

Martin Glinz

Requirements Engineering I

Kapitel 12

Sichten und Konsistenz



Universität Zürich
Institut für Informatik

12.1 Sichten

- Verschiedene Beteiligte sehen die Aufgaben eines Systems aus **unterschiedlichen Perspektiven**
- Verschiedene Personen in der gleichen Beteiligten-Rolle sehen ein System aus **unterschiedlichen persönlichen Situationen**
- Bei der Gewinnung von Anforderungen die richtige **Balance** finden zwischen
 - notwendigem **Konsens** → **Vereinheitlichung**, Beseitigung von **Inkonsistenzen** zwischen verschiedenen Sichten
 - notwendiger **Variabilität** → **bedürfnisgerechte Adaptierung** an unterschiedliche Gruppen von Beteiligten

Konsensbildung

- Mittel
 - Anforderungen **priorisieren**
 - **Abstimmen**
 - **Kartenabfragen**
 - **moderierte gemeinsame Sitzungen**
 - Stelle eines **Anforderungskoordinators** im Projekt schaffen

- Einflussfaktoren
 - (offizielle) **Organisationsstruktur** – Aufbau- und Ablauforganisation
 - **Gruppensoziologie** – faktische Machtverhältnisse, Kommunikations- und Ablaufstrukturen
 - Bereitschaft, sich zu exponieren, **Verantwortung für Anforderungen** zu übernehmen

12.2 Konsistenz

- Konsistente Anforderungsmodelle:
 - Modell ist **in sich widerspruchsfrei**
 - Modell steht **nicht im Widerspruch** zu den systemrelevanten Phänomenen der **Realität**
- Zu Grunde liegende Annahmen:
 - (a) die **Welt** selbst **ist** in sich **konsistent**
 - (b) die Welt ist **konsistent beschreibbar**
- Beide **Annahmen** sind **nicht überall haltbar**, zum Beispiel
 - Synchronisation verteilter Uhren (a)
 - Relativität von Raum und Zeit (a)
 - Klassische Mechanik vs. Quantenmechanik (a)
 - Verschiedene Sichten unterschiedlicher Beteiligter (b)

Konsistenz von Anforderungsmodellen

- **Hauptproblem**
Spezifikationen, die verschiedene **Aspekte** jeweils in einem **Teilmodell** behandeln, zum Beispiel
 - Statische Aspekte in Klassen oder Objektmodell
 - Interaktionsaspekt in Szenarien- oder Anwendungsfallmodell
 - Verhaltensaspekt in Zustandsmodell
- **Optionen**
 - **Erzwingen** von Konsistenz durch **integrierte Modelle** (z.B. ADORA)
 - **Leichtgewichtige** Konsistenz, primär durch **Verweise** zwischen Teilmodellen (z.B. Glinz 2000)
 - Leben mit **Inkonsistenz**

Leben mit Inkonsistenz

Inkonsistenzen

- **Erkennen** – zum Beispiel Verletzung von Regeln, Widersprüche
- **Analysieren** – Auswirkungen? Risiko?
- **Behandeln** – Mögliche Optionen:
 - Ignorieren
 - Tolerieren
 - Auflösen
 - Umgehen
 - Auf später verschieben
- **Verfolgen** – Wirkung der Behandlungsmaßnahmen

Literatur

Glinz, M. (2000). A Lightweight Approach to Consistency of Scenarios and Class Models. *Proceedings 4th IEEE International Conference on Requirements Engineering*. Schaumburg, Ill. 49-58.

Nuseibeh, B., S. Easterbrook, A. Russo (2000). Leveraging Inconsistency in Software Development. *IEEE Computer* **33**, 4 (April 2000). 24-29.

Nuseibeh, B., J. Kramer, A. Finkelstein (2003). ViewPoints: Meaningful Relationships are Difficult! *Proceedings of the 25th International Conference on Software Engineering (ICSE'03)*, Portland, Oregon. 676-681.

Eine Fülle weiterer Literaturveweise findet sich in [Nuseibeh, Kramer, Finkelstein 2003]