
Vorlesung „Methoden des Software Engineering“

Block B „Software-Architektur“ **Methodischer Entwurf**

Martin Wirsing

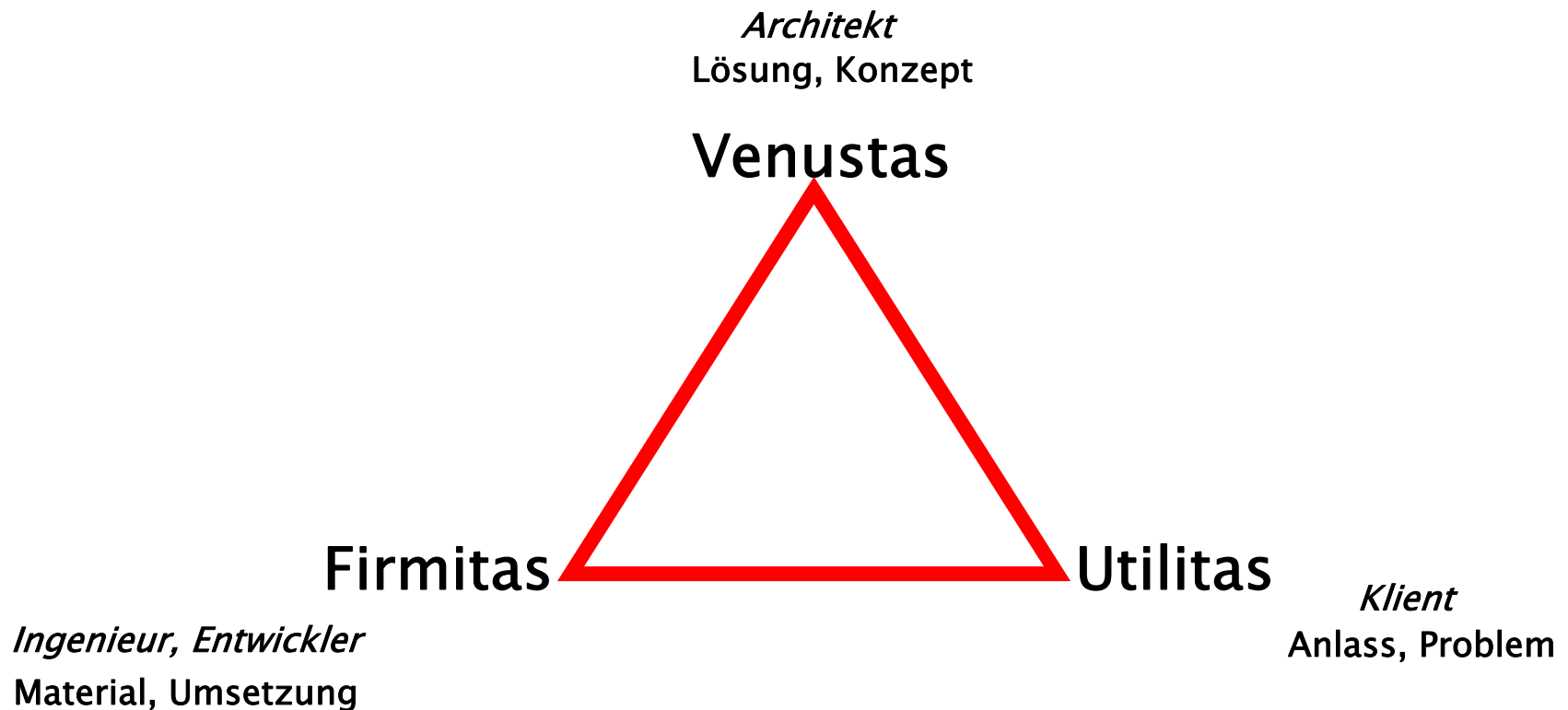
Einheit B.1, 07.11.2006

Gliederung B.1

- **Einleitung in den Block B „Software-Architektur“**
 - Definition, Gegenstand, Motivation
 - Gliederung und Einbettung in die Vorlesung
- **Teil 1: Methodischer Entwurf**
 - Notationen & Konzepte
 - Fallbeispiel
 - Hilfsmittel & Literatur
- **Ausblick auf Teil 2: Systemarchitektur**

Software-Architektur – Gegenstand

Die klassische Trias der Architektur seit Vitruv.



Software-Architektur – Definitionen

Architecture is the set of design decisions [...] that keeps its implementers and maintainers from exercising needless creativity.

(D`Souza & Wills, Catalysis)

Architecture is a framework for change.

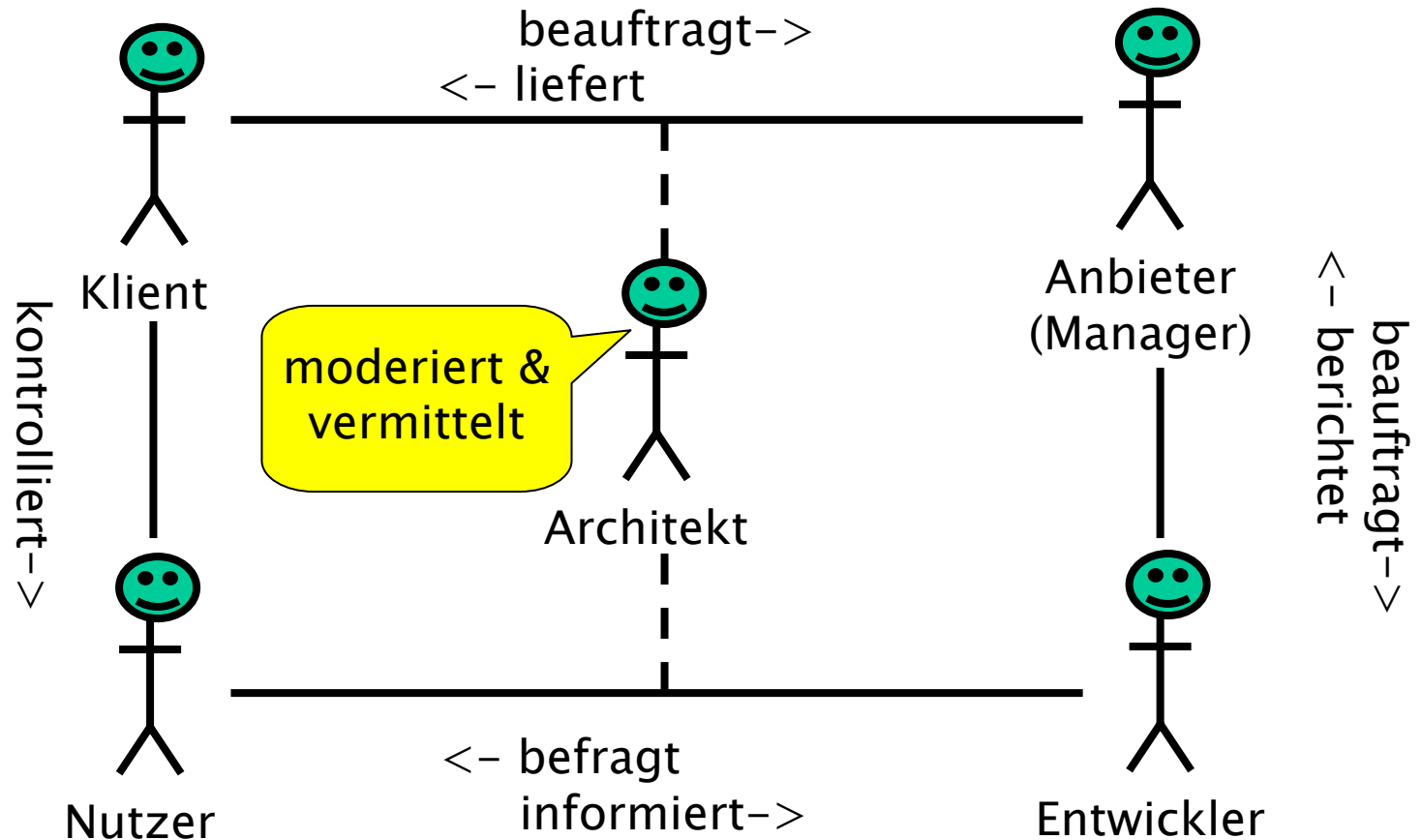
(Tom De Marco)

Architecture is defined [...] as the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution.

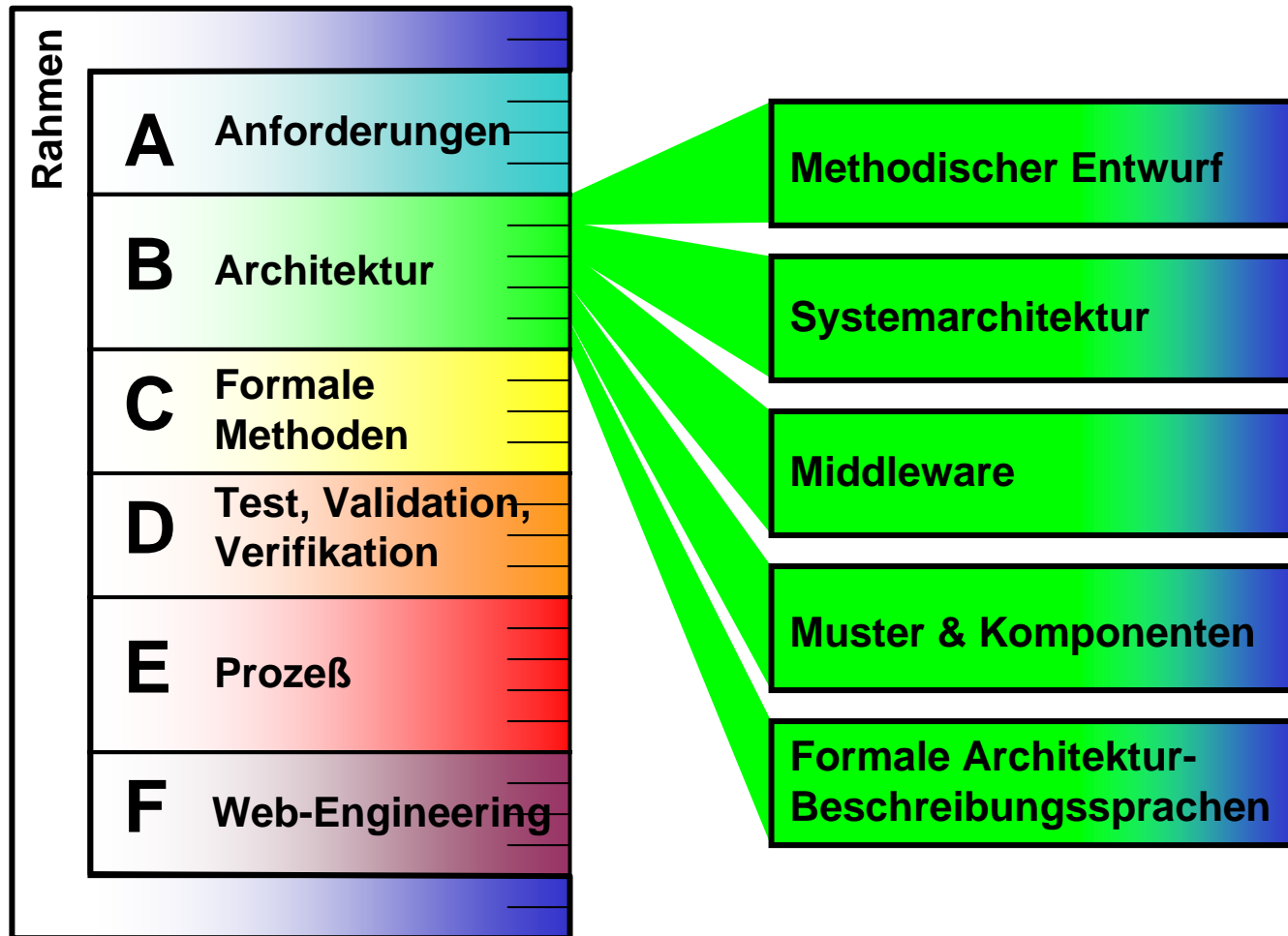
(IEEE Architecture Working Group, P1471)

Siehe z. B. www.sei.cmu.edu/architecture/definitions.html für weitere Definitionen.

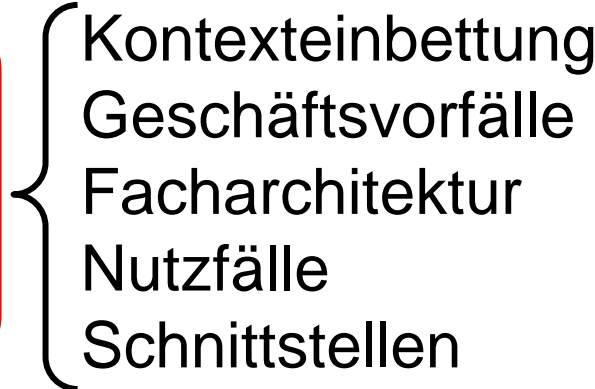
Die Rolle des Software-Architekten



Gliederung Block B



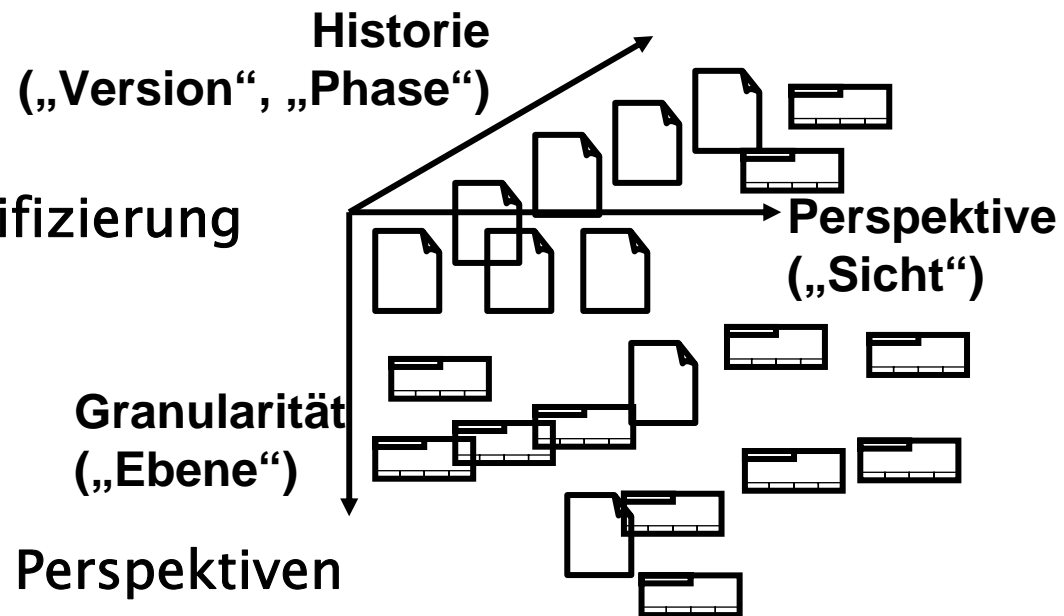
Gliederung B.1

- **Einleitung in den Block B „Software-Architektur“**
 - Definition, Gegenstand, Motivation
 - Gliederung und Einbettung in die Vorlesung
 - **Teil 1: Methodischer Entwurf**
 - Notationen & Konzepte
 - Fallbeispiel
 - Hilfsmittel & Literatur
 - **Ausblick auf Teil 2: Systemarchitektur**
- 
- Kontexteinbettung
 - Geschäftsvorfälle
 - Facharchitektur
 - Nutzfälle
 - Schnittstellen

Methodik Dokumentenbasierte Entwicklung

- Ein System entsteht und entwickelt sich als eine Menge von Dokumenten: Konzepte, Präsentationen, Testfälle, Code, ...

- Drei Dimensionen der Klassifizierung



- Verschiedene überlappende Perspektiven

- Überlappungen sind Quellen von Inkonsistenzen, aber auch Möglichkeit der Konsistenzprüfung („Sollbruchstellen“).
- Überlappungen bieten außerdem Möglichkeiten, von einer Sicht eine andere methodisch abzuleiten („opportunistisches Vorgehen“).

Betrachtete Aspekte und Darstellungsmittel

Kontext

Einbettung

Quantifizierung

Fachliches Klassenmodell

Zusammenhänge

Objektlebenszyklus

Klassendetails

Struktur

Geschäftsvorfall (GeVo)

hierarchischer Text

GeVo-Diagramm

GeVo-Tabelle

Aktivitätsdiagramm

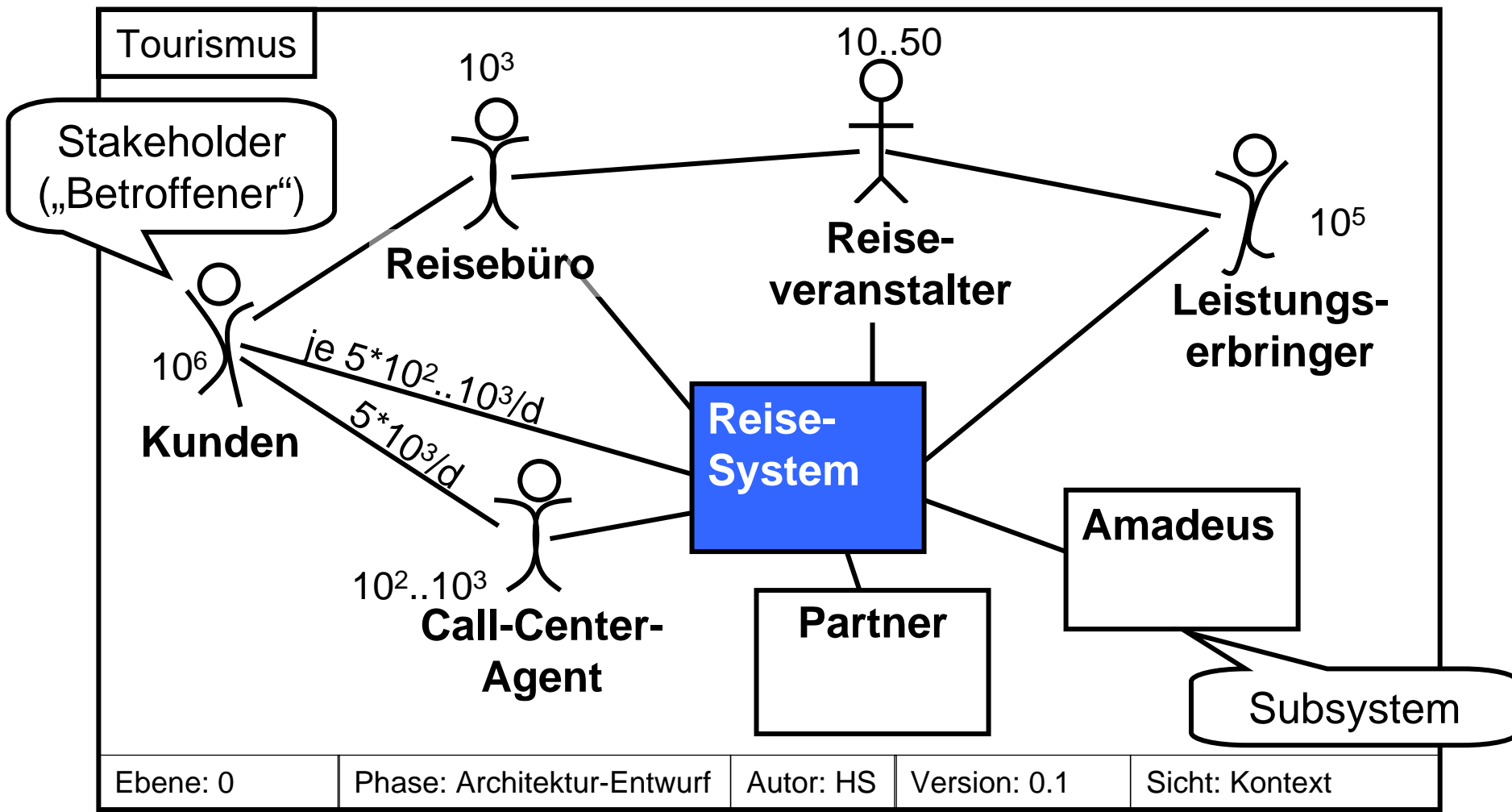
Facharchitektur

Subsysteme & Schnittstellen

(fachliche) Interaktionen

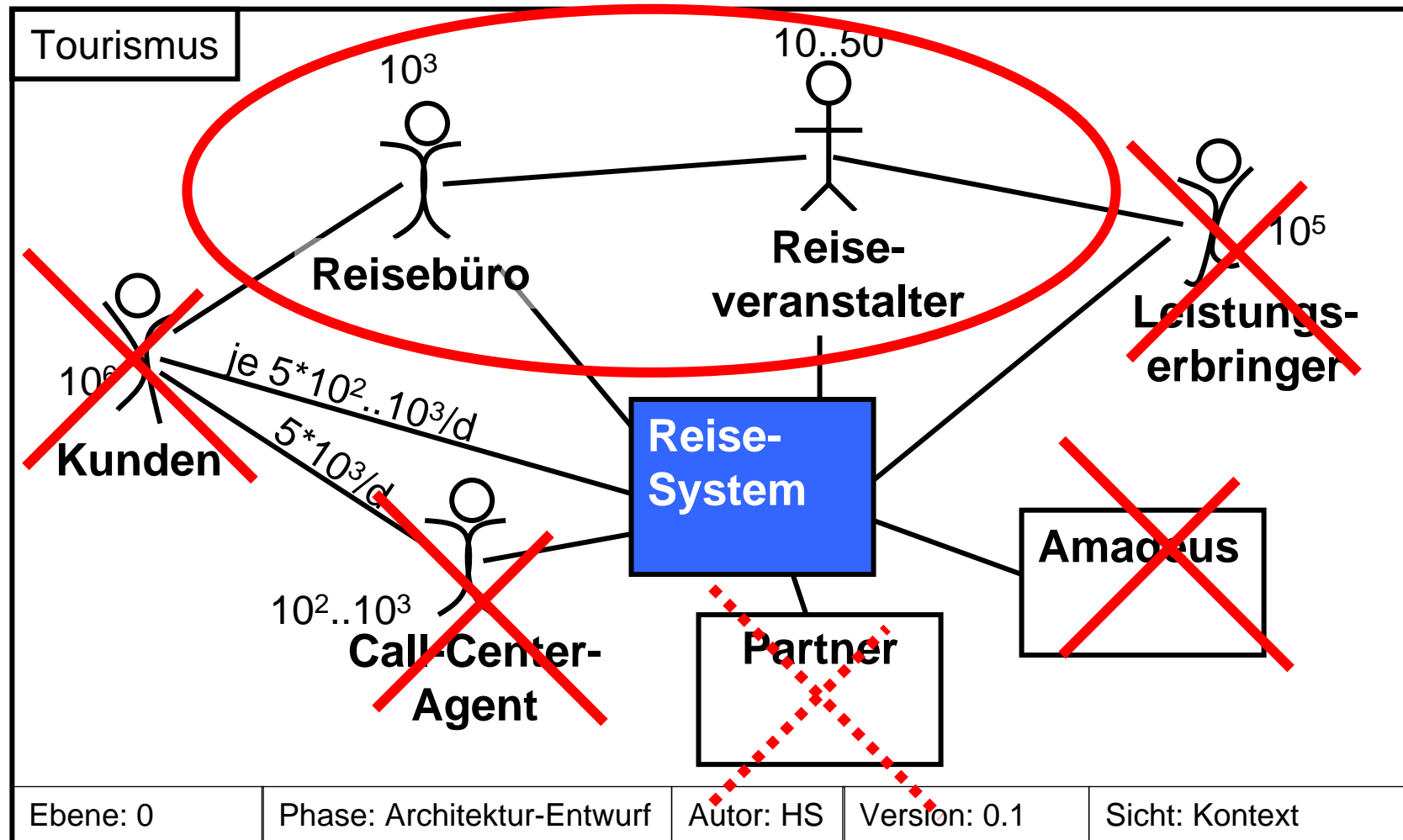
Ebene 0

Kontexteinbettung



Ebene 0

Kontexteinbettung



Betrachtete Aspekte und Darstellungsmittel

Kontext

Einbettung

Quantifizierung

Fachliches Klassenmodell

Zusammenhänge

Objektlebenszyklus

Klassendetails

Struktur

Facharchitektur

Subsysteme & Schnittstellen

(fachliche) Interaktionen

Geschäftsvorfall (GeVo)

hierarchischer Text

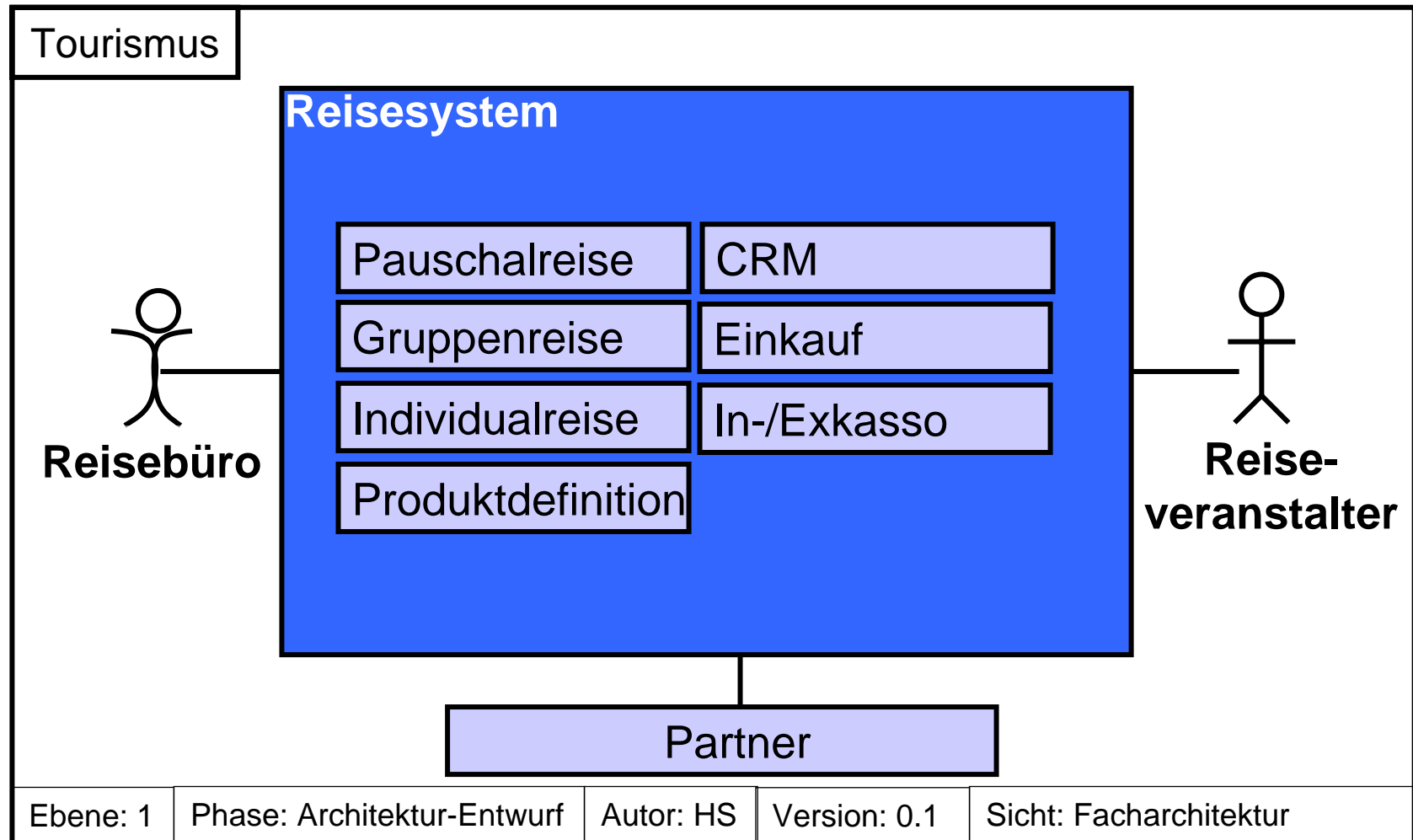
GeVo-Diagramm

GeVo-Tabelle

Aktivitätsdiagramm

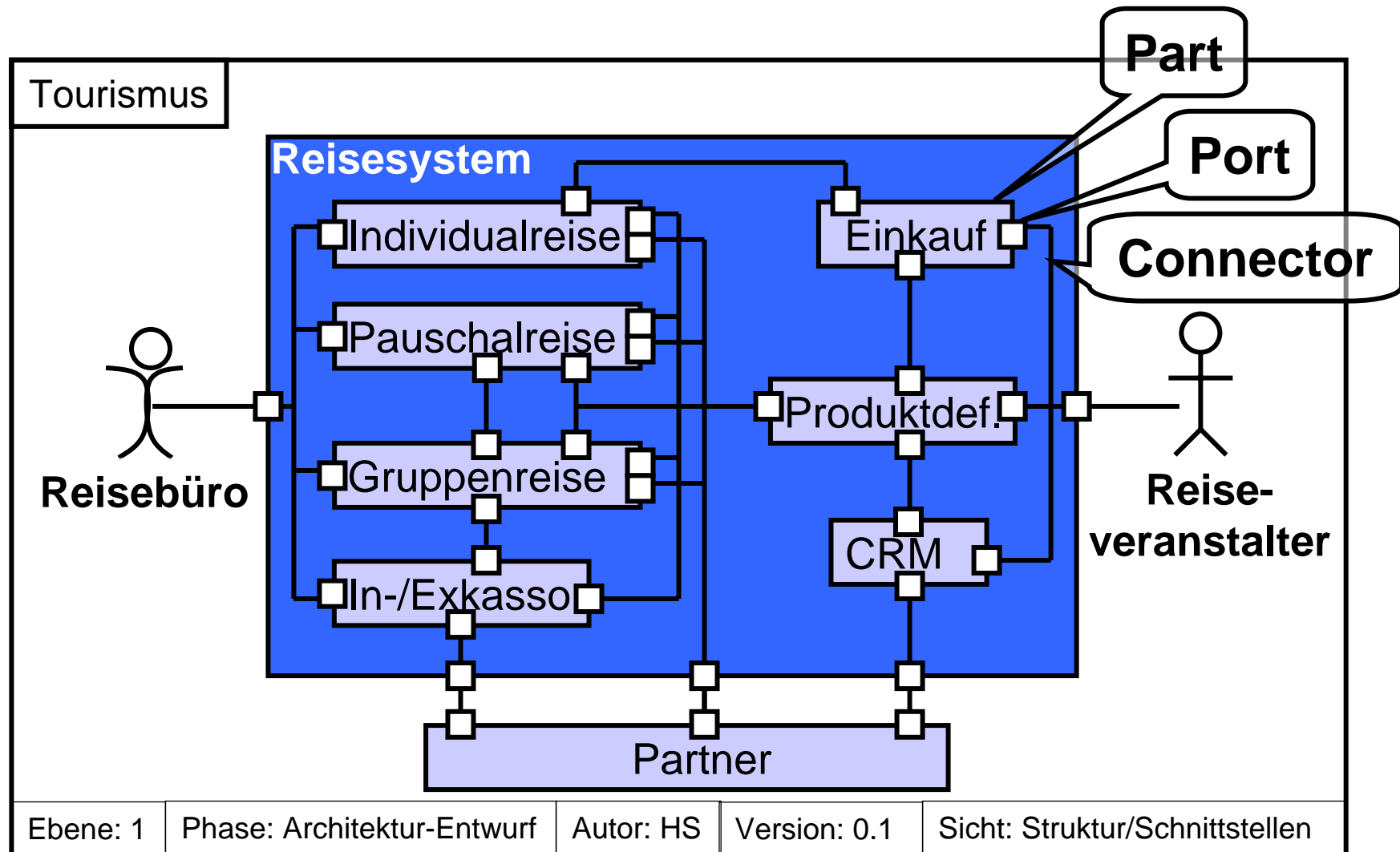
Ebene 1

Facharchitektur



Ebene 1

Facharchitektur



Spezifikation von Ports mit Protokollrollen

- **Im Allgemeinfall:**
 - je eine Liste für eingehende und ausgehende Signale plus ein gemeinsamer Zustandsautomat.
- **Im Modell:**
 - je ein (UML-)Interface pro Richtung (können optional einen Zustandsautomaten haben).
- **Für Benutzerschnittstellen:**
 - Window-Event-Diagramm, Storyboards, GUI-Prototypen.
- **Zur Spezifikation sind auch XML-DTDs, CORBA-IDLs, oder DB-Schemata (z. B. als DDL-Ausdrücke oder ER-Modelle) nützlich.**

Betrachtete Aspekte und Darstellungsmittel

Kontext

Einbettung

Quantifizierung

Fachliches Klassenmodell

Zusammenhänge

Objektlebenszyklus

Klassendetails

Struktur

Facharchitektur

Subsysteme & Schnittstellen

(fachliche) Interaktionen

GeVo

hierarchischer Text

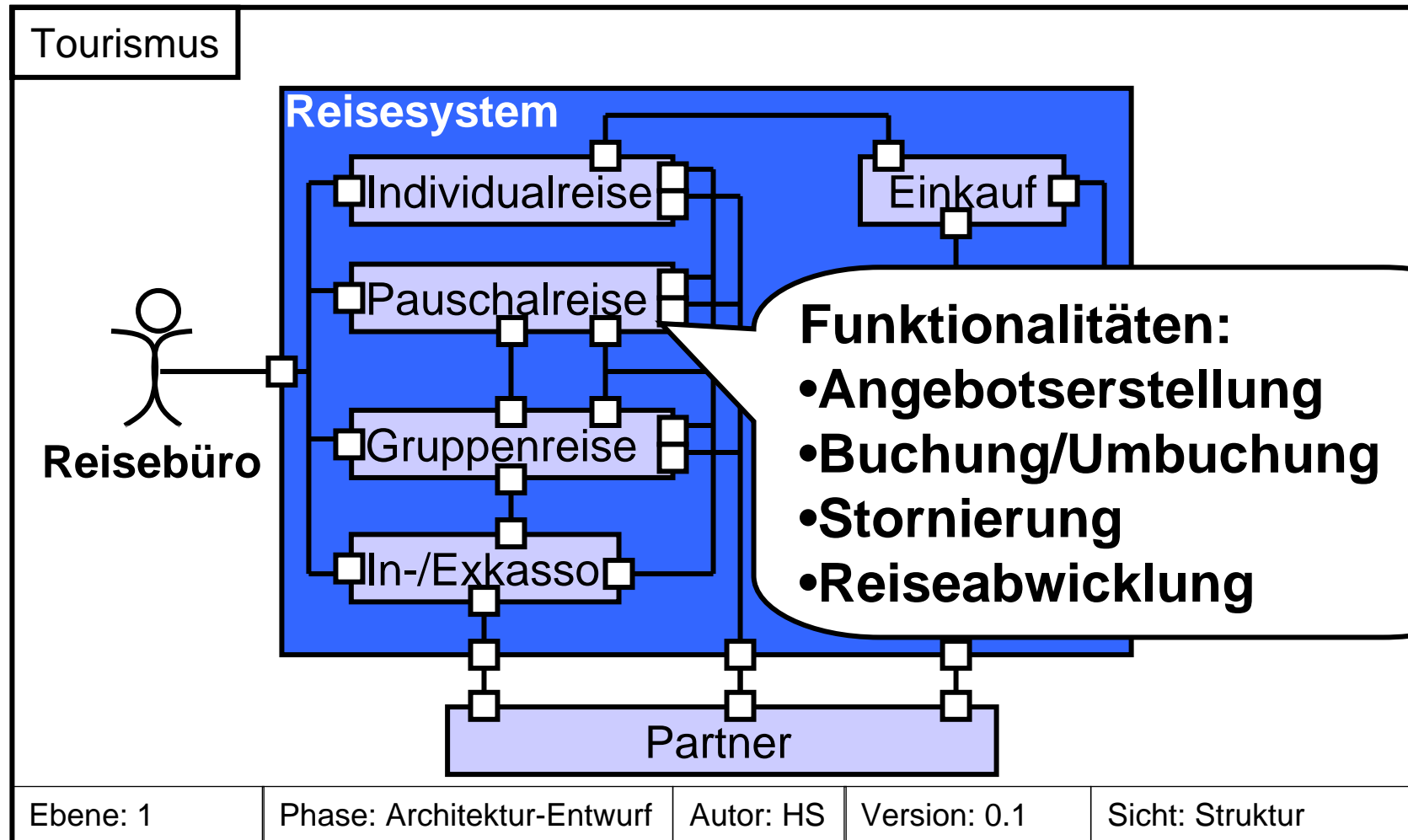
GeVo-Diagramm

GeVo-Tabelle

Aktivitätsdiagramm

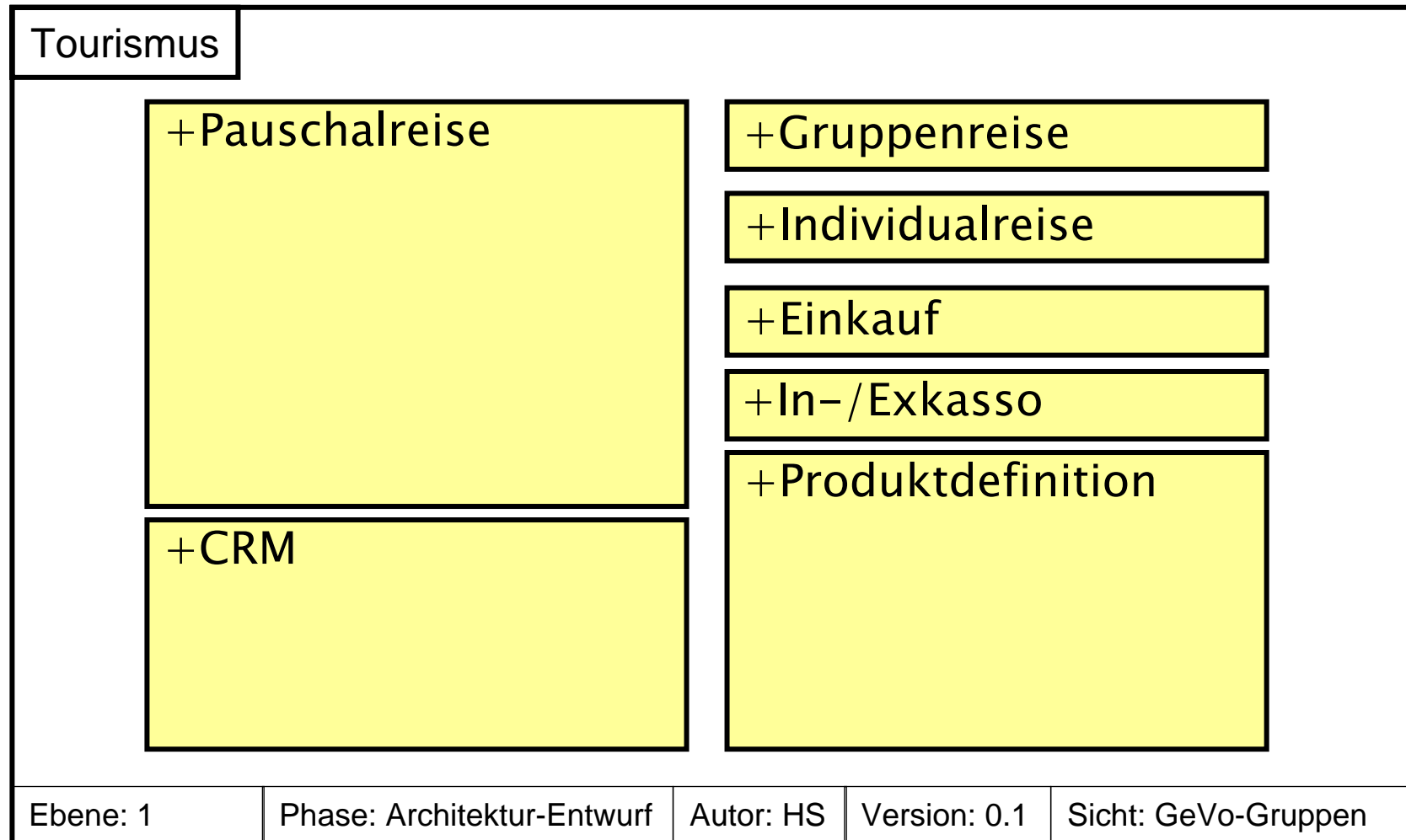
Ebene 1

Facharchitektur



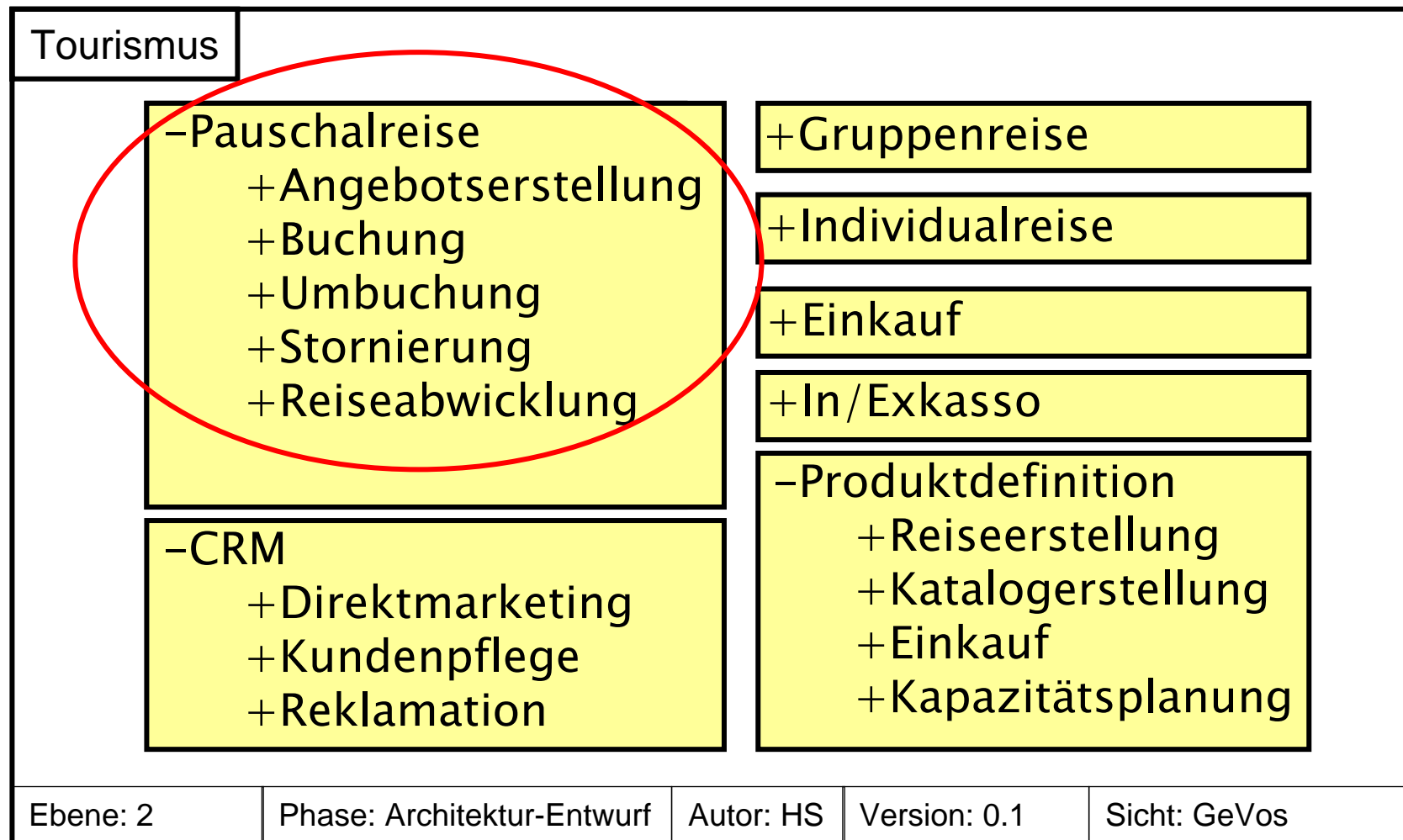
Ebene 1

Facharchitektur



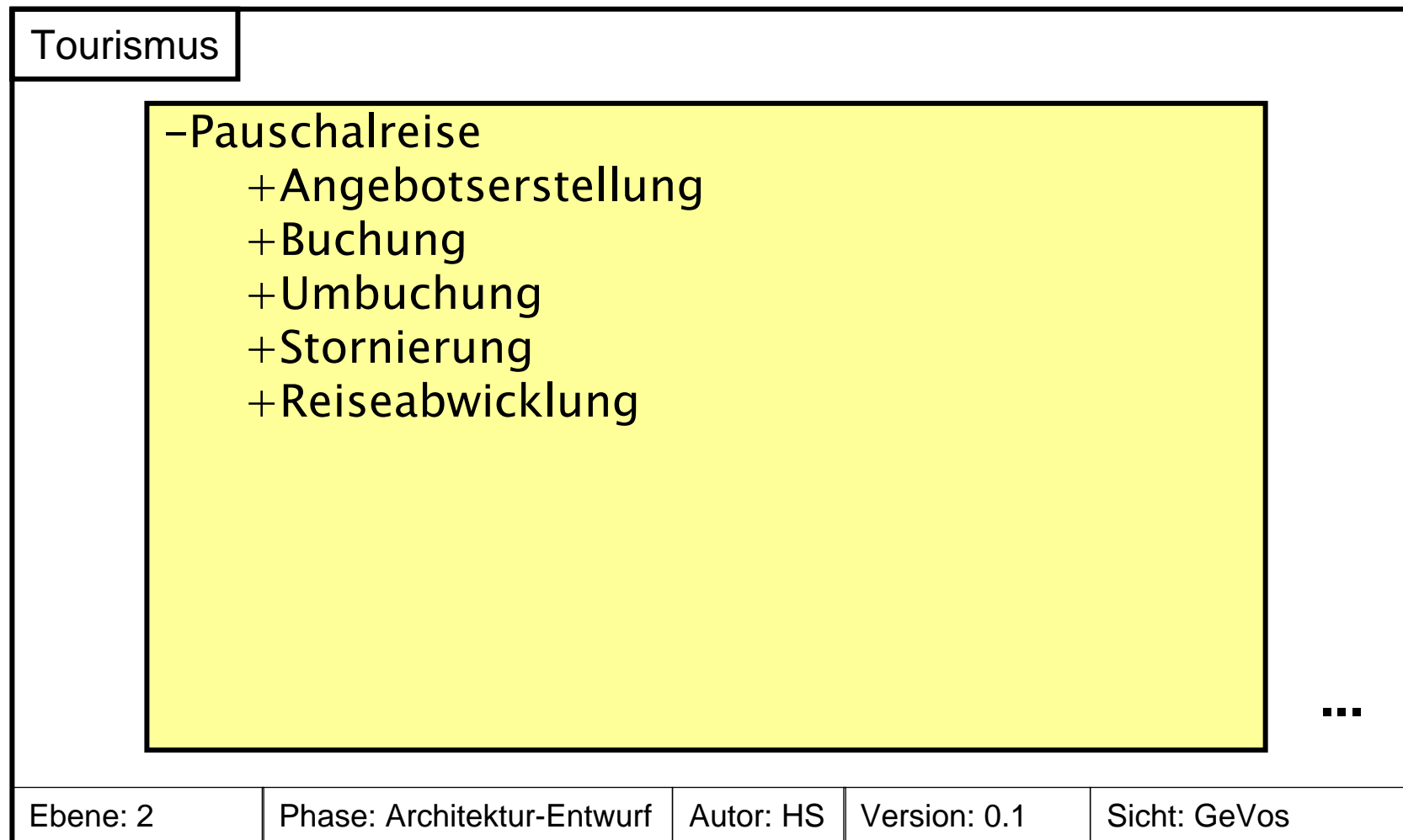
Ebene 2

Geschäftsvorfälle in der Facharchitektur



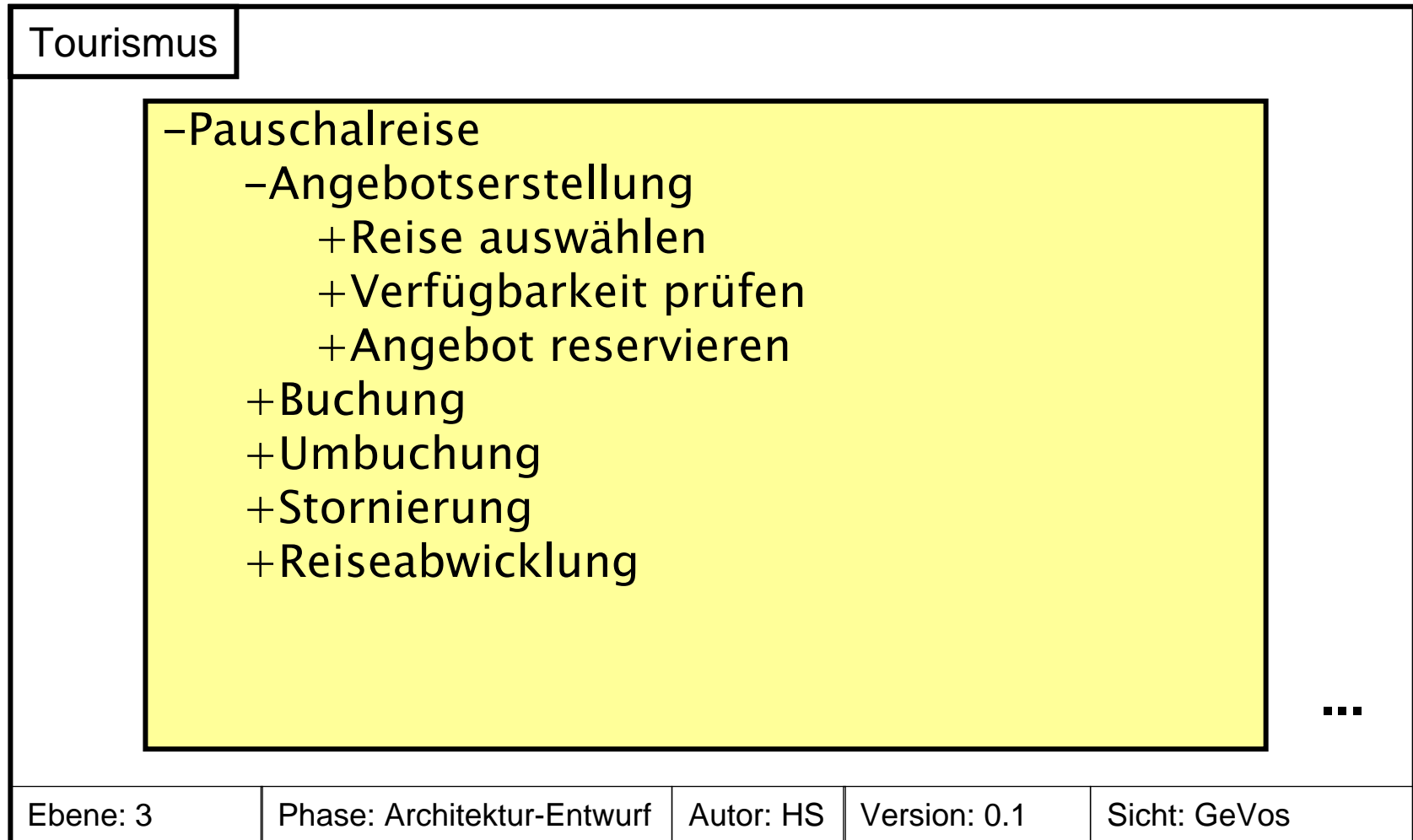
Ebene 2

Geschäftsvorfälle in der Facharchitektur



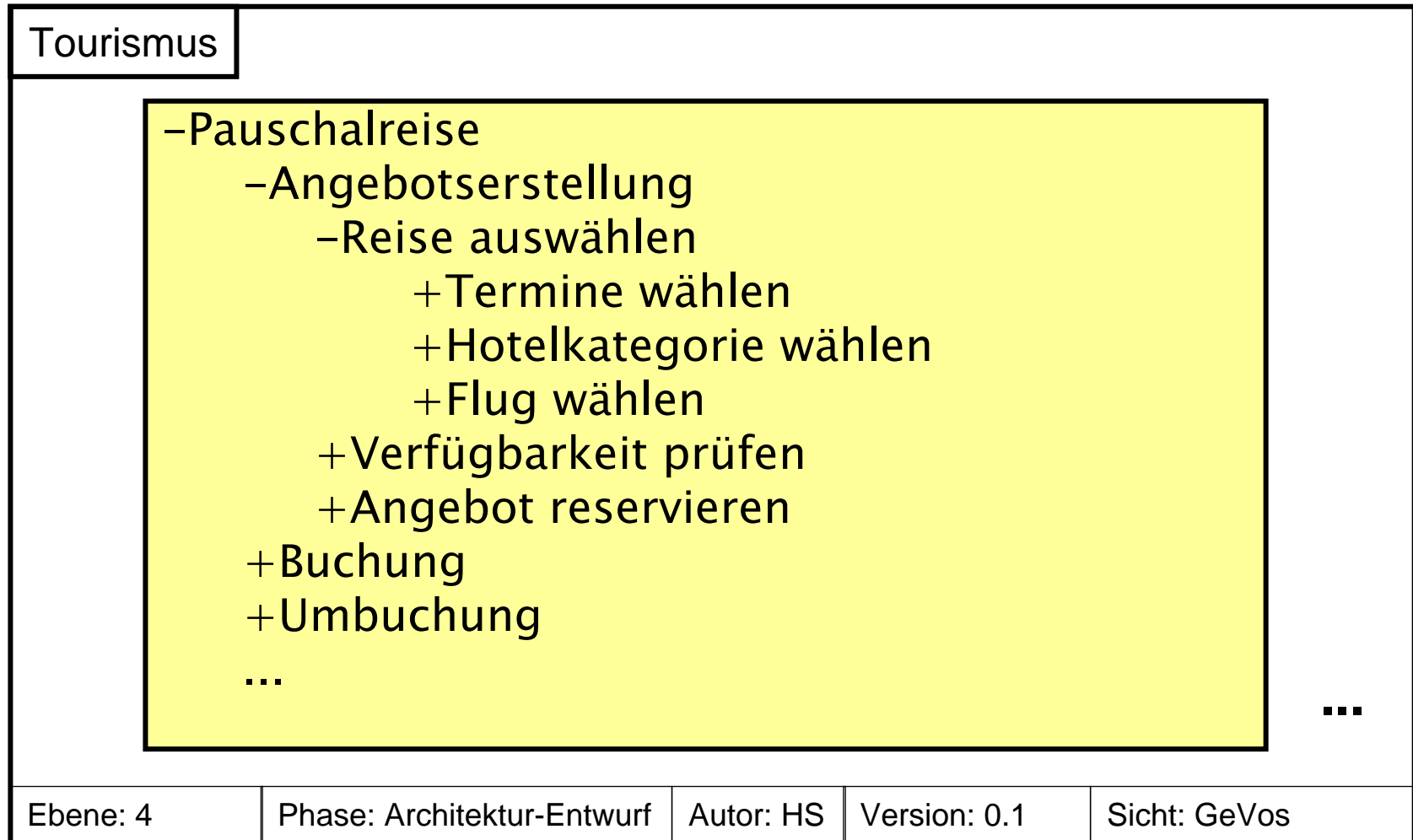
Ebene 3

Geschäftsvorfälle/Nutzfälle in der Facharchitektur



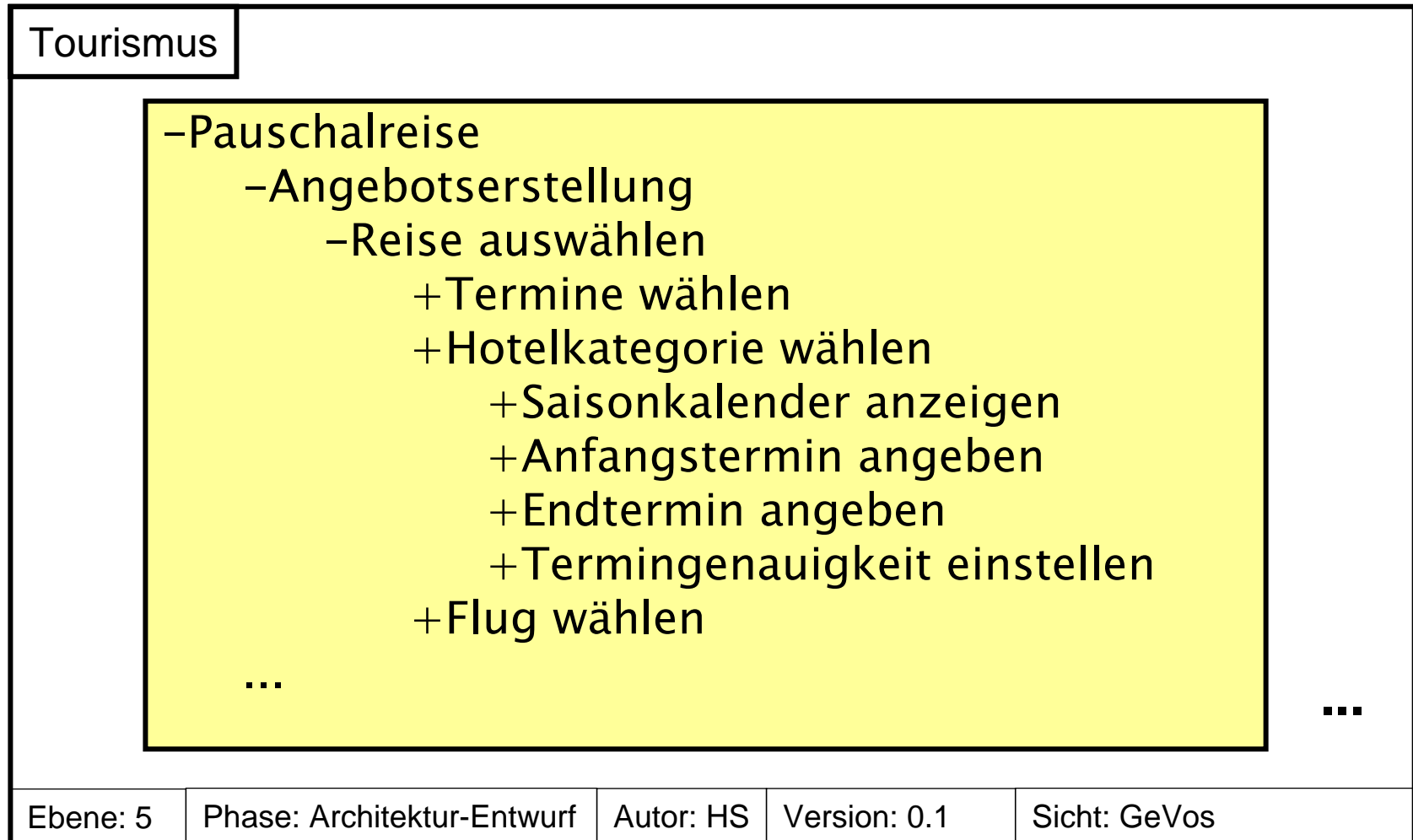
Ebene 4

Geschäftsvorfälle/Nutzfälle in der Facharchitektur



Ebene 5

Geschäftsvorfälle/Nutzfälle in der Facharchitektur



Unterschied Geschäftsvorfall/Nutzfall

Geschäftsvorfall

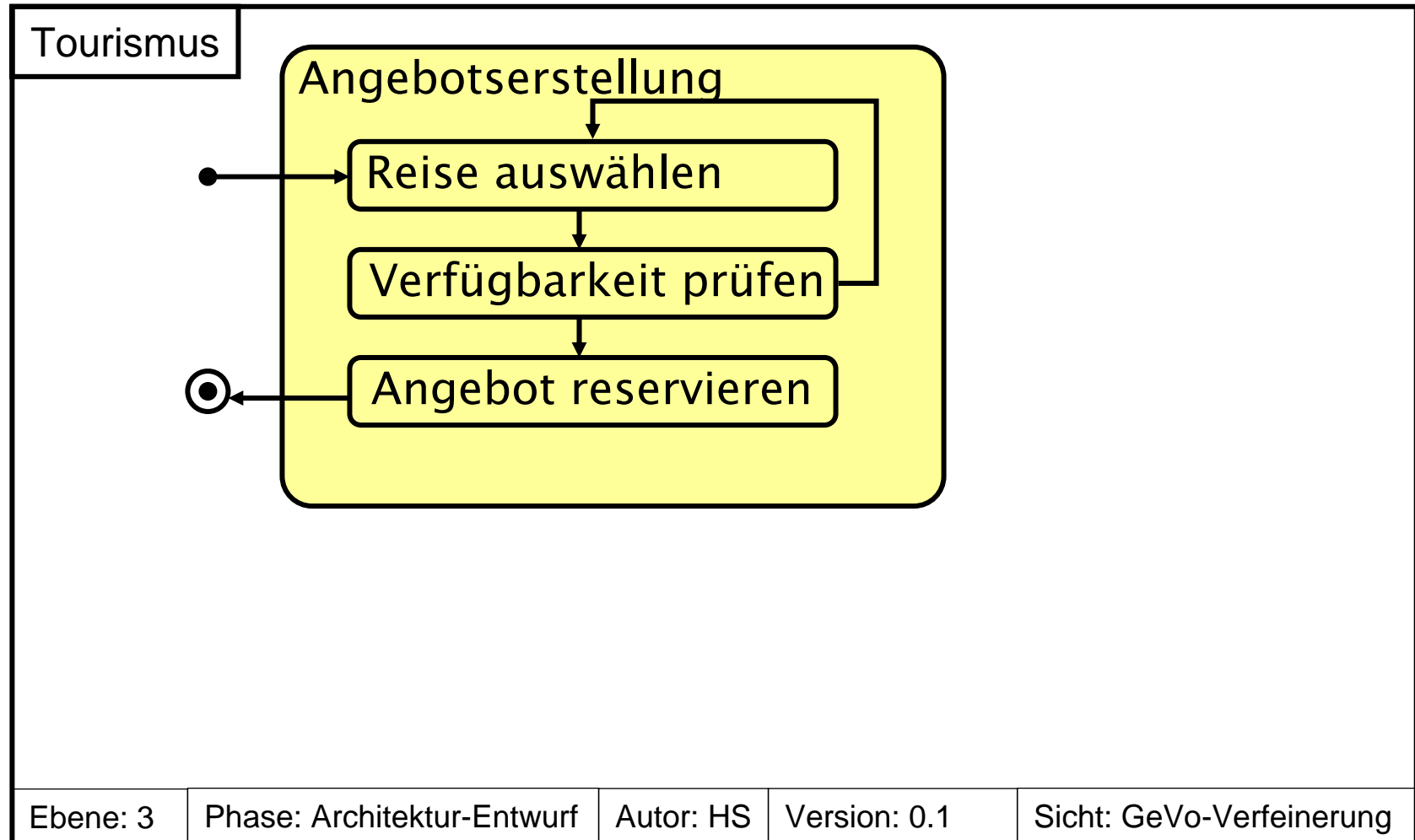
- überschreitet Systemgrenzen
- benutzt/enthält Nutzfälle
- möglicherweise nur teilweise automatisch
- kann (leicht) unterbrochen werden
- wird erbracht von einer Organisation, umfasst mehrere Systeme & Personen
- von messbarem Wert oder Kosten, für einen Aktor (von außen sichtbar)
- white box view

Nutzfall

- definiert Systemgrenze
- kann in GeVos vorkommen
- automatisch: umfasst keine manuellen Zwischenschritte
- läuft atomar in einem Zeitintervall ab
- wird erbracht von einem System
- black box view

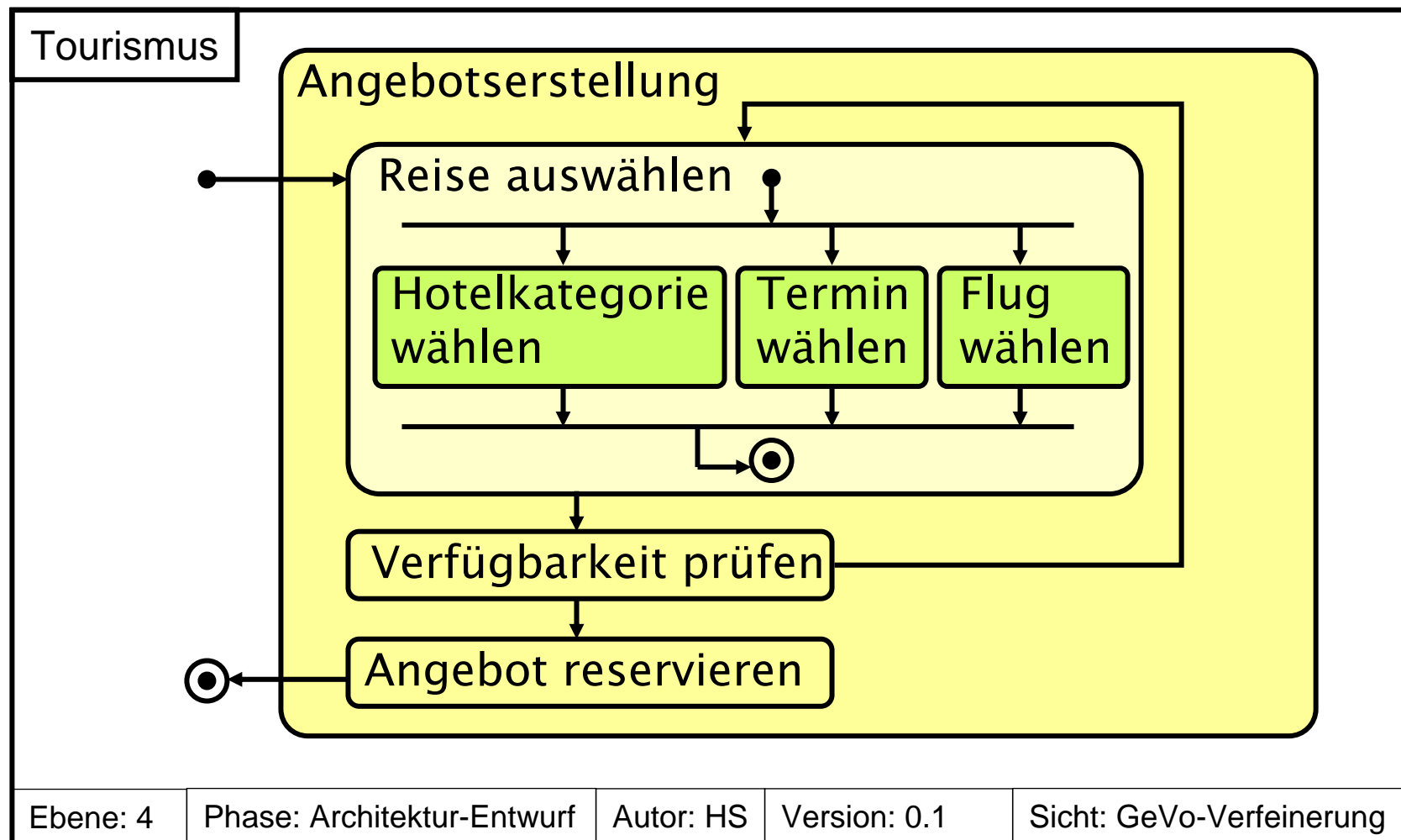
Ebene 3

Geschäftsvorfälle/Nutzfälle in der Facharchitektur



Ebene 3..4

Geschäftsvorfälle/Nutzfälle in der Facharchitektur



Ebene 3..4

Alternative Szenarios für Geschäftsvorfälle

Der Normalfall (Primärszenario):
der häufigste oder wichtigste Fall.

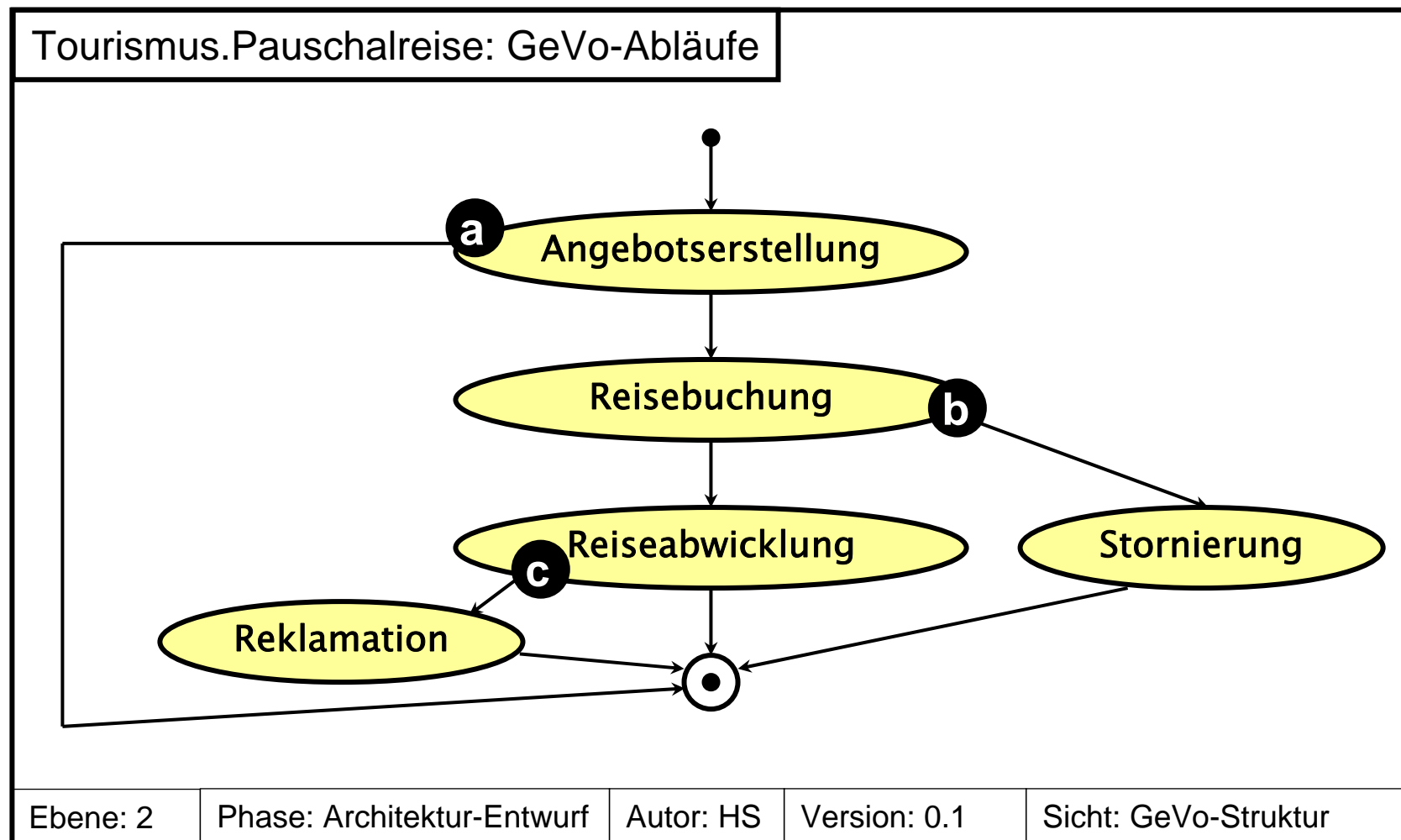
- Ablauf „Pauschalreise“
 - Angebotserstellung
 - Reisebuchung
 - Reiseabwicklung

Die Varianten (Sekundärszenarios):
alle anderen Fälle.

- a) keine Buchung
 - Angebotserstellung
- b) Stornierung
 - Angebotserstellung
 - Reisebuchung
 - Stornierung
- c) Reklamation
 - Angebotserstellung
 - Reisebuchung
 - Reiseabwicklung
 - Reklamation

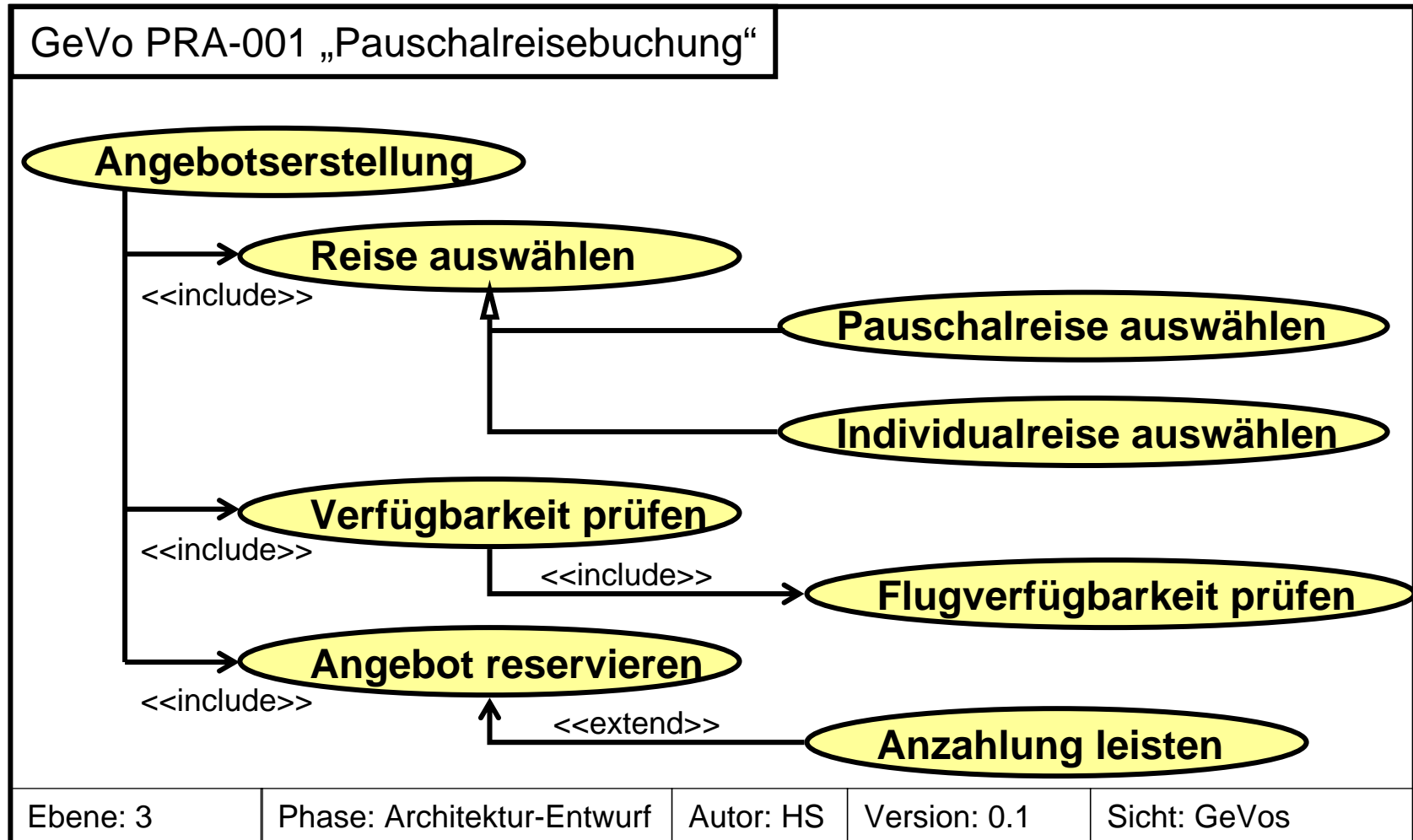
Ebene 3..4

Zusammenfassung alternativer GeVo-Szenarios



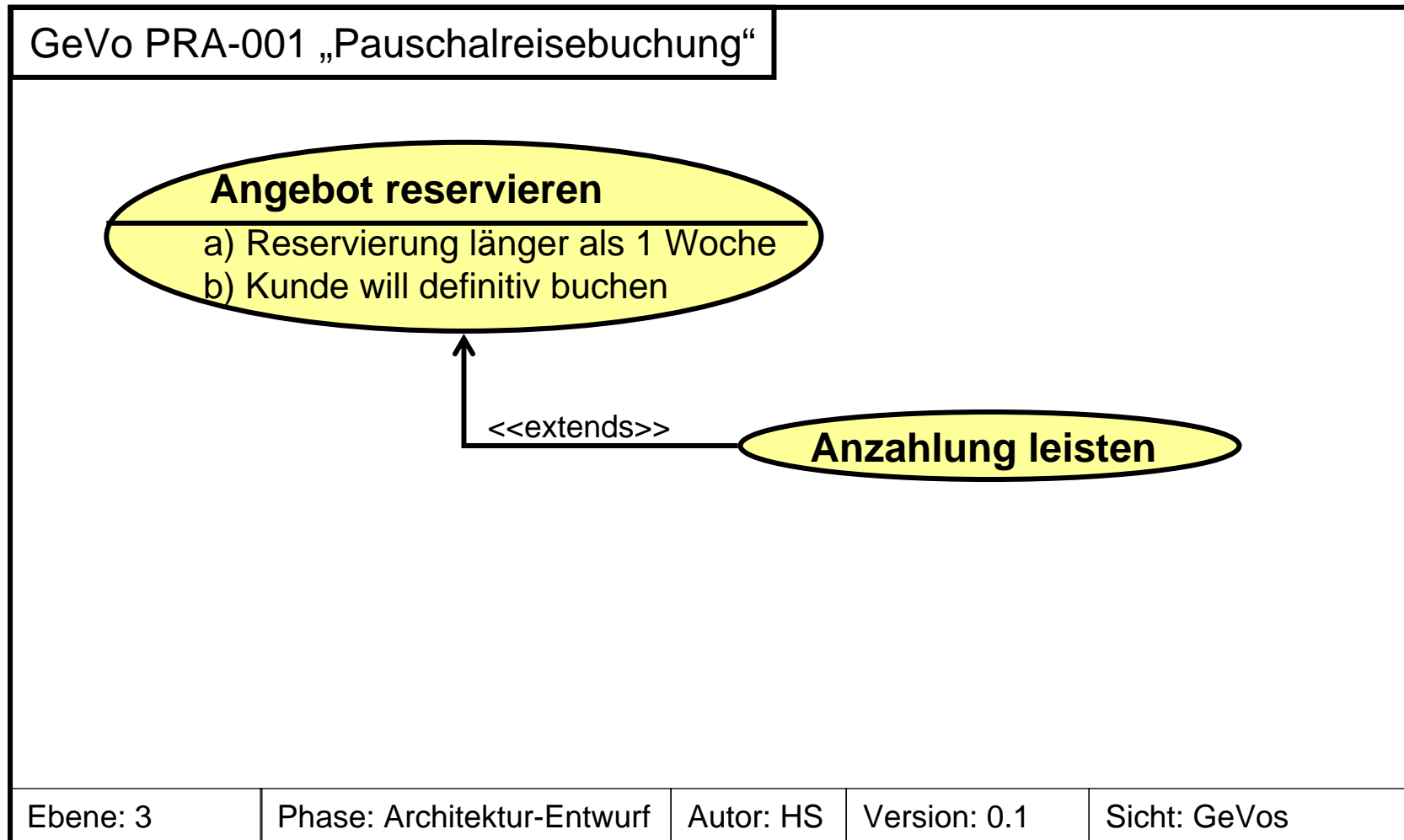
Ebene 3..4

Zusammenhang von GeVos mit include/extend



Ebene 3..4

Extension Points



Ebene 3..4

GeVo-Details in Tabellenschema

GeVo PRA-001 „Pauschalreise“				
Kurzbeschreibung Ein Sachbearbeiter im Reisebüro oder Callcenter macht dem Kunden ein Angebot. Der Kunde bucht die Reise, bezahlt und tritt die Reise an.				
Auslöser Kunde äußert Pauschalreisewunsch	Beteiligte Akteure 1. Kunde, 2. Sachbearbeiter			
Vorbedingung Angebot ist verfügbar	Häufigkeit ca. 10⁴ pro Tag und Vertriebskanal			
Primärszenario 1. Angebotserstellung 2. Reisebuchung 3. Reiseabwicklung	Varianten a) 2. Abbruch b) 3. Stornierung c) 4. Reklamation			
Nachbedingung Kapazitätsplanung angepasst	Referenzen -			
Ergebnis Kunde reiste.	Anmerkungen, offene Punkte -			
Ebene: 3	Phase: Architektur-Entwurf	Autor: HS	Version: 0.1	Sicht: GeVos-Details

Zusammenfassung Geschäftsvorfälle

- **Definition**
 - Jeder GeVo besteht aus vielen Einzelschritten, die maschinell/ manuell von verschiedenen Systemen/Aktoren erbracht werden.
- **Darstellung**
 - Die Abläufe und ihre Verfeinerung werden zunächst durch hierarchischen Text, später dann durch Aktivitätsdiagramme festgehalten.
 - Der Zusammenhang der GeVos wird durch GeVo-Diagramme dargestellt, die Details einzelner GeVos werden in einem vorgegebenen Schema tabellarisch angegeben.
- **Herstellung**
 - Die Menge der GeVos und ihre Details sind entweder bekannt (Vorwissen, Vorgängersystem, Kunden-Dokumente), oder werden z. B. durch Rollenspiele und Interviews erhoben.

Betrachtete Aspekte und Darstellungsmittel

Kontext

Einbettung

Quantifizierung

Fachliches Klassenmodell

Zusammenhänge

Objektlebenszyklus

Klassendetails

Struktur

Facharchitektur

Subsysteme & Schnittstellen

(fachliche) Interaktionen

GeVo

hierarchischer Text

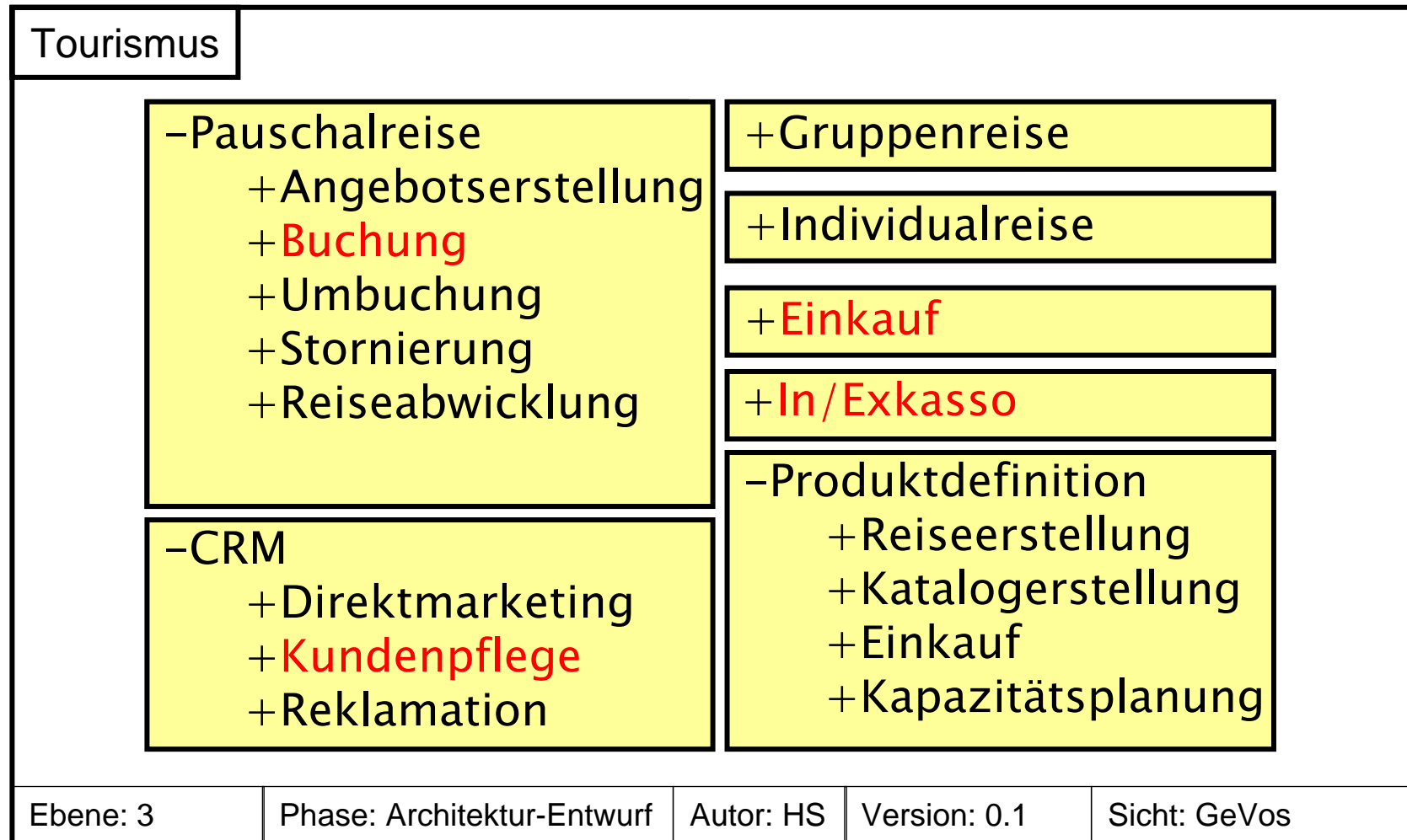
GeVo-Diagramm

GeVo-Tabelle

Aktivitätsdiagramm

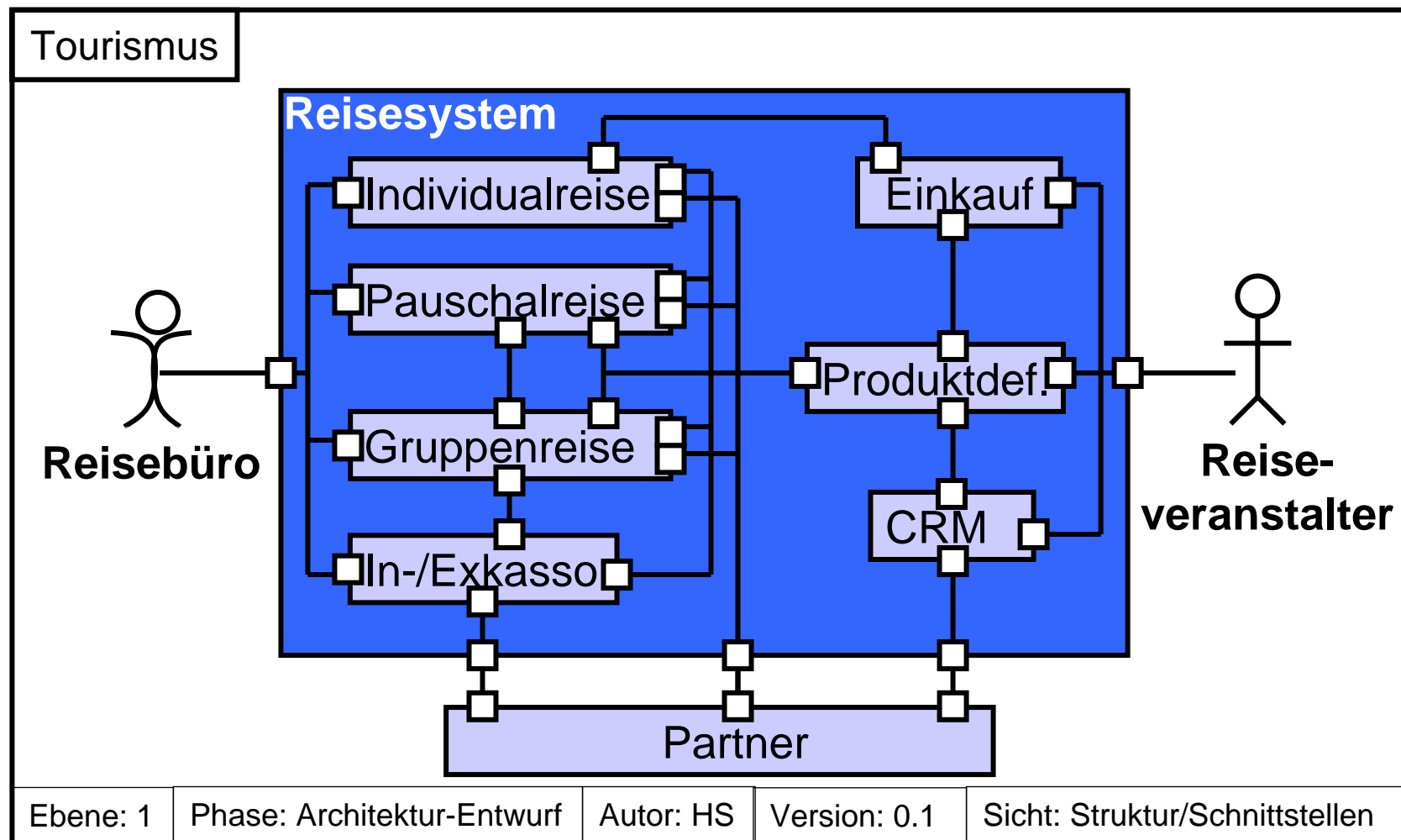
Ableitung von Schnittstellen

Ausgangspunkt Facharchitektur



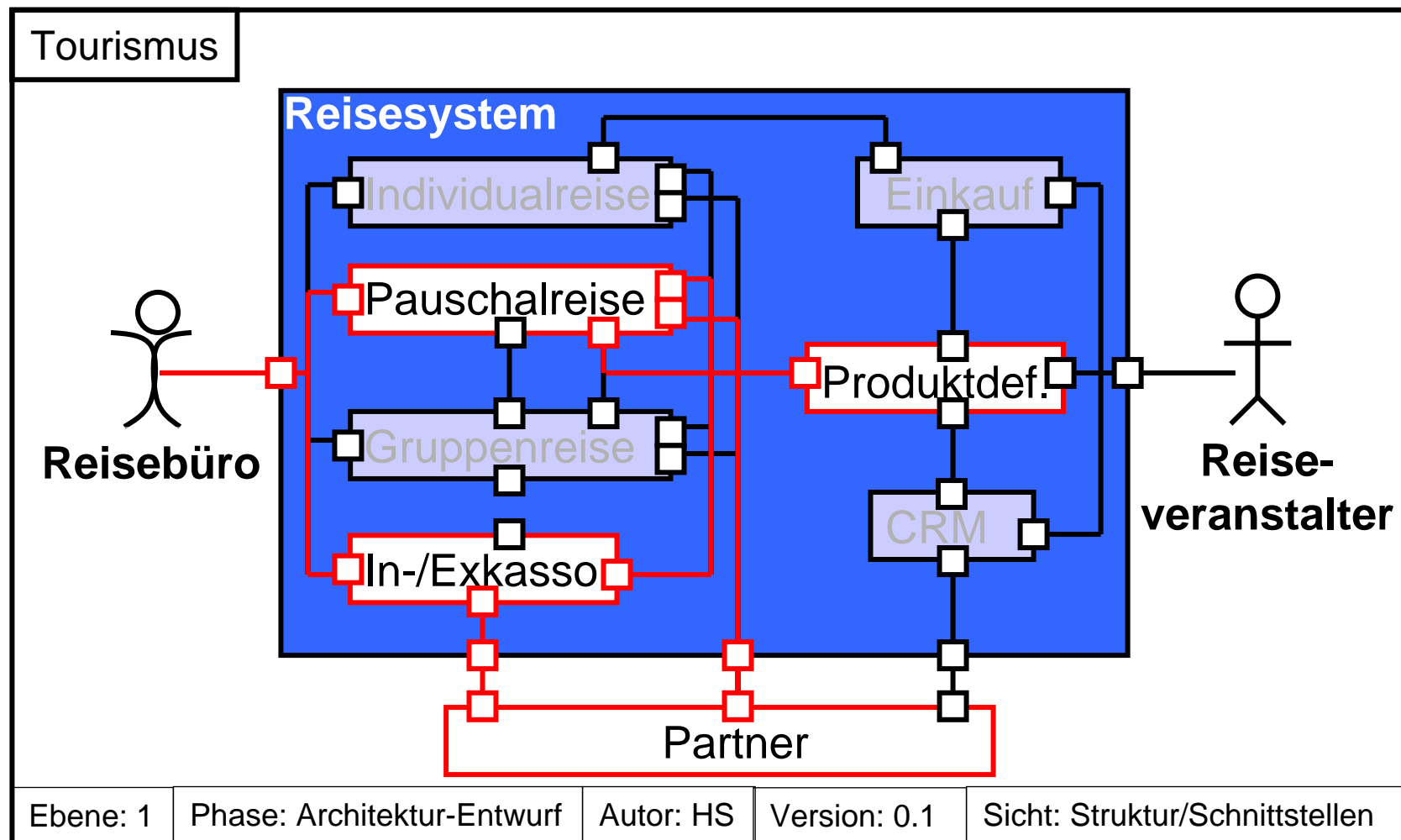
Ableitung von Schnittstellen

Ausgangspunkt Facharchitektur



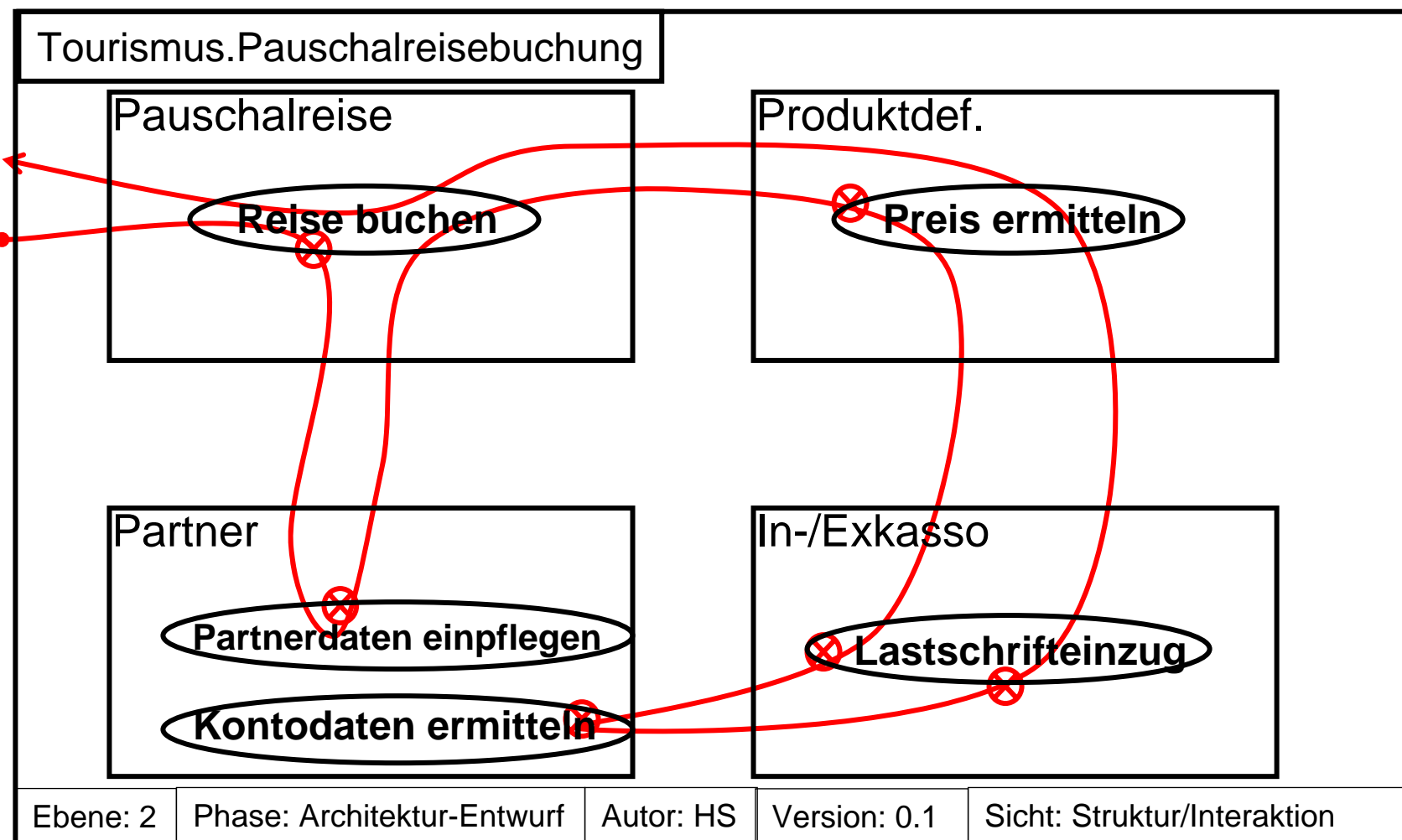
Ableitung von Schnittstellen

Ausgangspunkt Facharchitektur

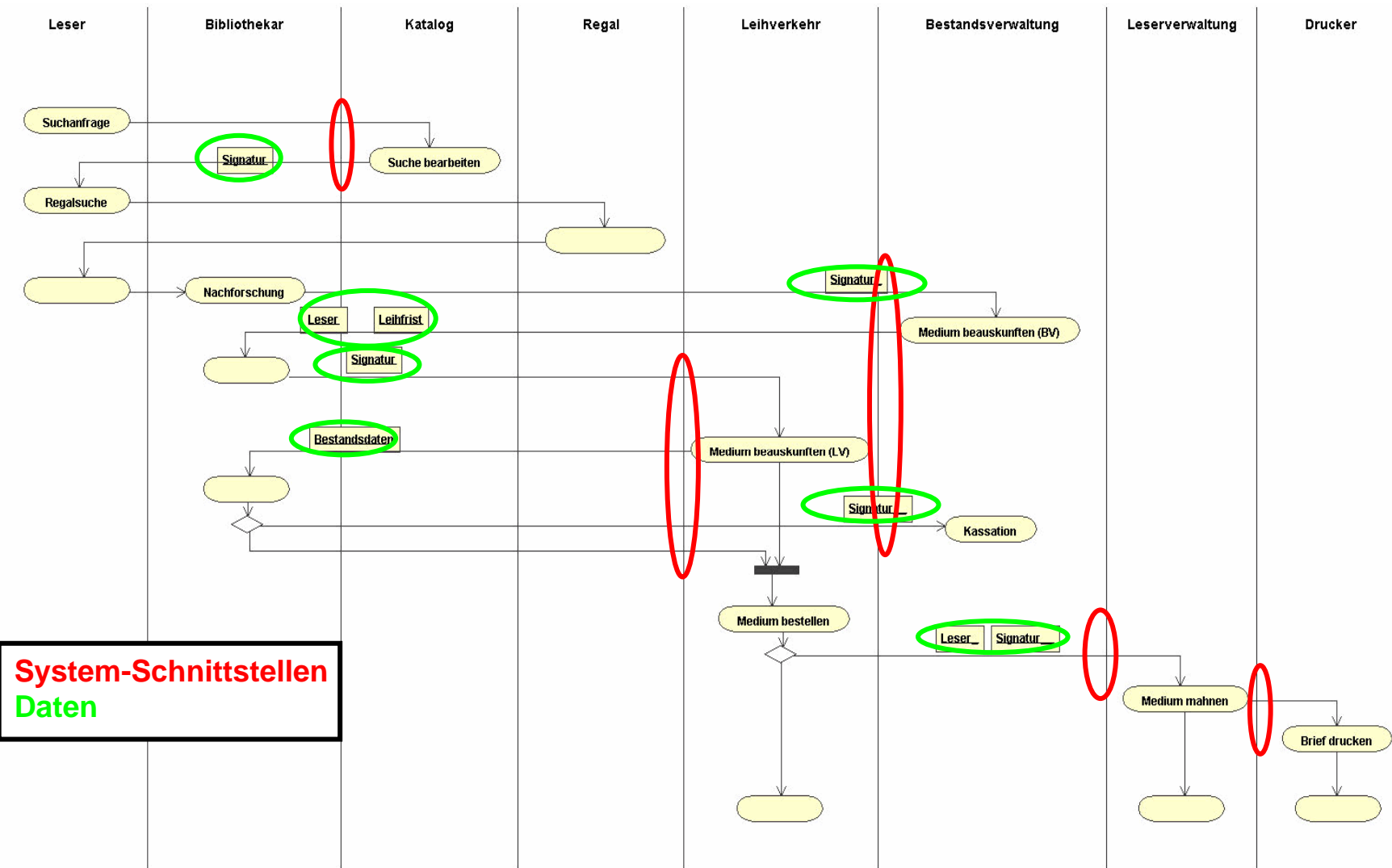


Ableitung von Schnittstellen

Von Facharchitektur zu Nutzfalkarten

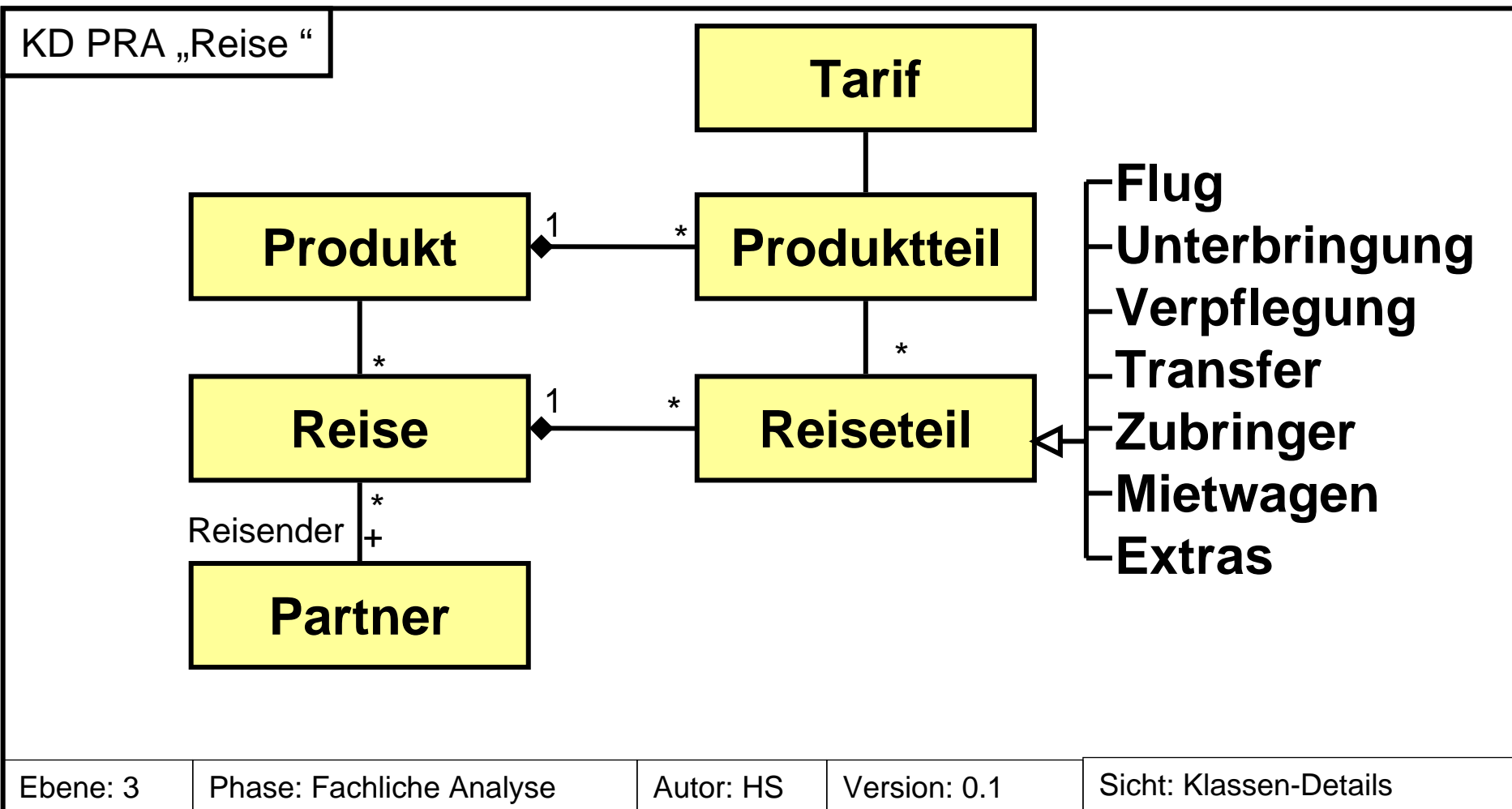


Alternative: Aktivitätsdiagramme statt Nutzfalkarten



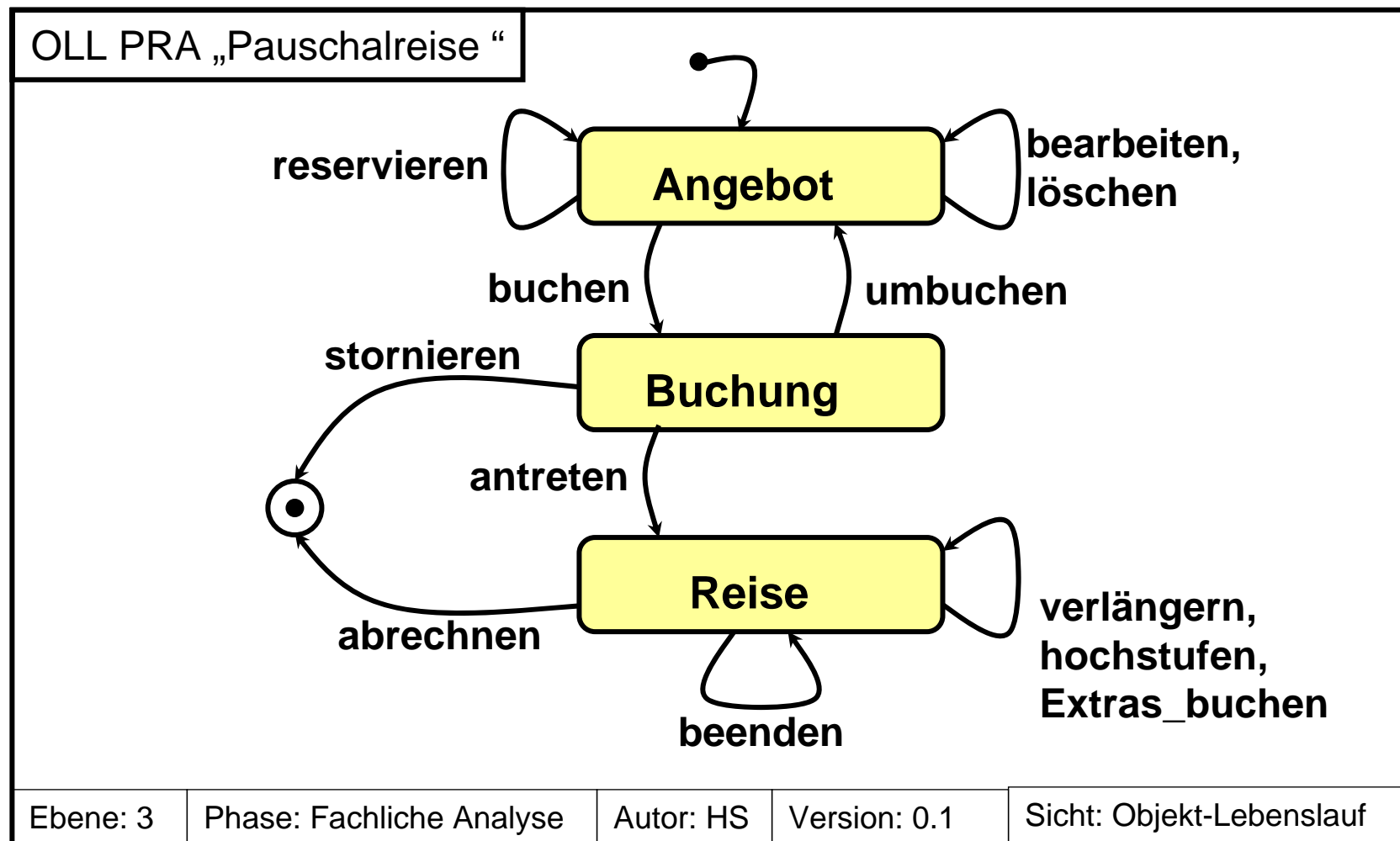
Klassenübersicht

„Angebot/Buchung/Reise“



Objektlebenslauf

„Angebot/Buchung/Reise“



Klassendetails

„Angebot/Buchung/Reise“

KD PRA „Reise“

Angebot

+ von: Datum
 + bis: Datum
 + ab: Ort
 + preis: Euro
 + anmerkungen: String

+ buchen(): Buchung
 + berechnen(): Euro
 + ist_verfügbar(): bool
 + reservieren(frist: Datum): void

Buchung

+ buchungsdatum: Datum
 + sachbearbeiter: Partner
 + agentur: Partner
 + anzahlung: Euro
 + restforderung: Euro
 + tickets_verschickt: bool

+ stornieren(): Angebot
 + antreten(): Reise
 + umbuchen(): Buchung
 + tickets_ausstellen(): void

Reise

+ ist_abgerechnet: bool

+ hochstufen(): Reise
 + verlängern() : Reise
 + beenden(): Reise
 + reklamieren(): Reise
 + abrechnen() : void

Ebene: 3

Phase: Fachliche Analyse

Autor: HS

Version: 0.1

Sicht: Klassen-Details

Hilfsmittel 1: Repository

Problem

- Ein Repository ist eine Datenbank für semantisch komplexe Daten.
- CASE-Tools haben zwangsläufig ein Repository eingebaut. Dies kann
 - als Datenbank realisiert sein (z. B. Rational Rose, Aonix StP),
 - als XMI-Datei gespeichert werden (z.B. Magic Draw, ArgoUML),
 - oder im Code versteckt sein (Together).
- Oft kann kein Repository benutzt werden. Mögliche Gründe:
 - Es ist keines (auf der „richtigen“ Plattform) vorhanden.
 - Es darf nicht für alle Zwecke/von allen Personen genutzt werden.
 - Es ist aufwendig zu bedienen, und man soll „nur mal eben schnell“... („Overkill“).
 - Es ist nicht (gut) integriert mit den anderen Werkzeugen.

Hilfsmittel 1: Repository

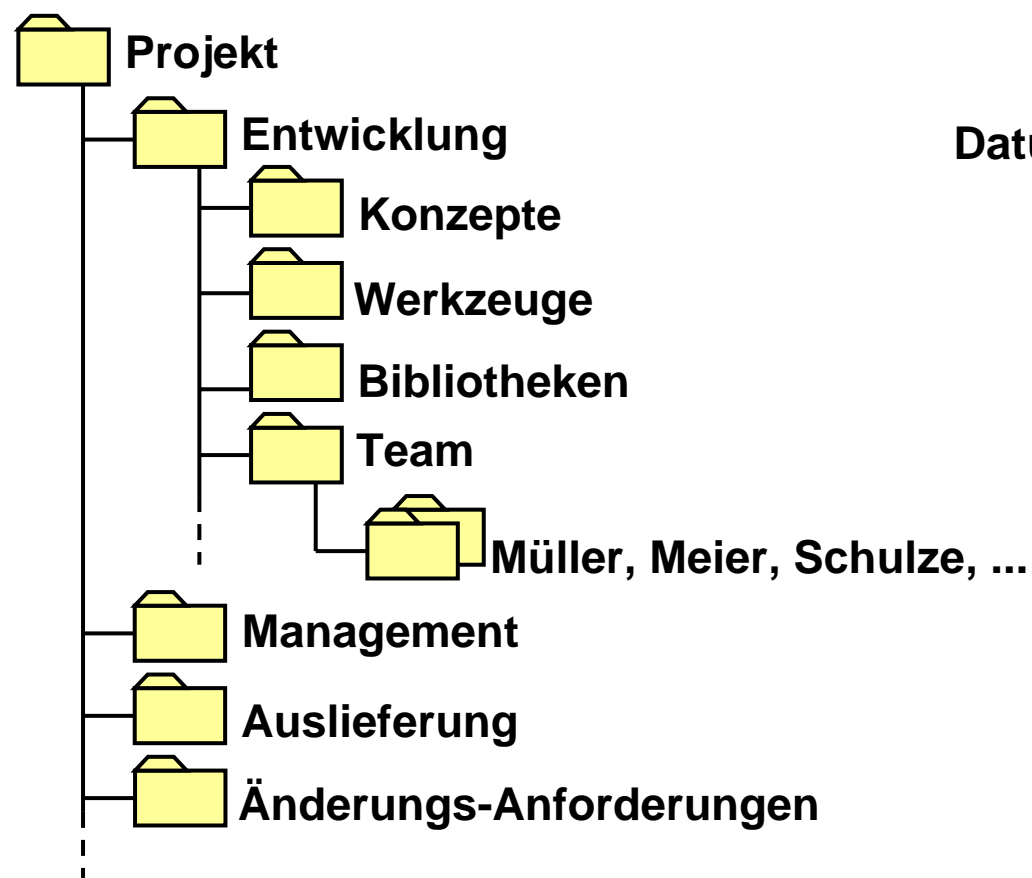
Lösung: Primitiv-Repository

- **Vorteile**
 - Gibt es auf jeder Plattform
 - keinerlei technischer Aufwand, minimale Kosten
 - keine Genehmigung erforderlich
- **Nachteile**
 - Vergabe von Identifiern & Zuständen wird nicht automatisch erledigt, dadurch Mehraufwand, und mögliche Inkonsistenzen
 - nebenläufiges Arbeiten schwierig
- **Anzuwenden bei**
 - kleinen Projekten / in der Frühphase
 - in technisch rückständigen Umgebungen – d.h. in 70% aller SW-Buden

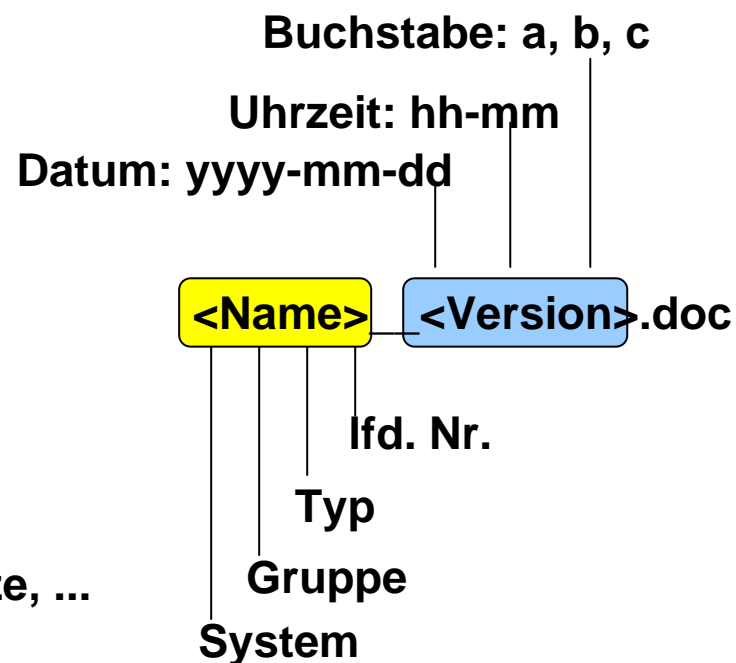
Hilfsmittel 1: Repository

Lösung: Primitiv-Repository

Ablagestruktur



Dokumente/Modelle



Beispiele

•Bib-AUS_GP_007

Hilfsmittel 2: Glossar

- Ist von Anfang an zu führen, zu pflegen und zu verwenden!
- In Excel, Word o.ä. -> Sortierung der Einträge

Begriff	Synonyme	Erklärung(en)	Verantw.	Status
Leser	-	Natürliche Person, die mit Namen, Geburtsdatum und Adresse im Leserverwaltungssystem erfasst ist.	Störrle	ok

- Sollte zentral gepflegt werden
 - z.B. über einheitlich Email: glossar@firma.com
 - 4-Augen-Prinzip (QS)
 - verfügbar im Intranet (-> Knowledge Management)

Hilfsmittel 2: Glossar Zweck & Gütekriterien

- Standardisierte Sprache
 - erhöht die Konsistenz, und
 - hilft Missverständnisse vermeiden.
- Technische Prosa darf nicht nur „langweilig“ sein, sie muß es!
- und bitte: kein Denglisch!
„Um die Usage-Policy zu updaten muss zuerst das File vom User downgeloadet werden.“

Weitere Tipps bei:

Peter Rechenberg: Technisches Schreiben. Hanser, 2002

Literatur für B.1

- Hofmeister, Nord, Soni: Applied Software Architecture. Addison-Wesley, 2000
- Störrle: Models of Software Architecture. Book-on-demand, 2001
- Sewell, Sewell: The Software Architect's Profession. Prentice Hall PTR, 2002
- Rechenberg: Technisches Schreiben. Hanser, 2002
- Mellor, Balcer: Executable UML. Addison-Wesley, 2002
- Marshall: Enterprise Modeling with UML. Addison-Wesley, 2000

Gliederung B.1

- **Einleitung in den Block B „Software-Architektur“**
 - Definition, Gegenstand, Motivation
 - Gliederung und Einbettung in die Vorlesung
- **Teil 1: Methodischer Entwurf**
 - Notationen & Konzepte
 - Fallbeispiel
 - Hilfsmittel & Literatur
- **Ausblick auf Teil 2: Systemarchitektur**

Ausblick auf Block B Teil 2: Systemarchitektur

