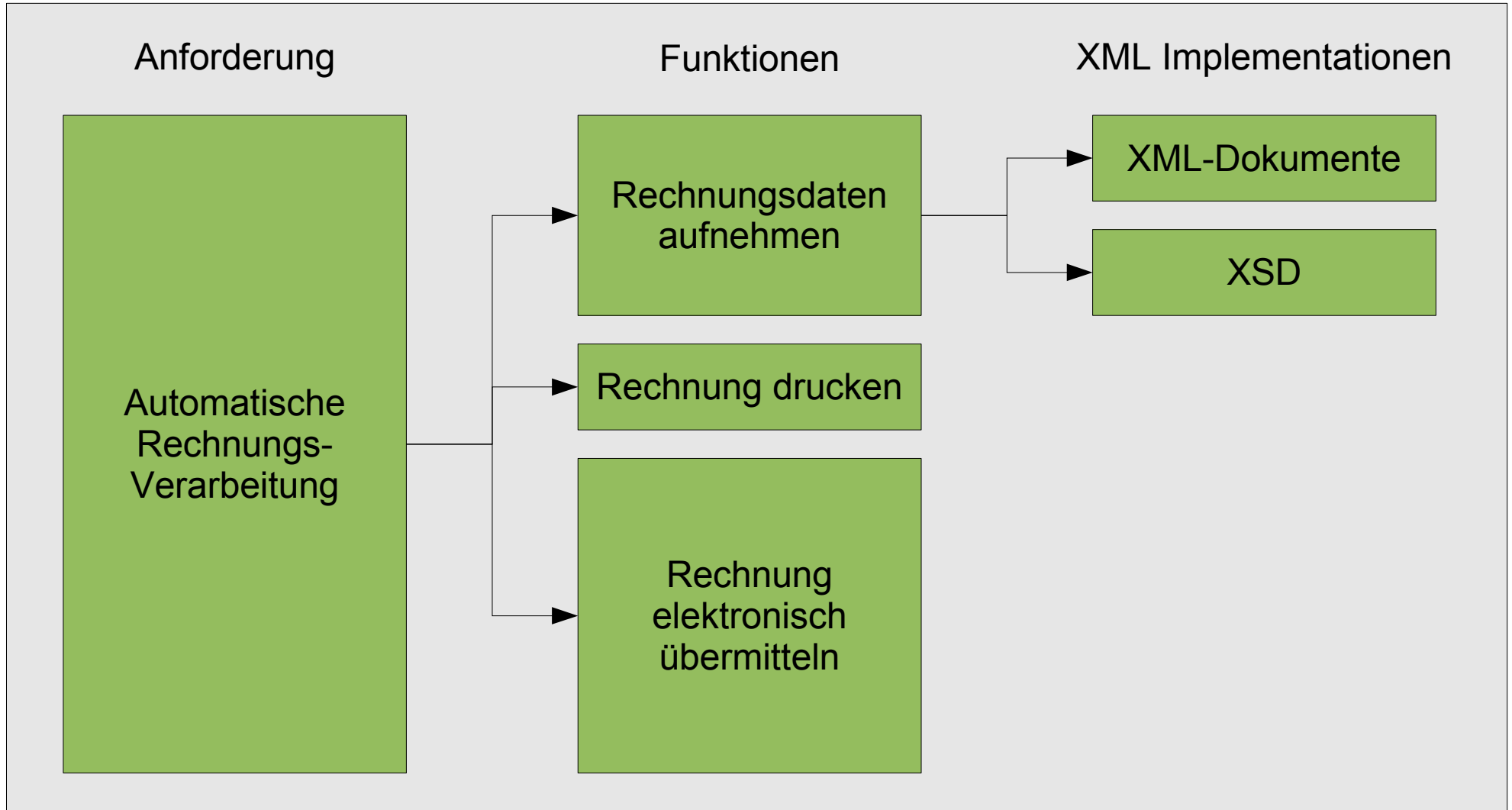


➔ Verstehen, worauf man sich einlässt

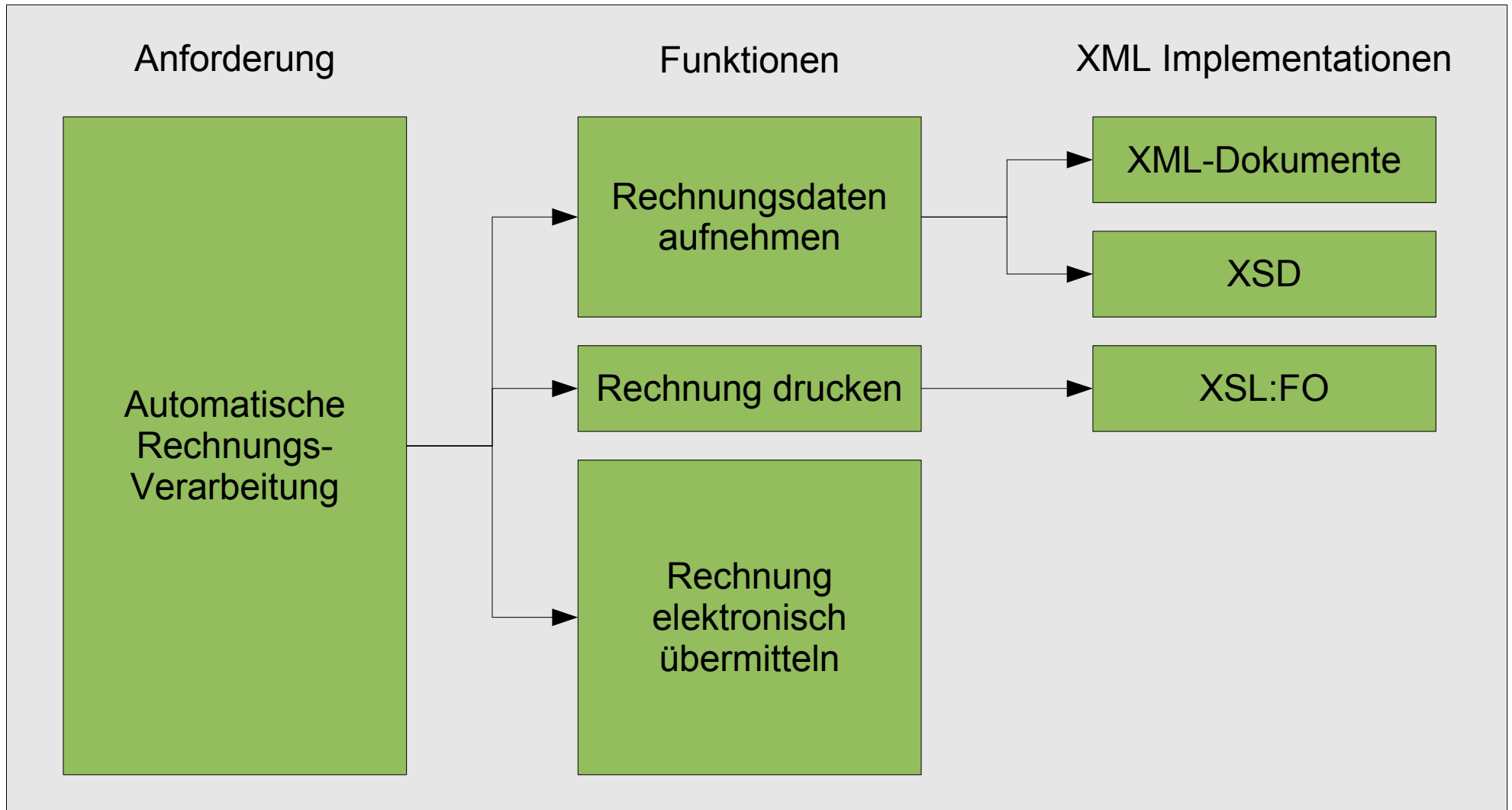




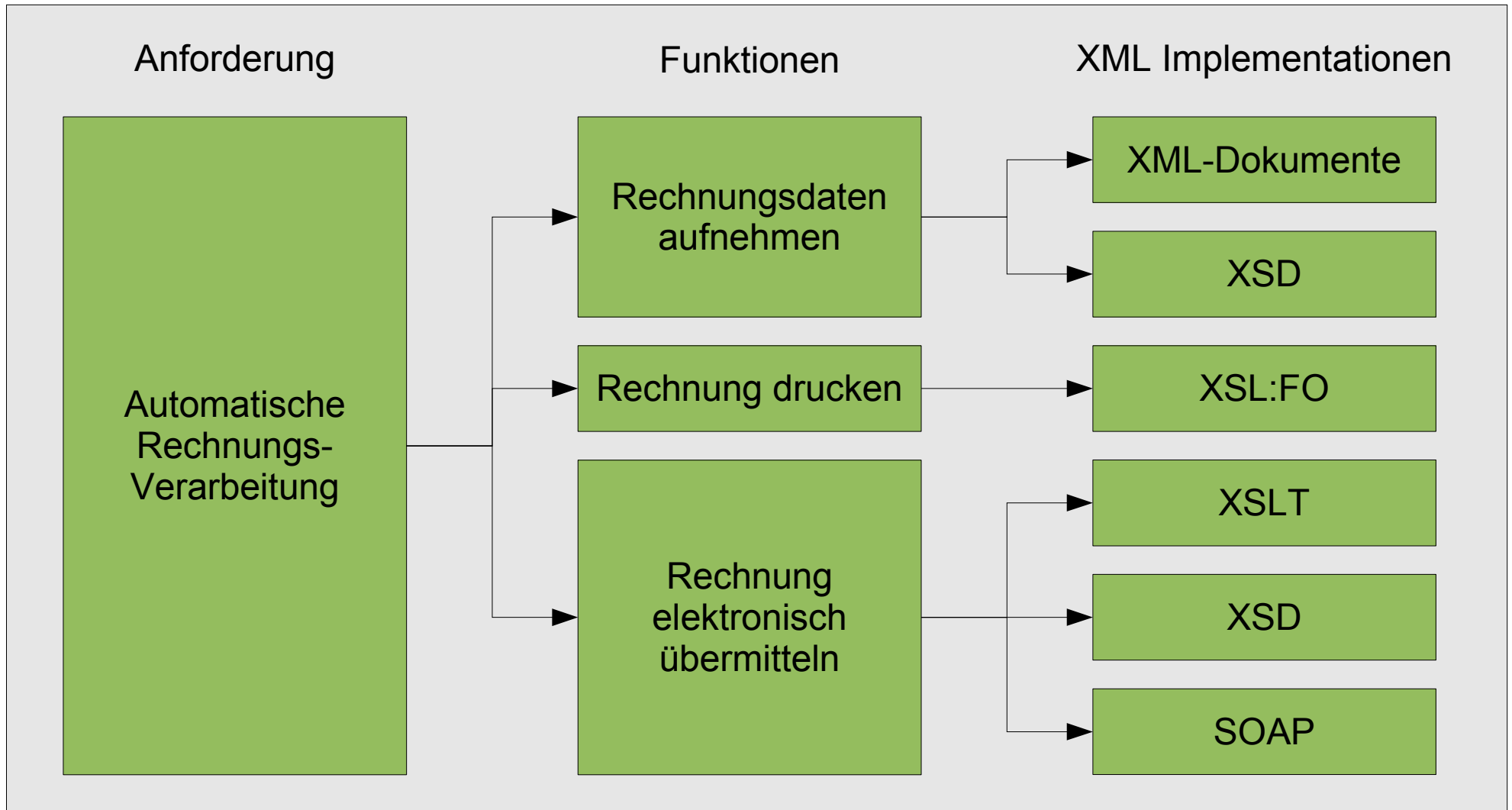
➔ Verstehen, worauf man sich einlässt



► Verstehen, worauf man sich einlässt



► Verstehen, worauf man sich einlässt





→ Verstehen, worauf man sich einlässt

Eine kleine Übung (Fortsetzung):

2. Auflisten der vorgeschlagenen XML-Technologien und beschreiben ihres allgemeinen Zwecks.



→ Verstehen, worauf man sich einlässt

Eine kleine Übung (Fortsetzung):

2. Auflisten der vorgeschlagenen XML-Technologien und beschreiben ihres allgemeinen Zwecks.
3. Begründen, wie die Fähigkeiten das aktuelle Team für die Arbeit mit diesen Technologien qualifiziert.



→ Verstehen, worauf man sich einlässt

Eine kleine Übung (Fortsetzung):

2. Auflisten der vorgeschlagenen XML-Technologien und beschreiben ihres allgemeinen Zwecks.
3. Begründen, wie die Fähigkeiten das aktuelle Team für die Arbeit mit diesen Technologien qualifiziert.
4. Einholen einer zweiten Meinung von einem qualifizierten Fachmann.



→ Verstehen, worauf man sich einlässt

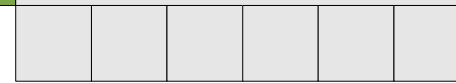
Einige Richtlinien:



→ Verstehen, worauf man sich einlässt

Einige Richtlinien:

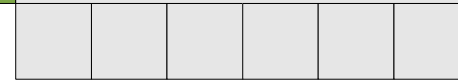
- Funktionen teilen, wenn sie zu komplex sind. (Der Grad der Komplexität sollte einigermaßen einheitlich sein.)



→ Verstehen, worauf man sich einlässt

Einige Richtlinien:

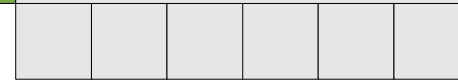
- Funktionen teilen, wenn sie zu komplex sind. (Der Grad der Komplexität sollte einigermaßen einheitlich sein.)
- Nicht auf “Front-End“-Funktionalität konzentrieren (statt Einzelheiten lieber Zusammenfassungen wie “Benutzer übermittelt Daten”)



→ Verstehen, worauf man sich einlässt

Einige Richtlinien:

- Funktionen teilen, wenn sie zu komplex sind. (Der Grad der Komplexität sollte einigermaßen einheitlich sein.)
- Nicht auf “Front-End“-Funktionalität konzentrieren (statt Einzelheiten lieber Zusammenfassungen wie “Benutzer übermittelt Daten”)
- Beim ersten Durchführen der Übung nur Kern-Funktionen beachten.



→ Verstehen, worauf man sich einlässt

Einige Richtlinien:

- Funktionen teilen, wenn sie zu komplex sind. (Der Grad der Komplexität sollte einigermaßen einheitlich sein.)
- Nicht auf “Front-End“-Funktionalität konzentrieren (statt Einzelheiten lieber Zusammenfassungen wie “Benutzer übermittelt Daten”)
- Beim ersten Durchführen der Übung nur Kern-Funktionen beachten.
- Bei der Team-Bewertung auch verfügbare Tools beachten, die u.U. verwendet werden.

classifiedBallDes	
E number	ballType
E detected	boolean
E x-position	decimal
E y-position	decimal
E info	string

classifiedBalls (classifiedBalls)	
E ball	classifiedBallD

calibrateBall ballType	
------------------------	--

ball classifiedBallDescriptionType	
E number	ballType
E detected	boolean
E x-position	decimal
E y-position	decimal
E info	string

Index

Suchen nach:

XSD-Schemas

Filter:

(Kein Filter)

XSD-Dateien

- xsd-info-available funktion
- xsd-Präfix in XML-Schemas
- XSD-Schemas
- (Siehe auch XML-Schemas)
- Erstellen einfacher Typen
- Erstellen von benutzerdefinierten XM
- Hinzufügen von Beziehungen
- Hinzufügen von Einschränkungen in

Pr... Kl... In... In...

Dynamische Hilfe

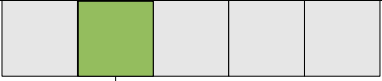
Hilfe

- XML-Designer
- schema
- <Schema> Node (<SubscriptionCla
- <Schema> Node (<NotificationClas
- <Schema> Node (<EventClass>)
- IADs Property Methods
- Beispiele
- Visual Studio-Beispiele
- Erste Schritte
- Exemplarische Vorgehensweisen fü
- Installieren der Hilfe zu Visual Studi

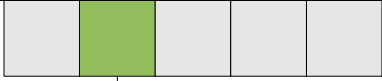
Schema XML

Aufgabenliste - 0 Buildfehler angezeigte Aufgaben (gefiltert)

!	Beschreibung	Datei	Zeile
	Klicken Sie hier, um eine neue Aufgabe hinzuzufügen		



→ Standards erstellen und einbeziehen



► Standards erstellen und einbeziehen

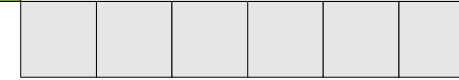
Standards sind wie Ampeln



► Standards erstellen und einbeziehen

Tipp:

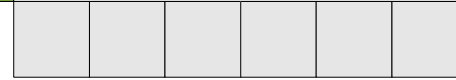
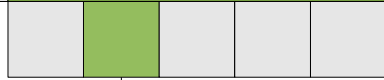
Standardisiere immer die Integration von XML Technologien.



► Standards erstellen und einbeziehen

Häufige Standards:

- Entwicklungs-Standards
- Anwendungs-Design-Standards
- Architektur-Standards
- Infrastruktur-Standards
- Wartungs-Standards
- Projekt-Standards
- Organisations-Standards
- Unternehmens-Standards



► Standards erstellen und einbeziehen

Entwicklungs-Standards

► Standards erstellen und einbeziehen

Entwicklungs-Standards

- Namenskonventionen



Benennung von
Klassen,
Eigenschaften
und
Funktionen

► Standards erstellen und einbeziehen

Entwicklungs-Standards

- Namenskonventionen
- Exception-Handling

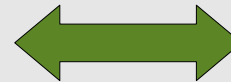


Wo und Wie werden Fehler
bearbeitet

► Standards erstellen und einbeziehen

Entwicklungs-Standards

- Namenskonventionen
- Exception-Handling
- Grammatik-Standards



Wie werden Sprachkonstrukte verwendet?

Beispiele:

Vermeiden negativen
Vergleiche
(== statt !=)

Keine Verschachtelung von
if-Anweisungen für
Fehlerfälle

► Standards erstellen und einbeziehen

Entwicklungs-Standards

- Namenskonventionen
- Exception-Handling
- Grammatik-Standards
- Design-Standards



ausschließlich:
Dokumentenstrukturen
Funktionsaufteilung
(z.B. Modularisierung)



► Standards erstellen und einbeziehen

Anwendungs-Design-Standards

- Standards
- Richtlinien
- Muster

} wie und wann soll welche XML-Technologie benutzt werden (einschließlich Alternativen)



► Standards erstellen und einbeziehen

Architektur-Standards

- Standard-Architektur für den Entwurf zukünftiger Entwicklungs-Projekte:
 - Positionierung der XML-Technologien (konzeptionell und physikalisch)
 - Beziehungen zwischen XML-Technologien und Problemen der Integration und Interoperabilität



► Standards erstellen und einbeziehen

Infrastruktur-Standards

- Hosting-Umgebungen für Anwendungen:
 - Server-Konfiguration
 - Richtlinien für den Internet-Zugriff
 - Sicherheit
 - Konfiguration
 - Skalierbarkeit

ggfs.:

- Entwicklungs-Umgebung und entsprechende Tools



► Standards erstellen und einbeziehen

Wartung-Standards

- Administrations-Abläufe
- Verwaltung von Benutzer-Konten
- Skalierbarkeit
- Verfügbarkeit
- Routine-Aufgaben der Wartung



► Standards erstellen und einbeziehen

Projekt-Standards

- Qualifikations-Anforderungen
- Prozesse
- Ermitteln aller bisher erwähnten Standards, die für das neue Projekt relevant sind (inkl. Abweichungen vom bisherigen Anwendungs-Design und von Entwicklungs-Konventionen)



► Standards erstellen und einbeziehen

Organisations-Standards

- Neue organisatorische Einheiten (Stellen oder sogar Abteilungen), die für verschiedene Aspekte der Verwendung und Wartung der XML-Technologien verantwortlich sind
- Festsetzen, welche Entwicklungs-Tools und Editoren vom Projekt-Team verwendet werden dürfen.
- Zuordnung bisheriger Standards



► Standards erstellen und einbeziehen

Unternehmens-Standards

- Typische “High-Level”-Anforderungen (aus Langzeit-Strategien)

Unternehmens-Strategie:

XML soll innerhalb von
9 Monaten zur
Schlüsseltechnologie werden

Unternehmens-Standard:

XML muss innerhalb von
3 Monaten in einem
abgegrenzten Bereich
verwendet werden



XML zur Standardisierung
des Datenzugriffs



XML zur Standardisierung des
Datenzugriffs

Tipp:

Das Maximieren der Verwendung von XML im
“Data Access Layer”
erhöht die Kontrolle des Daten-Managements
und
standardisiert die Applikations-Architektur



XML zur Standardisierung des
Datenzugriffs

Der Zweck von XML:

Repräsentation von Daten

Beim Design einer XML-Architektur:

Viel Aufwand für die Vereinheitlichung heterogener Daten



“Daten-Repräsentation” wird zum zentralen Aspekt



XML zur Standardisierung des
Datenzugriffs

Es gibt Gründe, um weiter zu gehen:

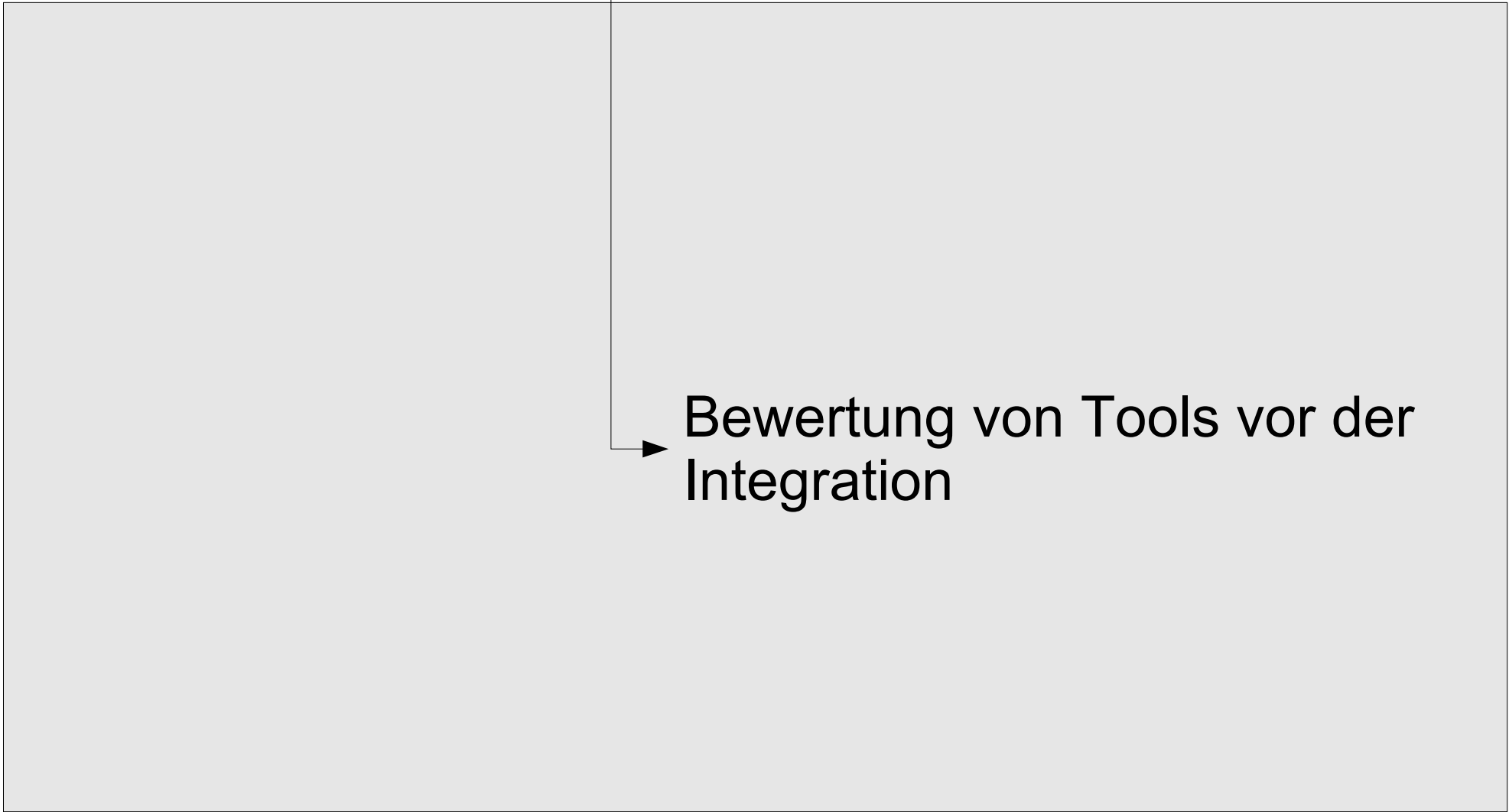
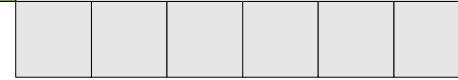
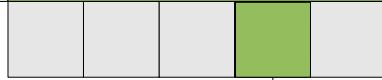
XML zur Vereinheitlichung der Programm-Logik



XML zur Standardisierung des Datenzugriffs

Vorteile:

- Anwendungen müssen sich nicht mehr um Datenbank-Anfragen und XML-Parsing kümmern
- Verringerung der Anforderungen für Teile von Applikationen, in denen dies realisiert wird.



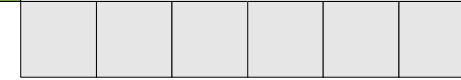
→ Bewertung von Tools vor der Integration



Bewertung von Tools vor der Integration

Tipp:

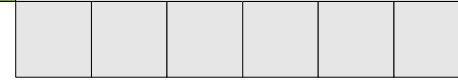
Bewerte die den Tools
zu Grunde liegende Arbeitsweise vorsichtig.
Vermeide jene mit proprietären Haken



→ Bewertung von Tools vor der Integration

Tools erlauben es, XML zu ...

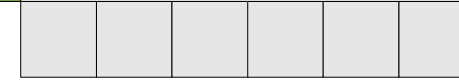
- editieren
- verwalten
- generieren



Bewertung von Tools vor der
Integration

Vorsicht ist geboten:

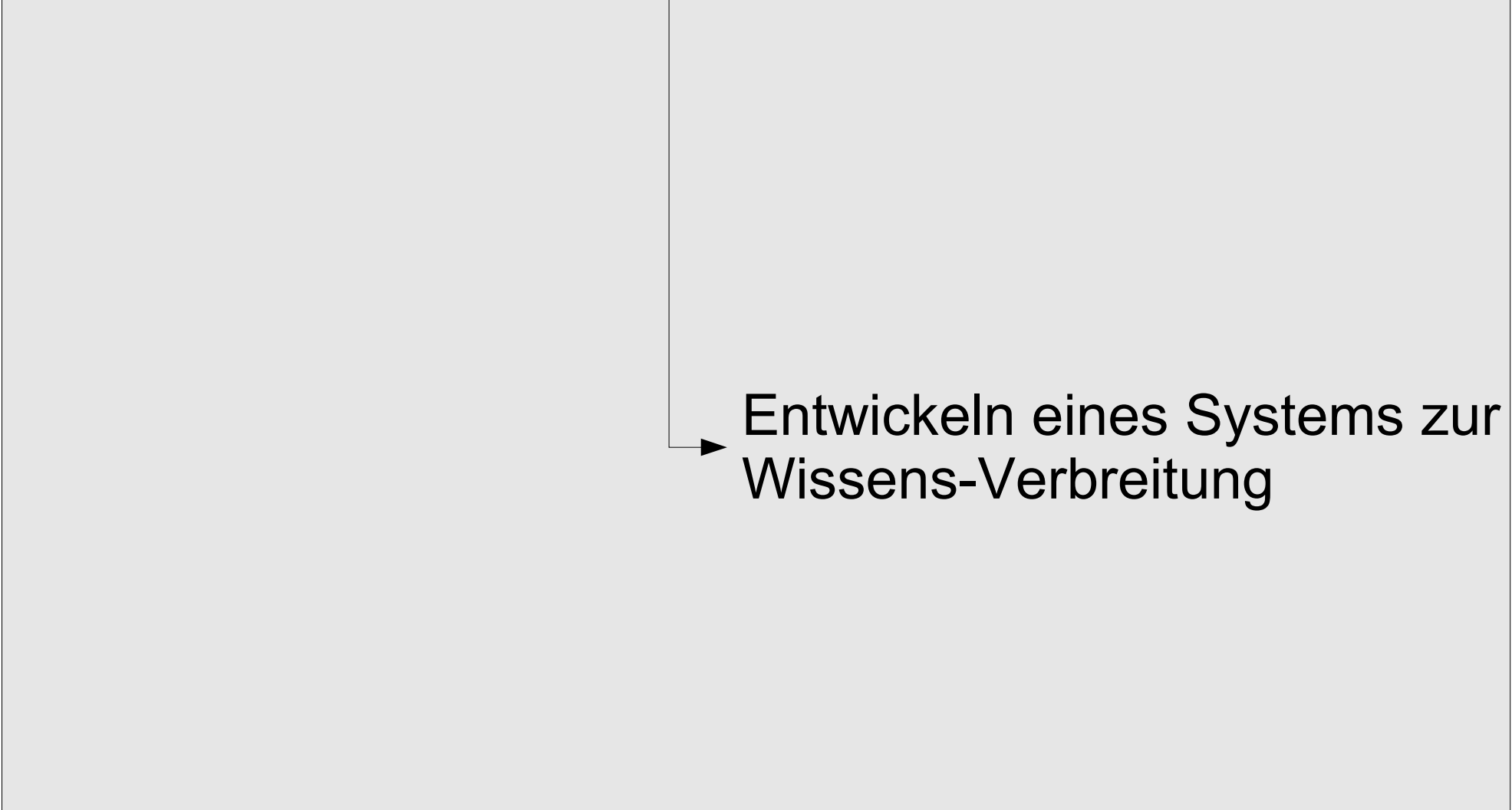
Viele führen proprietäre Erweiterungen und Dateiformate ein, die einen an diesen Hersteller binden.



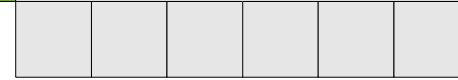
→ Bewertung von Tools vor der Integration

Vor der Aufnahme eines Tools in einen Standard:

- Interne Arbeitsweise der Tools analysieren und bewerten
- Kompatibilität zur bisherigen Entwicklungsplattform testen



Entwickeln eines Systems zur
Wissens-Verbreitung



Entwickeln eines Systems zur
Wissens-Verbreitung

Tipp:

Implementiere ein System zur Wissens-
Verbreitung, um die Kommunikation zu
verbessern und die Standards aktuell zu
halten.

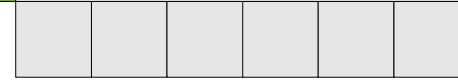
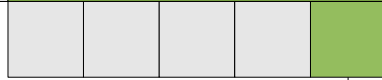


Entwickeln eines Systems zur
Wissens-Verbreitung

Wie läuft es normalerweise?

- Projekt-Teams gehen mit Herausforderungen individuell um
- Die Kompatibilität der verwendeten Umgebung ist einzigartig

Ein System, um derartige Information zu speichern und zu teilen kann extrem förderlich sein



Entwickeln eines Systems zur Wissens-Verbreitung

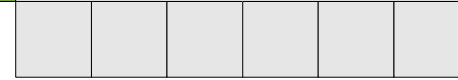
Projekt A

Projekt B

Projekt C

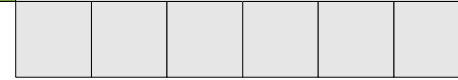
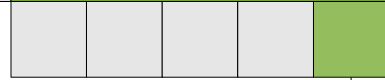
Standards

Standards

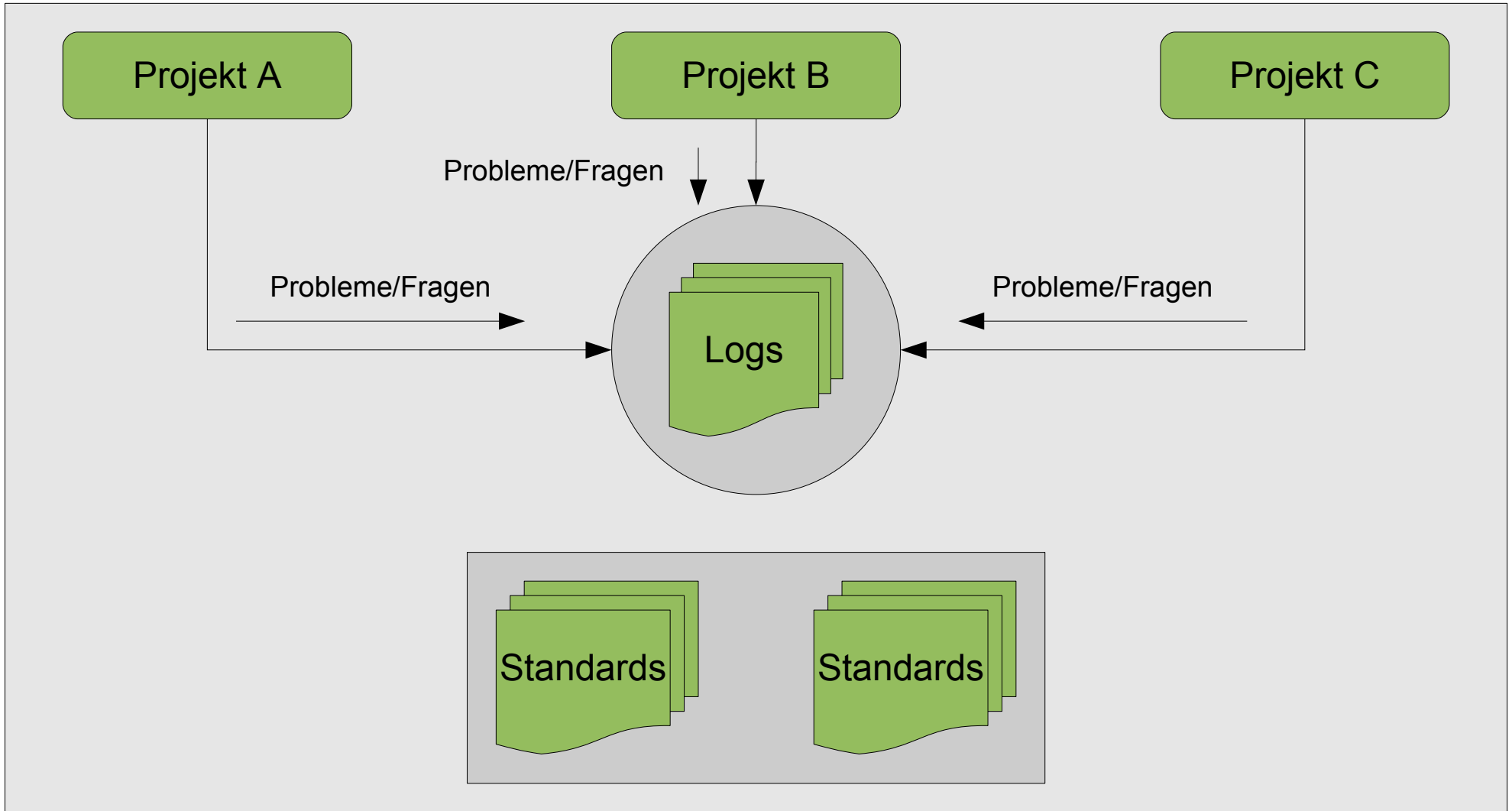


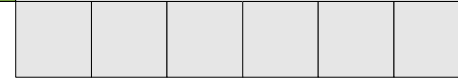
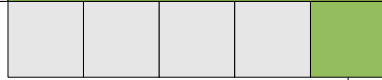
Entwickeln eines Systems zur
Wissens-Verbreitung



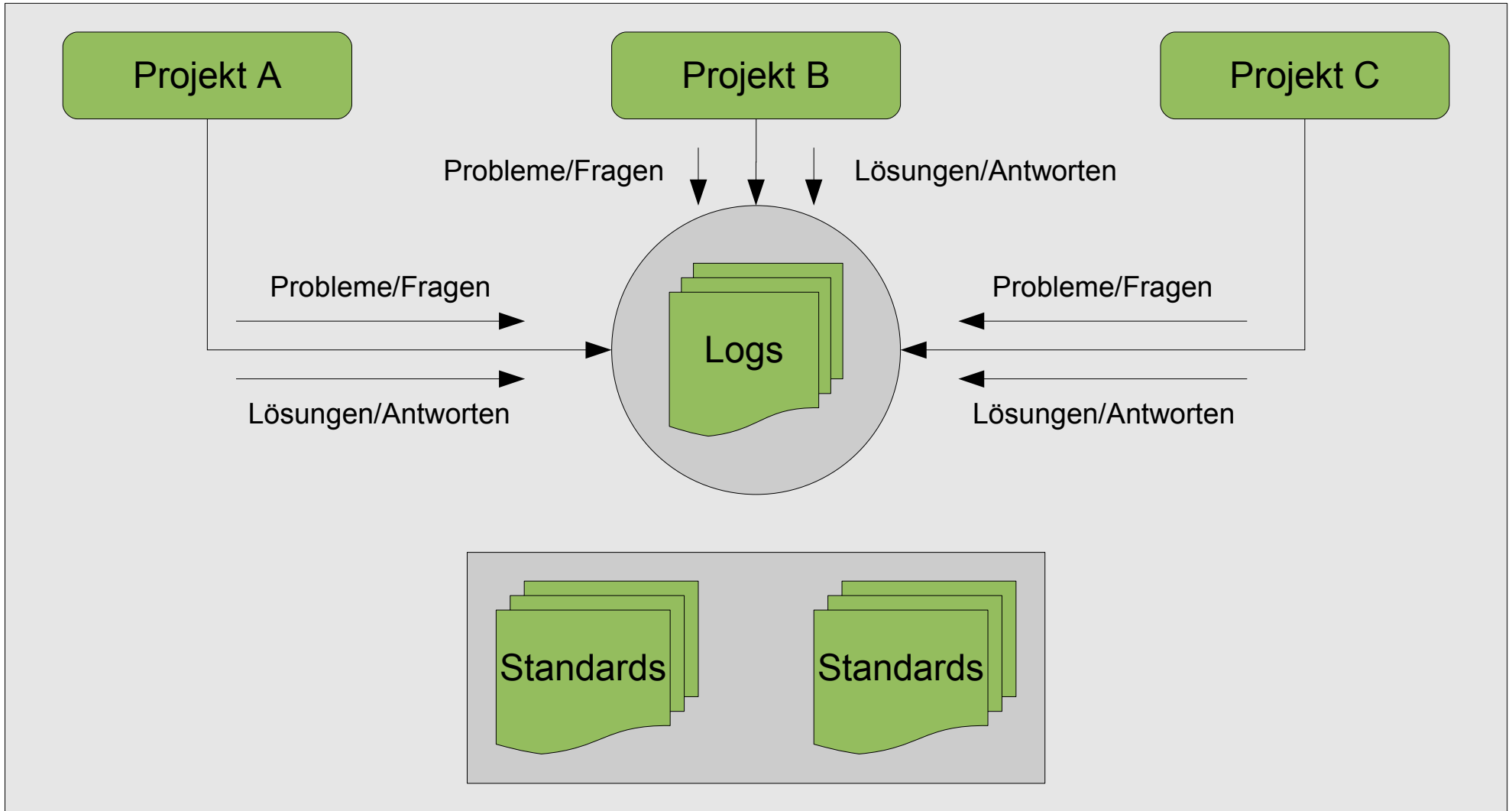


Entwickeln eines Systems zur Wissens-Verbreitung



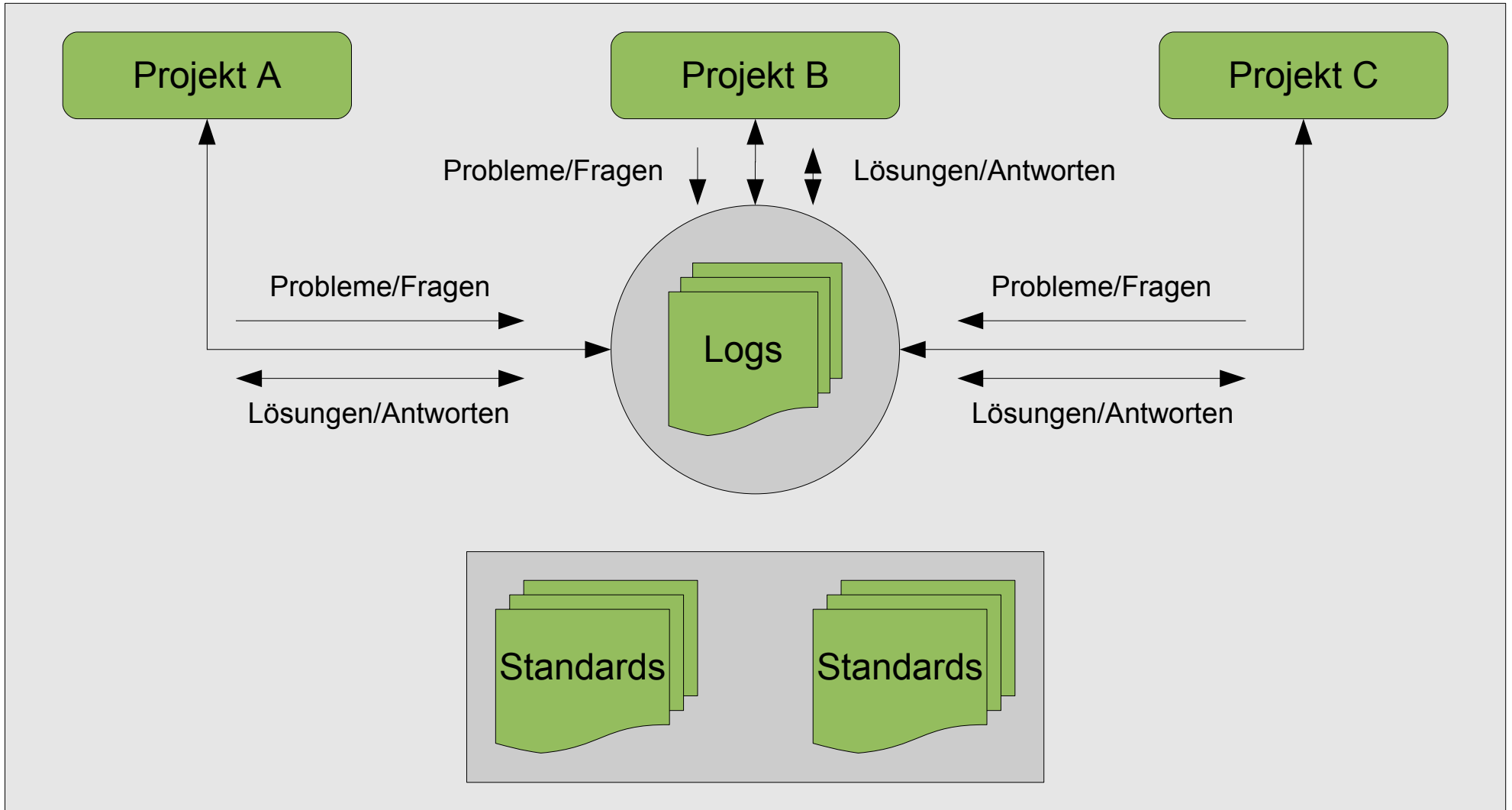


Entwickeln eines Systems zur Wissens-Verbreitung



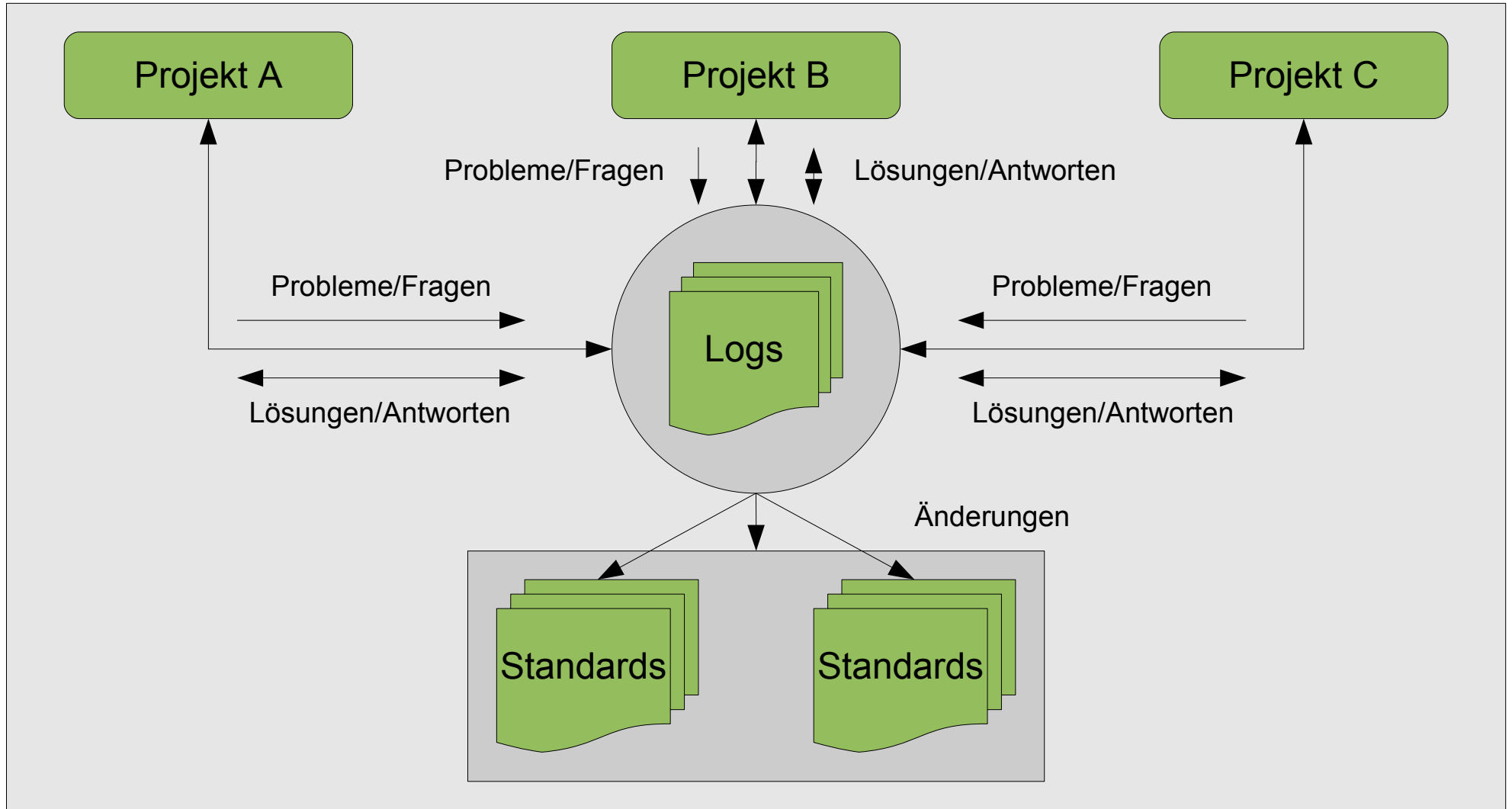


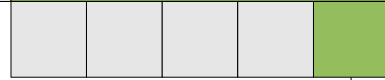
Entwickeln eines Systems zur Wissens-Verbreitung





Entwickeln eines Systems zur Wissens-Verbreitung





Entwickeln eines Systems zur
Wissens-Verbreitung

Zwei unmittelbare Vorteile dieses System:

- beschleunigte Einführung von XML-Technologien

Grund



Projekt-Teams erhalten Zugriff auf Untersuchungen,
Probleme und Lösungen anderer Teams



Entwickeln eines Systems zur
Wissens-Verbreitung

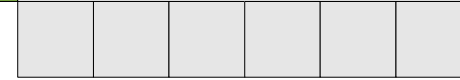
Zwei unmittelbare Vorteile dieses System:

- beschleunigte Einführung von XML-Technologien

Grund → Projekt-Teams erhalten Zugriff auf Untersuchungen,
Probleme und Lösungen anderer Teams

- Einbringen hoch-relevanter Informationen in Projekt-, Design- und
Entwicklungs-Standards.

Auswirkung → Standards bleiben aktuell



Entwickeln eines Systems zur
Wissens-Verbreitung

Ein solches System sollte frühzeitig realisiert werden!

Tipps zur Integration von

Web Services



Wissen, WANN man Web
Services verwenden sollte

Wissen, WANN man Web Services
verwenden sollte

Tipp:

Definiere das Ausmaß, in dem Web Services
verwendet werden sollen, bevor
diese entwickelt werden.

Wissen, WANN man Web Services
verwenden sollte

Einige Gründe für Web Services

- früh Verständnis erlangen
- keine neue Applikations-Architektur erforderlich
- lose Kopplung
- service-orientiertes Design wird bereits verwendet
- unterstützende Entwicklungs-Tools



Wissen, WIE man Web Services verwenden sollte

Wissen, WIE man Web Services
verwenden sollte

Tipp:

Limitiere die Reichweite der Web Services
auf die Reichweite Deines Kenntnis-Standes.
Wenn Du keine Ahnung hast, sollte nichts
service-orientiert sein, was wichtig ist.

Wissen, WIE man Web Services
verwenden sollte

Mögliche erste Projekte:

- risikoarme Prototypen
- Pilotapplikationen



Wissen, WANN man Web Services NICHT verwenden sollte

Wissen, WANN man Web Services
NICHT verwenden sollte

Tipp:

Auch wenn Web Services zum IT-Mainstream
werden:
Beziehe Web Services nur dort mit ein,
wo sie Vorteile bringen.

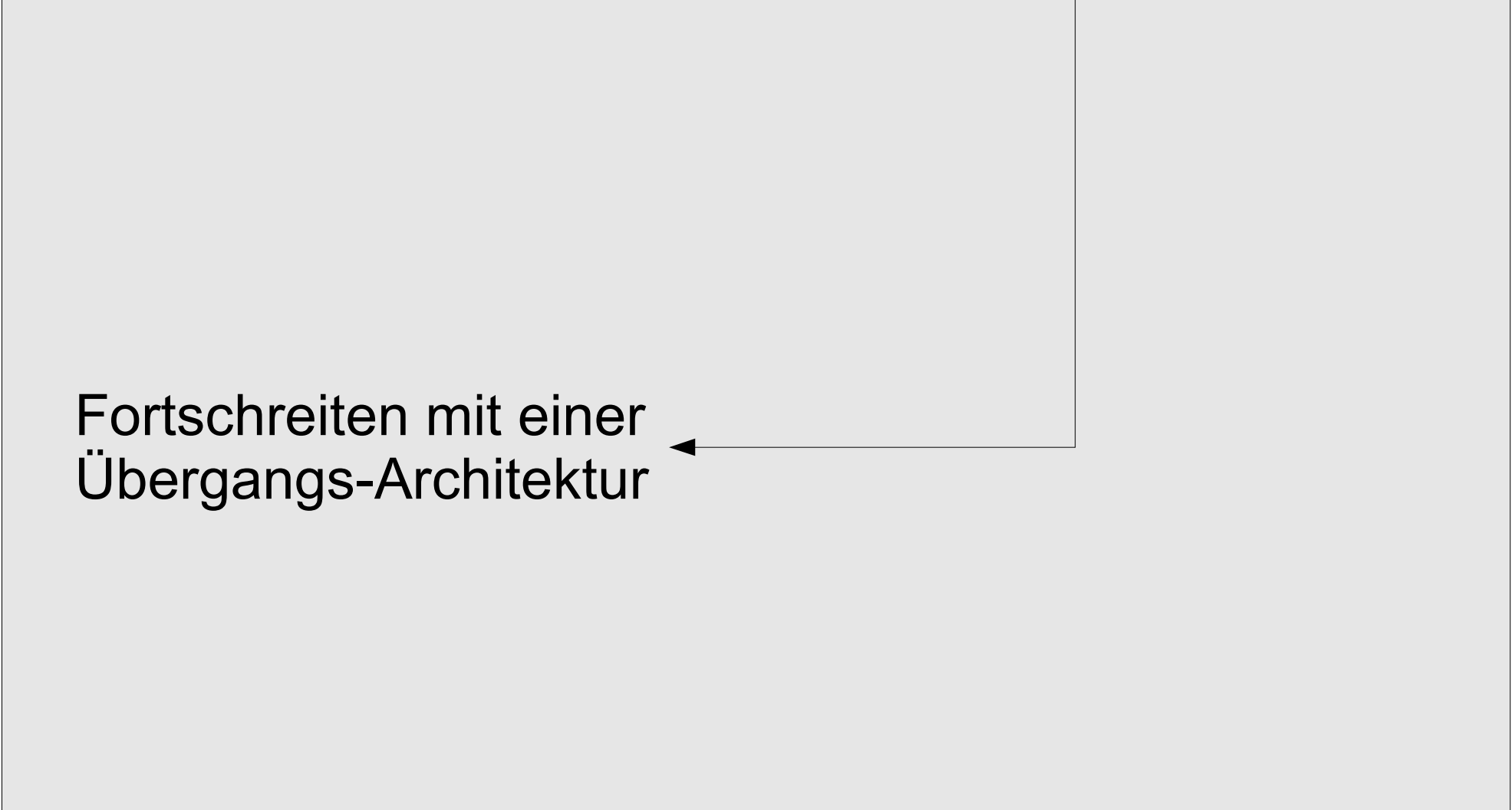
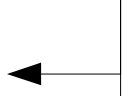
Wissen, WANN man Web Services
NICHT verwenden sollte

Einige Gründe gegen Web Services

- Anbieter-Unterstützung
- keine automatische Hilfe für Design-Optimierungen
- proprietäre Erweiterungen
- Web Services u.U. gar nicht notwendig



Fortschreiten mit einer Übergangs-Architektur



Fortschreiten mit einer Übergangs-Architektur



Fortschreiten mit einer Übergangs-
Architektur

Tipp:

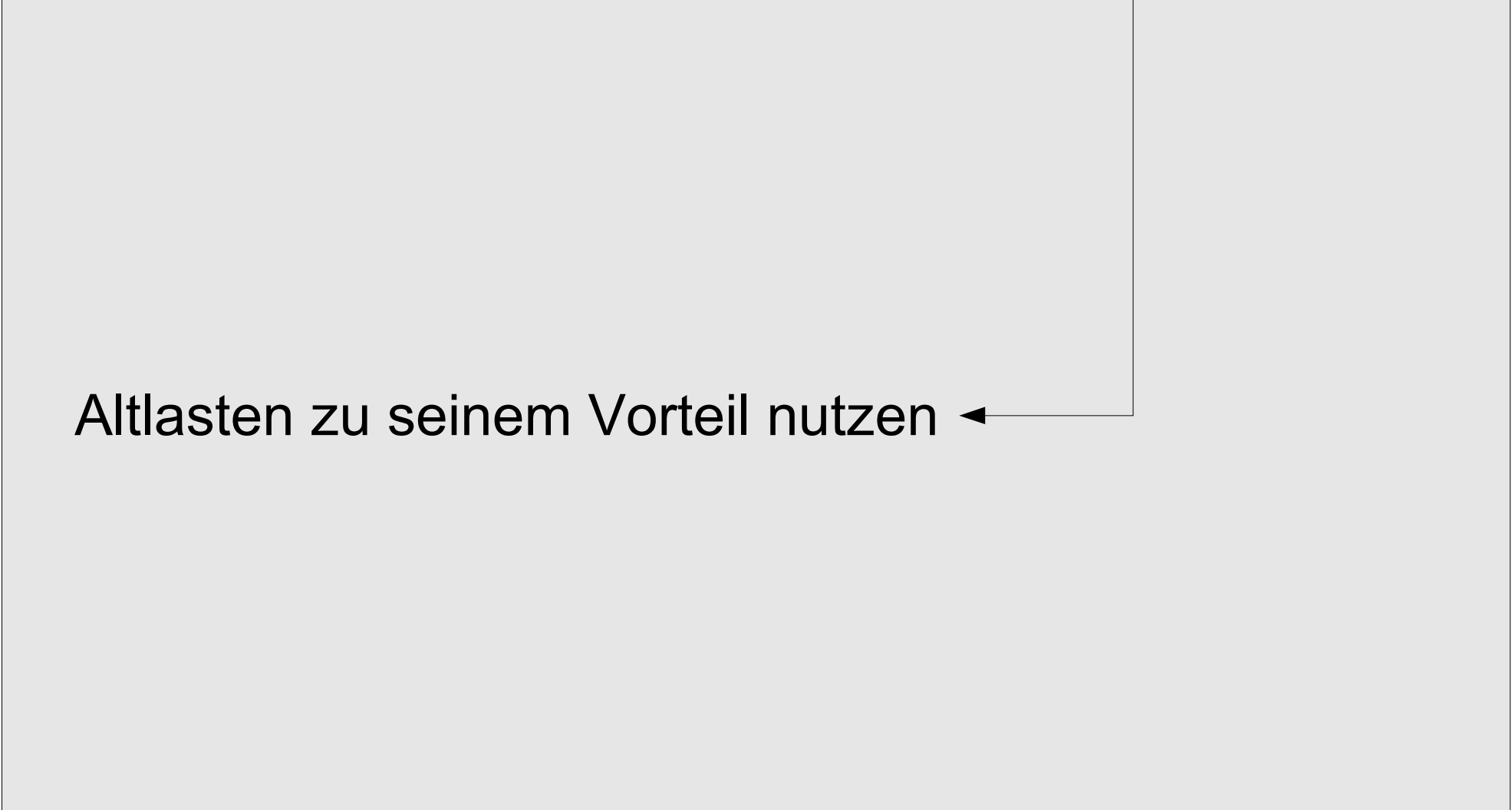
Ziehe eine Übergangs-Architektur in Betracht,
die service-orientierte Konzepte einführt,
nicht jedoch die entsprechende Technologie.



Fortschreiten mit einer Übergangs-
Architektur

Vorteile dieses Ansatzes:

- Ein Zurückkehren ist relativ problemlos möglich
- als Notfallplan einsetzbar
- Vorbereiten auf zukünftige SOA-Migration



Altlasten zu seinem Vorteil nutzen ←

Altlasten zu seinem Vorteil nutzen ←

Oft ist es sinnvoll Alt-Umgebungen zu ...

- ersetzen
- erneuern

Wichtige Alternative:

- service-orientierte Integrations-Architektur



Altlasten zu seinem Vorteil nutzen ←

Tipp:

Erwäge immer die Wiederverwendung alter Logik, bevor Du sie ersetzt.

Altlasten zu seinem Vorteil nutzen

Basis:

- Adapter
- ein "Service Interface Layer"

} funktionale
Abstraktion



Altlasten zu seinem Vorteil nutzen ←

Vorteil:

- u.U. geringerer Aufwand (Zeit, Geld, ...)

Altlasten zu seinem Vorteil nutzen ←

Tipp:

Definiere funktionale Grenzen um
Alt-Anwendungen und
integriere nicht über sie hinaus.

Altlasten zu seinem Vorteil nutzen ←

Die Integration von Alt-Logik:

- Schwierigkeiten bei ehemals isolierten Anwendungen
- Schnelle Erweiterung des System
- Probleme mit Systemen, die nicht für externe Integration geeignet sind



Altlasten zu seinem Vorteil nutzen ←

Dennoch:

Sinnvoll, solange man die Grenzen kennt.

Altlasten zu seinem Vorteil nutzen ←

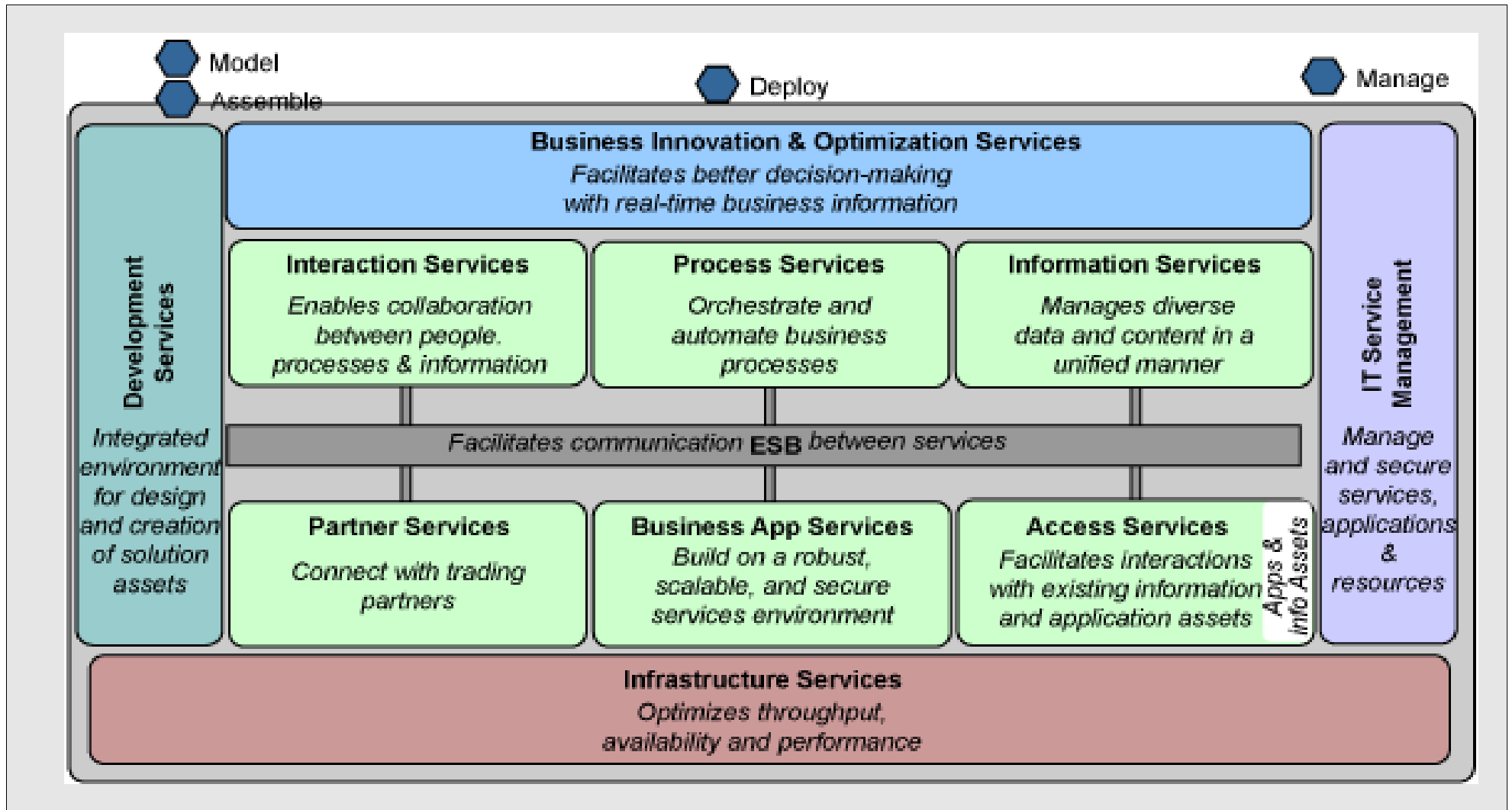
Zusätzliche Informationen:

- EAI-Lösungen (egal ob service-orientiert oder nicht) basieren auf dem selben Prinzip. (Verwendung von Adaptern):

Beispiel:

- WebSphere (IBM): WebSphere Business Integration Adapters

Altlasten zu seinem Vorteil nutzen



Altlasten zu seinem Vorteil nutzen ←

Zusätzliche Informationen (Fortsetzung):

- Intelligente Adapter verringern die Auswirkungen der Integration von alter Logik.



Um ein vernünftiges Sicherheits-
Modell herumbauen

Um ein vernünftiges Sicherheits-Modell
herumbauen



Tipp:

Das Anwendungs-Design muss auf einem
Sicherheits-Modell aufbauen,
nicht andersrum.

Um ein vernünftiges Sicherheits-Modell
herumbauen



Grund:

Sicherheits-Systeme sind:

- einzigartig
- komplex
- multi-dimensional

Es gibt also viele Faktoren, die andere Teile des
Anwendungs-Designs einschränken

Um ein vernünftiges Sicherheits-Modell
herumbauen



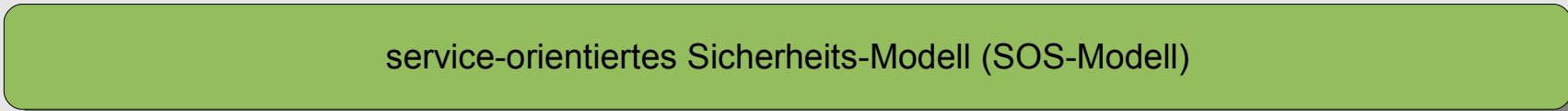
Tipp:

Ein wichtiger Teil eines standardisierten
service-orientierten Systems ist ein
service-orientiertes Sicherheits-Modell.

Um ein vernünftiges Sicherheits-Modell
herumbauen



SOS-Modell als Basis

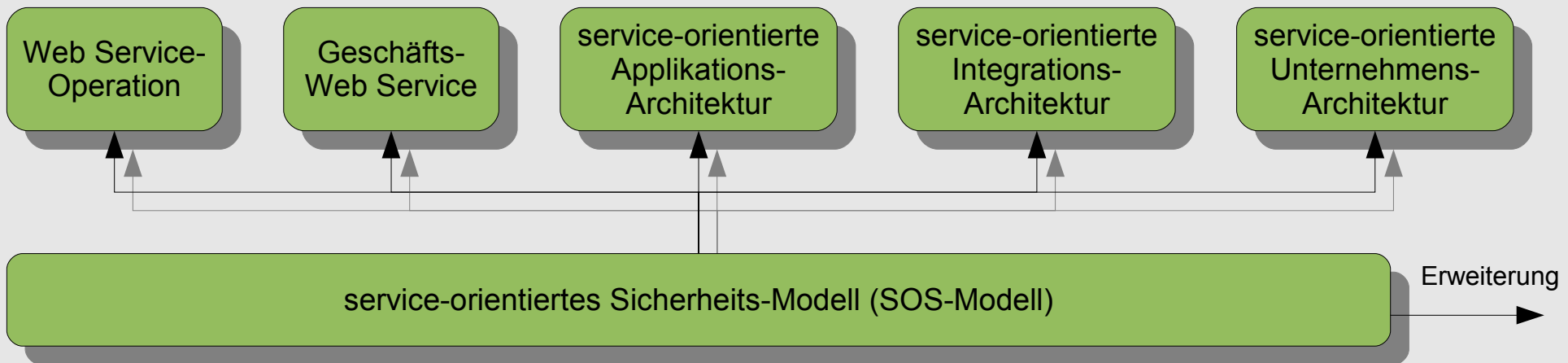


Erweiterung →

Um ein vernünftiges Sicherheits-Modell herumbauen



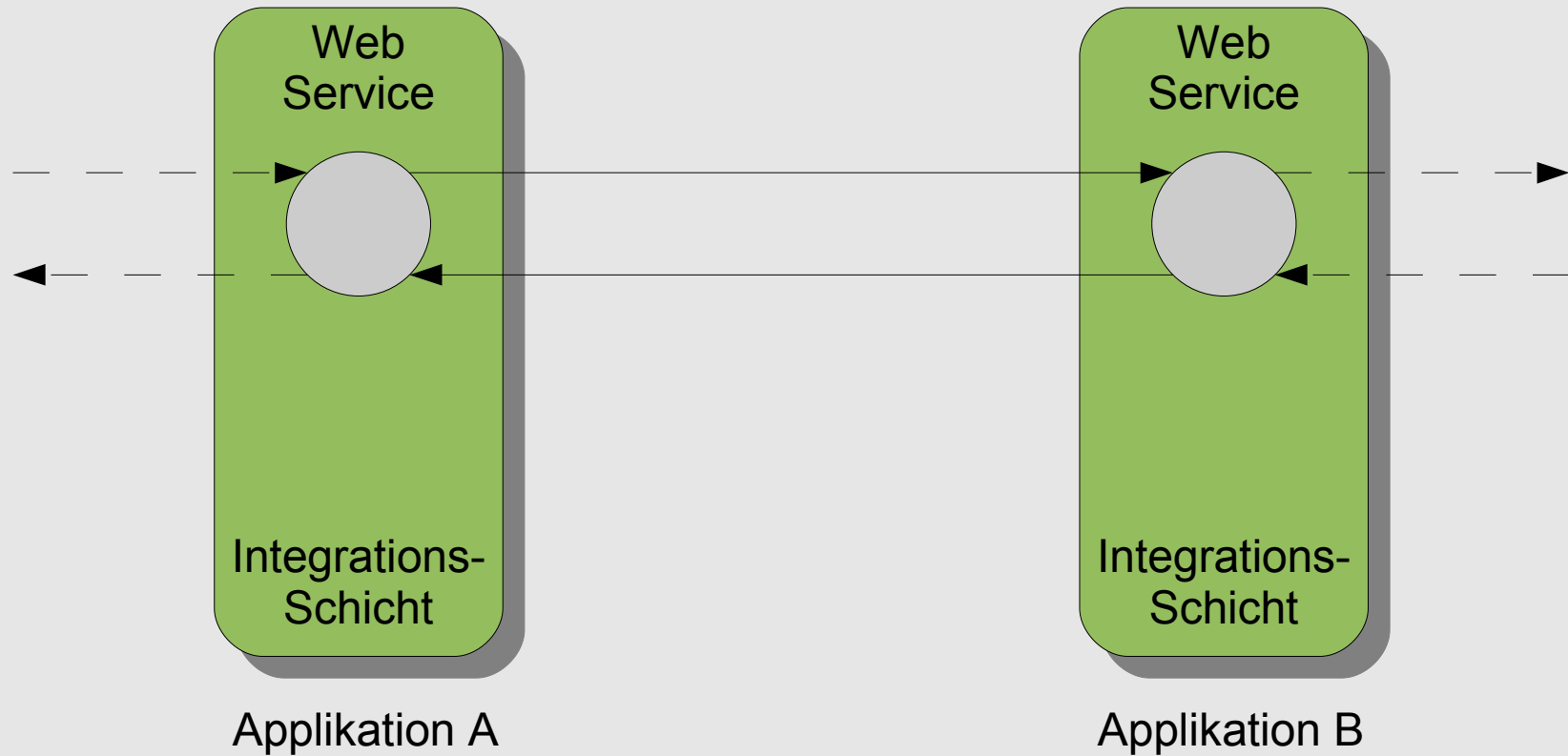
SOS-Modell zum Aufbau einer Standard-Sicherheits-Plattform



Um ein vernünftiges Sicherheits-Modell
herumbauen



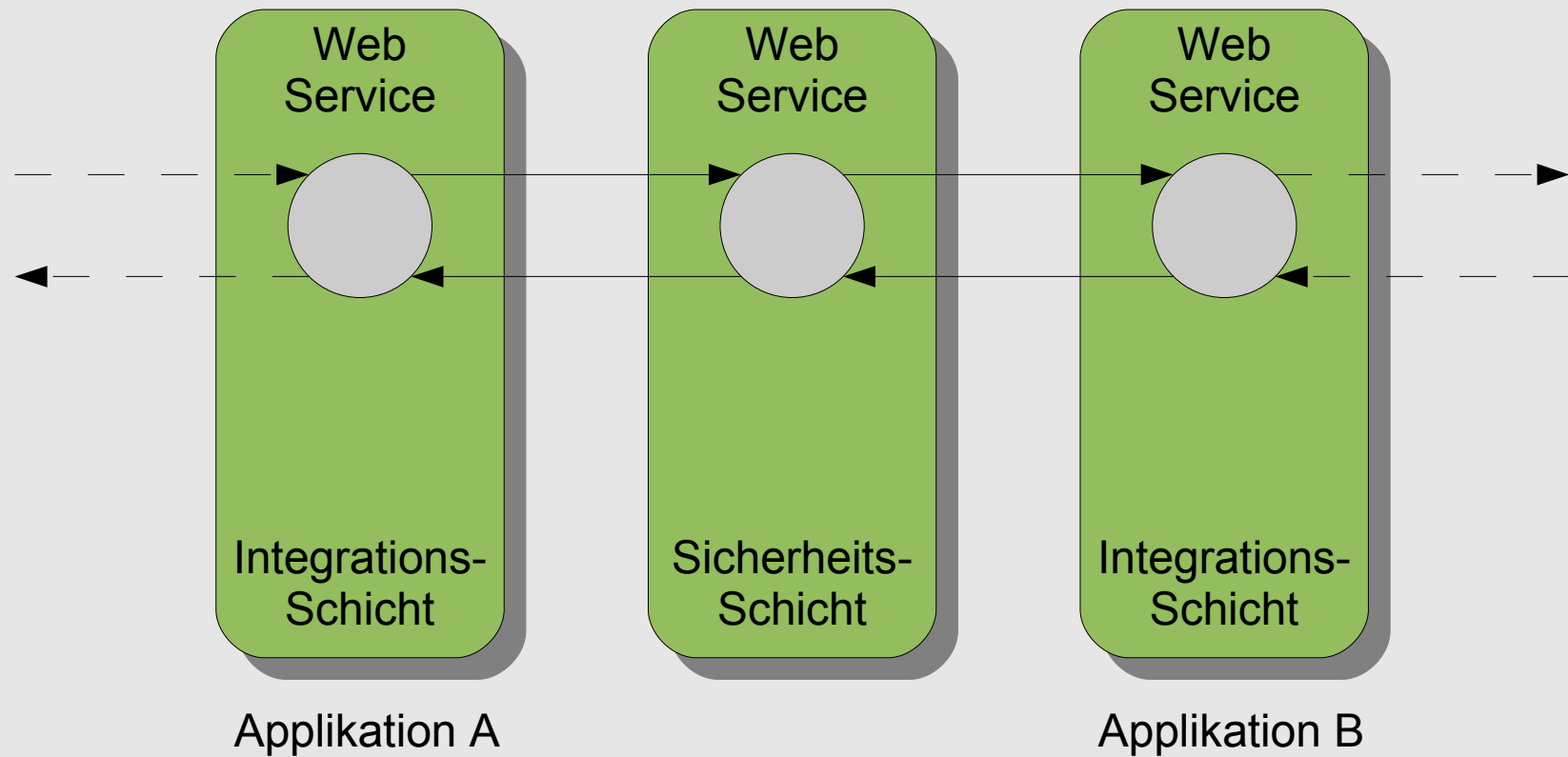
Situation ohne Sicherheits-Modell



Um ein vernünftiges Sicherheits-Modell
herumbauen



Implementation des SOS-Modells als eigenständige Sicherheits-Schicht



Um ein vernünftiges Sicherheits-Modell
herumbauen



Vorteil:

- Anwendungen werden vom Umgang mit Belangen bzgl. der Sicherheits (größtenteils) befreit.

Um ein vernünftiges Sicherheits-Modell
herumbauen



Nachteil:

- Das Sicherheits-Modell kann zu bedeutenden Einschränkungen des Anwendungs-Designs führen

Tipp:

Teste für das Unbekannte.



A large, light gray rectangular area that serves as a background for the main content. It contains a green graphic consisting of two overlapping rectangles. The larger, top-right rectangle is a medium green and contains the word 'ENDE' in large, black, sans-serif capital letters. The smaller, bottom-left rectangle is a darker green and overlaps the bottom-left corner of the larger one. Below this graphic, the text 'Vielen Dank für die Aufmerksamkeit' is centered in a black, sans-serif font.