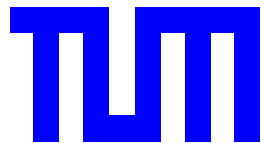
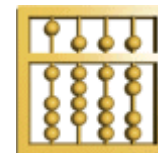

Was ist eine Spezifikation?

Und wie spezifizieren wir?

Manfred Broy



Technische Universität München
Institut für Informatik



Was ist eine Anforderungsspezifikation?

- Eine Beschreibung aller Anforderungen an ein Software/Hardwaresystem mit Betonung von
 - ◇ Systemnutzen (Nutzungsfälle)
 - ◇ Systemverhalten (beobachtbares Verhalten, Szenarien)
 - ◇ der Nebenbedingungen
 - Zuverlässigkeit
 - Performanz
 - Kompatibilität mit Vorgaben

Der Prozeß der Erarbeitung der Anforderungen
heißt Requirements Engineering

Stellenwert von RE für softwareintensive Systeme

- Qualität eines Systems kann nicht sein besser als seine Anforderungen
- RE bestimmt Systemnutzbarkeit, -kosten und -komplexität
- Anforderungen unklar, fehleranfällig, instabil, Entwurfsräume größer, Spielräume für Anforderungen
- Verhalten softwareintensiver Systeme komplex
- Rollen- und Zielkonflikte: Kommunikation zwischen Beteiligten (Nutzer, Manager, Ingenieur) schwierig
- RE ist an der Grenzlinie zwischen informeller und formaler Arbeitsweise
- RE ist immer ein Prozess des Lernens und Verstehens

Schwierigkeiten im RE

- Wie die richtigen Anforderungen finden?
- Komplexität der Anforderungen: Wie dokumentieren?
- Funktionale und nichtfunktionale Anforderungen
- Strukturierung, Repräsentation, Analyse von Anforderungen
 - ◇ Abstraktionsgrad
 - ◇ Inkonsistenz
 - ◇ Vollständigkeit
 - ◇ Über/Unterspezifikation
- Adäquate Modellbildung und Beschreibung
- Formalisierung und Beschreibungsmittel
- Vorgehensweise, Anforderungsmanagement

Schlüsselfaktoren im RE

- Nutzerpartizipation
- Systematische Anforderungsidentifikation und -festlegung
- Wohlgewählte "Gedankenmodelle"
- Modellbasiertes RE
- Strukturierung von Anforderungen
- Logische Analyse von Anforderungen
- Nichtfunktionale Anforderungen
- Von den Anforderungen zum Design
- Validierung
- Werkzeugunterstützung

Personengruppen involviert

- Personen verantwortlich für die technische Entwicklung und Realisation von die System, die *Ingenieure*;
- Personen verantwortlich für das Management und Organisation des Projekts, die *Projektmanager*;
- Personen verantwortlich für Finanzierung und die Marketing der Produkte, genannt die *Auftraggeber*, *Projektträger* und *Marketingmanager*;
- Personen, die das Ergebnis des Entwicklungsprozesses nutzen, die *Kunden* und *Nutzer*.

Problem: Kommunikationsschwierigkeiten, Ziel- und Rollenkonflikte der involvierten Gruppen

Vorteile eines systematischen Anforderungs-Engineering-Prozesses:

Sorgfältig erarbeitete und dokumentierte Anforderungen sind Basis für

- gemeinsames Verständnis und Verständigung,
- Produktplanung und Marketing,
- Projektplanung und Entwurfsentscheidungen,
- Begrenzung der Entwicklungsanstrengungen,
- Kostenschätzung und Zeitplanung,
- Startpunkt für Entwurf und Implementierung,
- Systemdokumentation,
- Validierung und Verifikation,
- Leichte, einfache Nutzbarkeit,
- Erweiterungen.

Überlappende Teilphasen des RE Prozesses

- (1) Vor-Phase:
Ausarbeitung der generellen Produktstrategie und -
positionierung, Identifikation der strategischen Ziele,
Grobe Anforderungsbeschreibung, Risiken, Vorgaben

- (2) Hauptphase:
Domänen Engineering, Identifikation der detaillierten
fachlichen und technischen Anforderungen,
Strukturierung und Validierung der Anforderungen

- (3) Nach-Phase:
Änderungsmanagement, Anforderungsverfolgung und
Verifikation in der Realisierung

Detaillierter: Aktivitäten in RE

- Identifikation der globalen Ziele und Nebenbedingungen,
- Domänenanalyse und Domänenmodellierung,
- Anforderungserfassung, Sammlung informeller Anforderungen,
- Anforderungsanalyse, Bewertung, Strukturierung und Priorisierung, Risikoanalyse
- Anforderungskonsolidierung und Spezifikation,
- Anforderungsvalidierung,
- Anforderungsumsetzung und -verfolgung,
- Anforderungsverifikation,
- Anforderungsänderungsmanagement

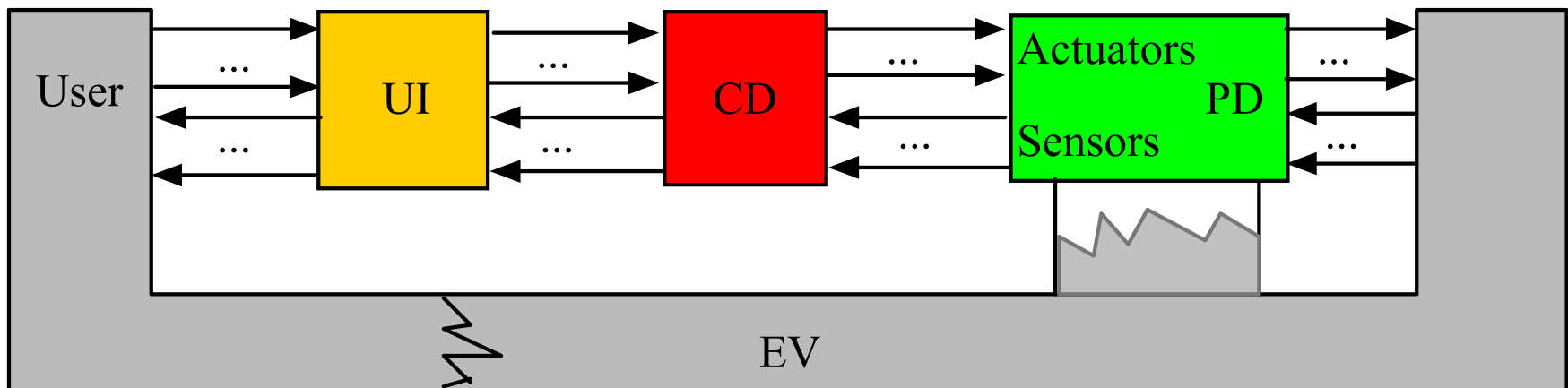
Anforderungsanalyse

- Domänenanalyse
- Anforderungserfassung:
Nutzerpartizipation und Nutzungsangemessenheit
 - Nutzerprozesse an der Schnittstelle des Systems,
 - Nutzermodell,
 - Modus des Betriebs,
 - kritikale Situationen und Nebenbedingungen,
 - Funktionsumfang,
 - Nutzungsfälle,
 - Fehlerbehandlung.

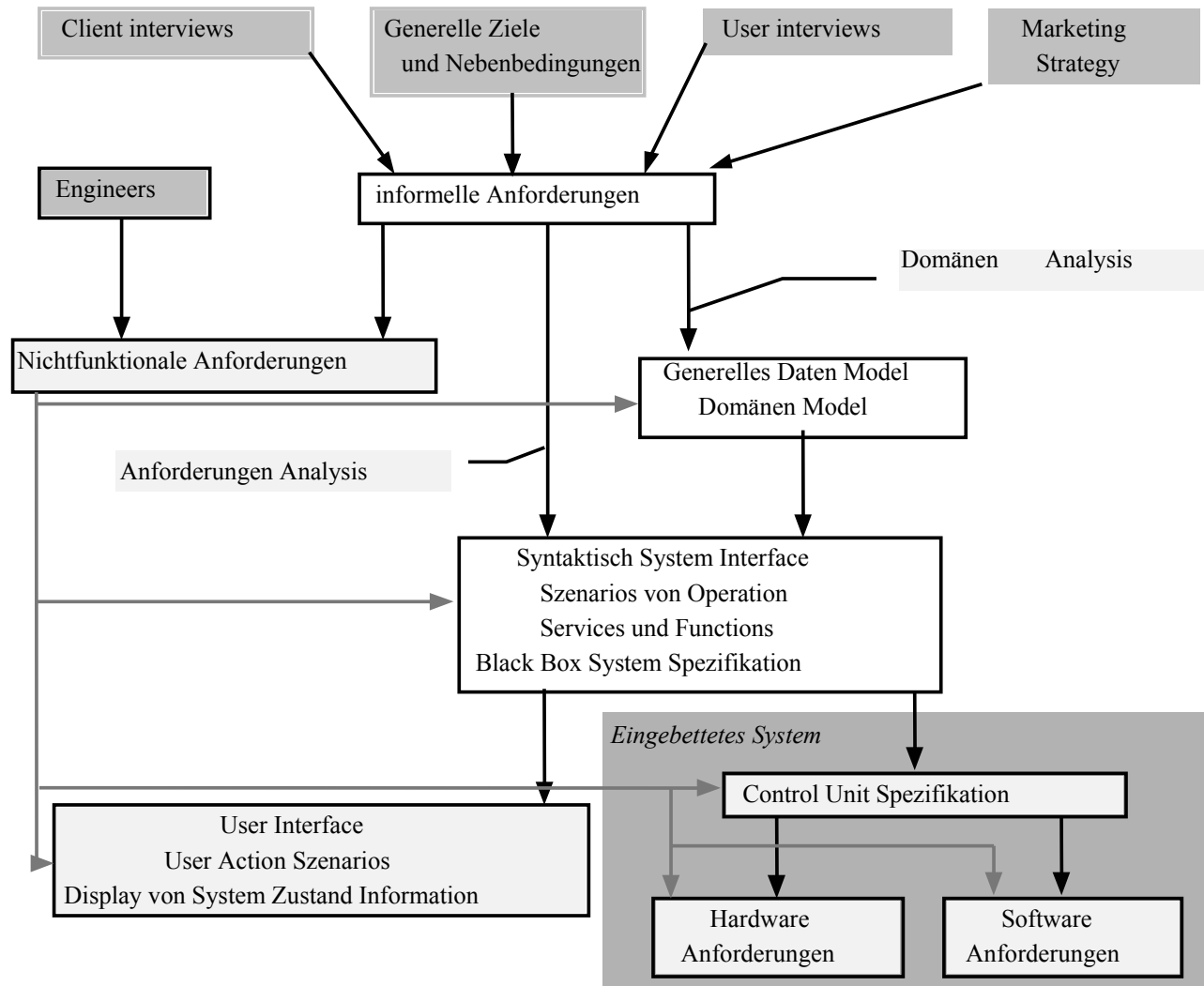
Strukturierte Systemsichten

- **funktionale Architektur**: Struktur der Funktionen aus Sicht der Nutzungsfälle, ihrer Beziehungen und Abhängigkeiten,
- **logische Architektur**: Strukturierung des Systems in funktionale Teilsysteme („Komponenten“), ihrer Rollen und ihrer Kommunikationsstruktur,
- **Implementierungsarchitektur**: Strukturierung des Systems in Programmmodule und ihrer Realisation durch Code,
- **Deployment-Architektur**: Abbildung der Softwareanteile auf die Hardware (Partitionierung)

Anforderungenspezifikation für eingebettete Systeme



Grobstruktur eines RE Prozesses



Anforderungsvalidierung

- Reflektieren die erfassten Anforderungen tatsächlich die Nutzererwartungen und -bedürfnisse?
- Sind die erfassten Anforderungen konsistent, vollständig, wohl-strukturiert und wohldokumentiert?
- Methoden zum Sicherstellen der Nutzbarkeit:
 - ◇ Reviews der Anforderungen,
 - ◇ Strukturierte „Walk Throughs“ und Inspektionen,
 - ◇ Werkzeug basierte Konsistenzchecks,
 - ◇ Verifikation von sicherheitskritischen Eigenschaften,
 - ◇ Prototypen und Simulation einschließlich Nutzerexperimente.

Je höher die Formalisierung, um so gezielterer Werkzeugeinsatz möglich!

Methoden der Anforderungsverifikation

- Zeigen, dass die Implementierung die Anforderungen erfüllt, durch folgende Techniken
 - ◇ Inspektion
 - ◇ Review
 - ◇ Test
 - ◇ Verifikation
- Validieren durch Nutzbarkeitsüberprüfungen

Strukturierung der Anforderungen durch Systemsichten - Beschreibungsmittel

- *Datenmodelle*: Datentypdeklarationen, Abstrakte Datentypen, Entity/Relationship-Diagramme, Modellierung der Ereignisse, Zustände, Nachrichten eines Systems,
- *Prozessmodelle*: Ereignisspuren, Message Sequence Charts, Interaktionsdiagramme, Prozessdiagramme,
- *Komponentenmodelle* (Schnittstellenmodelle, black box view): syntaktische Schnittstellen, Interaktionsdiagramme, Input/Output-Relationen, Schnittstellenautomaten
- *Modelle verteilter Systeme*: Datenflussdiagramme, System- und Softwarearchitekturen,
- *Zustandübergangsmodelle*: Zustandstransitionsdiagramme,
- *Interaktionsmodelle*

Nutzungsfälle: Funktionsnetze und Dienste

- Strukturierung der Anforderungen in Nutzungsfälle (Funktionen)
 - ◇ Beschreibung der Nutzungsfälle
 - ◇ Darstellung in Szenarien als Interaktionsmuster
 - ◇ Formalisierung als Dienste
- Strukturierung der Funktionen in Teilfunktionen
- Angabe der Beziehungen und Abhängigkeiten zwischen den Funktionen
- Priorisierung
- Risikoanalyse/ kritische Funktionen

Erfassung der globalen Systemanforderungen: Strukturierung

Input: Informelle Beschreibung der groben Anforderungen und Nebenbedingungen

- Ziel: Informelle detaillierte Erfassung der globalen Systemanforderungen

Ergebnisse

- Globale Systembeschreibung
- Repository der wichtigen Begriffe, ihrer Erklärungen und Definitionen
- Klassifizierung der Anforderungen
- Strukturierung

Technisches und fachliches RE: Formalisierung

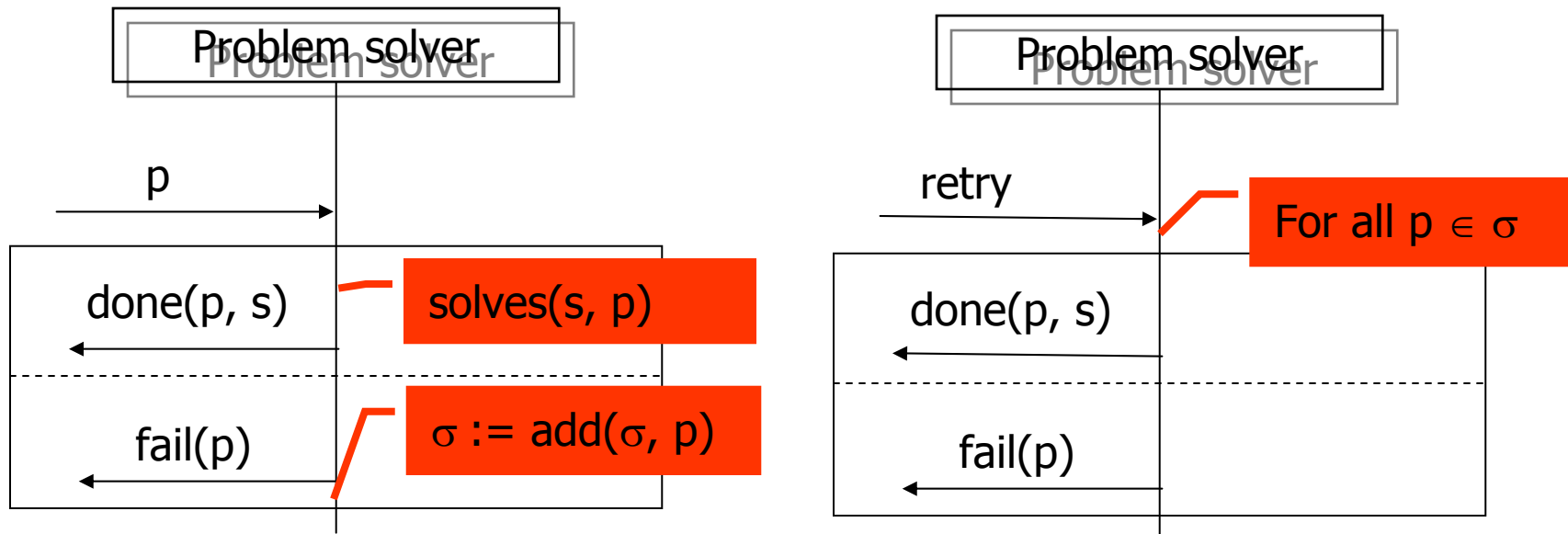
Nutzungsfälle – Funktionsnetze

- Liste aller Nutzungsfälle für das System, hierarchisch strukturiert (falls angemessen) und Abhängigkeiten
- Rationale für Anforderungen

Modellierung der Systemumgebung

- Beschreibung der Schnittstellen zwischen System und Umgebung
- informelle Beschreibung der Annahmen über die Umgebung - Formalisierung

MSCs für Szenariospezifikationen - Nutzungsfälle das Beispiel Problemsolver



$\text{Problem_solver}[\sigma].p \hat{x} = \text{done}(p,s) \hat{x} \text{Problem_solver}[\sigma].x \wedge \text{solves}(s, p)$

✓ $\text{Problem_solver}[\sigma].p \hat{x} = \text{fail}(p) \hat{x} \text{Problem_solver}[\text{add}(\sigma, p)].x$

$\text{Problem_solver}[\sigma].\text{retry} \hat{x} = \text{tryall}(\sigma) \hat{x} \text{Problem_solver}[\text{fails}(\text{tryall}(\sigma))].x$

Details der Anforderungen

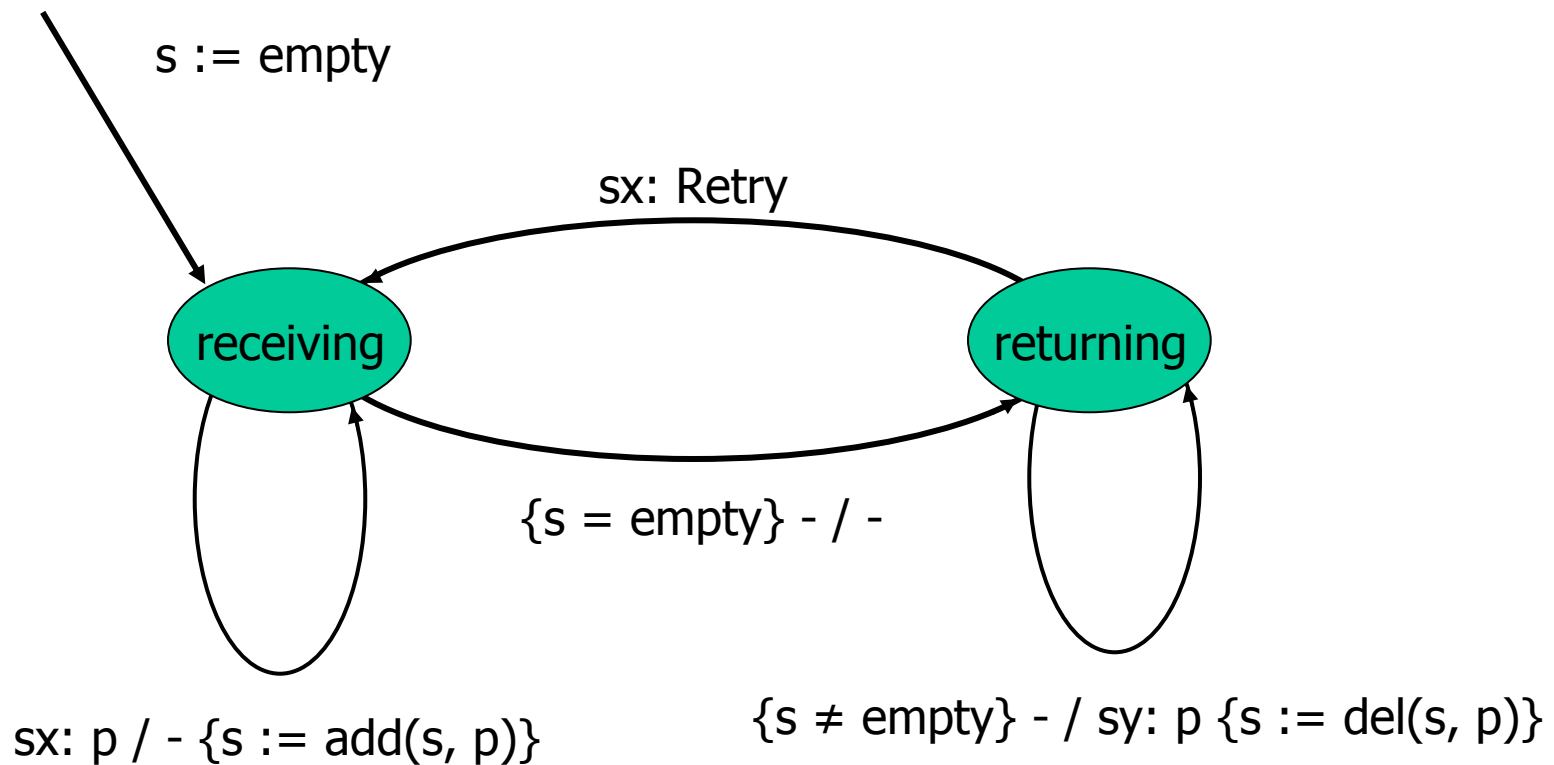
Sicherheit

- Regeln und Prinzipien des Betriebs des System
- Risikoanalyse
- Beschreibung von Risiken des Systembetriebs, informelle Beschreibung kritikalere Anforderungen
- Zerlegung in Funktionale/Nichtfunktionale Anforderungen
 - ◇ Produkt
 - ◇ Prozess
 - ◇ Randbedingungen

Formale Anforderungsdokumentation

- Detailliertes Domänenmodell
 - ◇ Datenmodell
 - ◇ Regeln
- Externe Schnittstellen
 - ◇ Datenmodell der Nachrichten, Signale, Ereignisse
- Logisches Funktionsmodell der Nutzungsfälle
 - ◇ Szenarios durch MSCs
 - ◇ Fehlermodell
- Detaillierte Verhaltensbeschreibung
 - ◇ Zustandsmodell
- Identifizierung der Black Box Tests

Problemstore als Zustandsmaschine



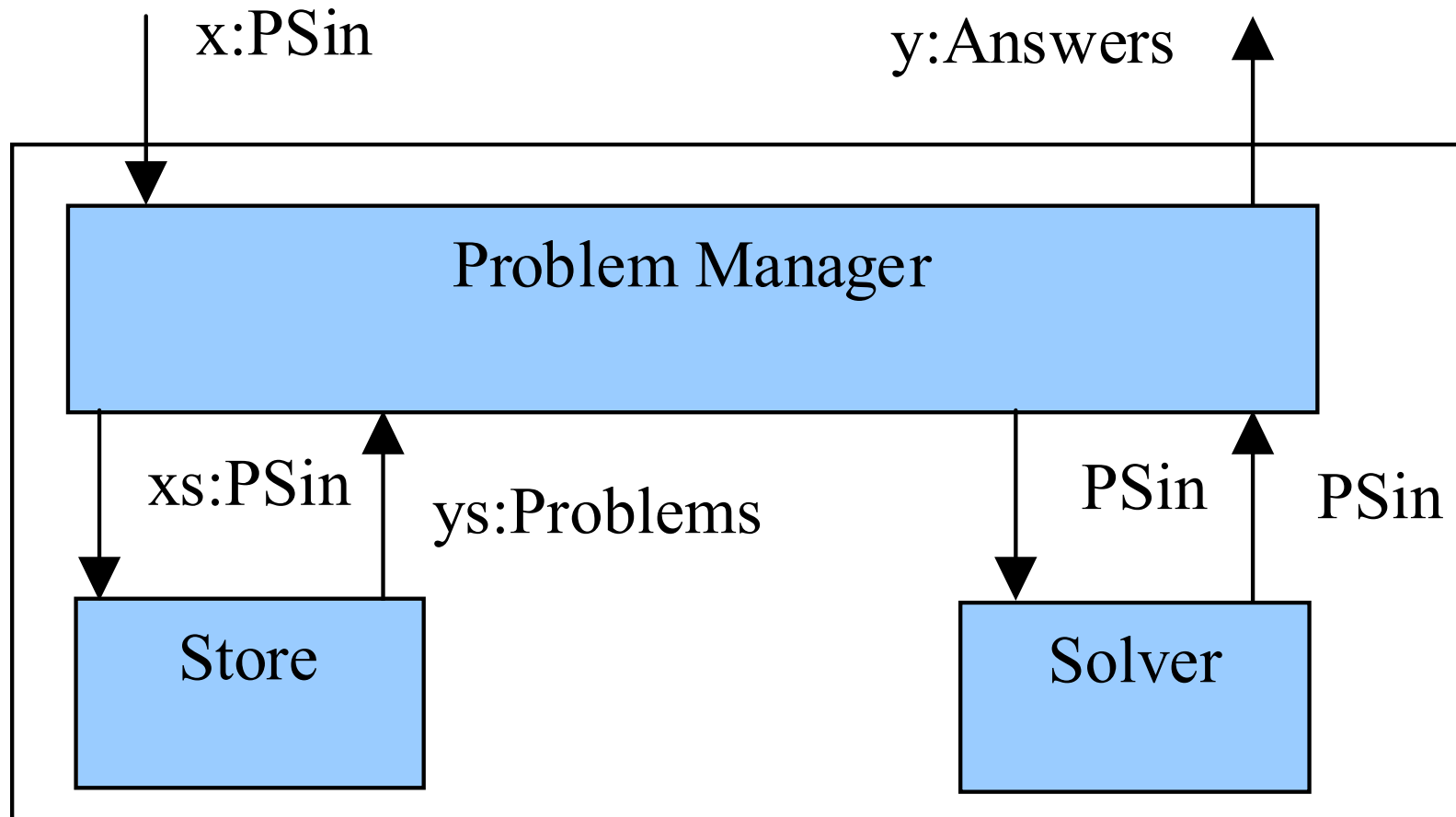
Von Anforderungen zum Design: Dekomposition

Die Anforderungsspezifikation als Ausgangspunkt

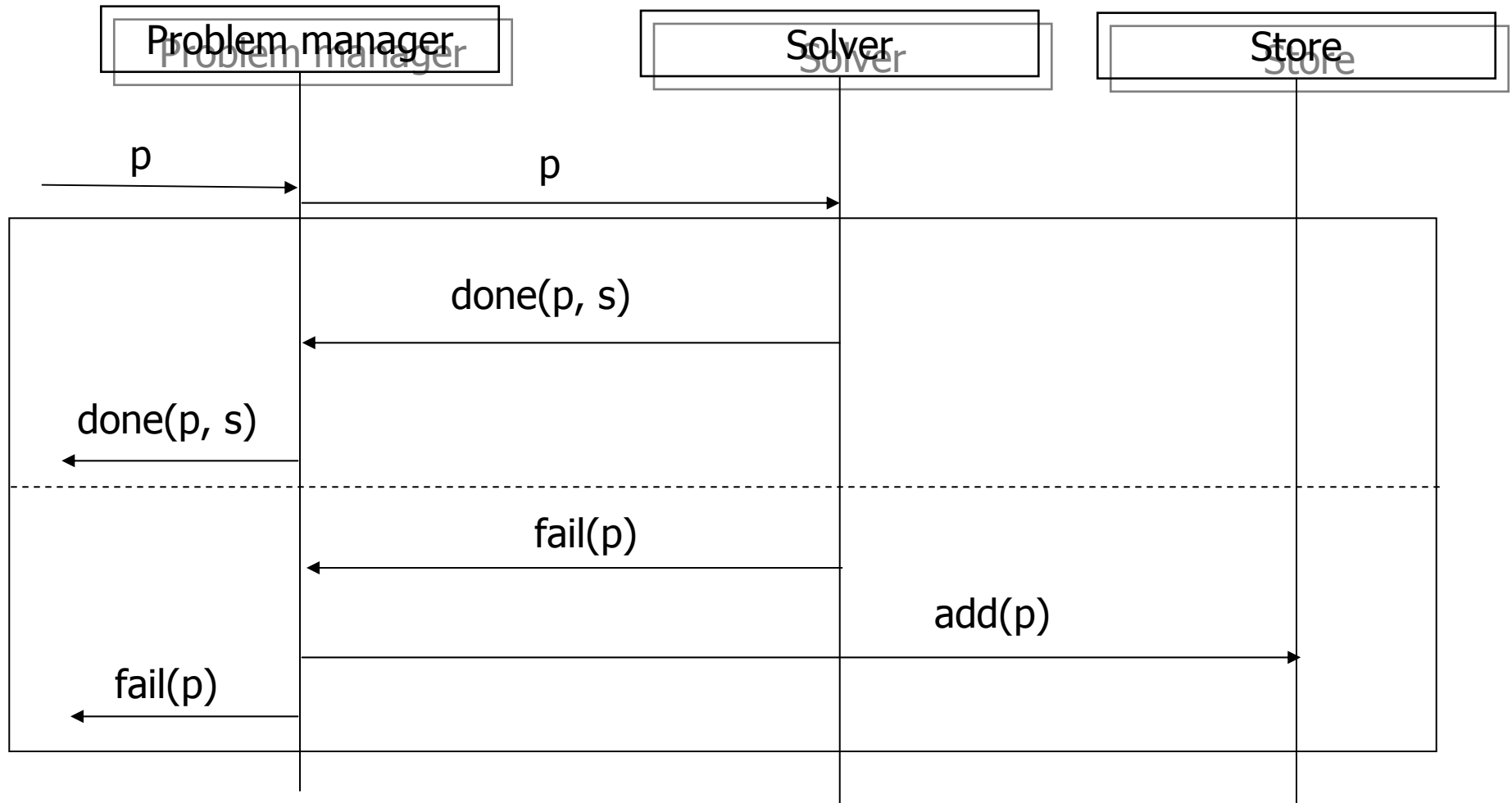
- Zerlegung in Komponenten
- Szenarien der Komponenteninteraktion anhand der Nutzungsfälle
- Schnittstellenspezifikation
- Risikoanalyse
- Nicht-funktionale Anforderungen
- Validierung
- Verifikation

Differenz zwischen Anforderungen und Spezifikation

Beispiel: Der Problem Solver



Detaillierung von Nutzungsfälle



Anforderungsverfolgung

- Rationale: Wo kommen Anforderungen her und wie sind sie begründet?
- Sicherstellen, dass die Anforderungen systematisch in die Entwicklung eingehen
- Verifizieren, dass die Anforderungen berücksichtigt worden sind
- Aufzeigen wie Anforderungen das Design bestimmen
- Überprüfen der Anforderungen in jedem Designschritt
- Ändern der Anforderungen (wenn nötig!) in kontrolliertem Prozeß mit eindeutigen Zuständigkeiten

Abschließende Bemerkungen: Schlüsselfaktoren des RE

- Nutzerpartizipation
- Abstraktion
- Strukturierung
- RE Prozess
- Modellkonstruktion
- Funktionale Anforderungen
- Nichtfunktionale Anforderungen
- Anforderungsverständnis
- Anforderungsrationale
- Anforderungsverfolgung
- Änderungsmanagement