Towards a Formalization of the Model by Jonsson and Olovsson

Herbsttreffen des Fachbereichs SYS

Billy Naumann

21. November 2019

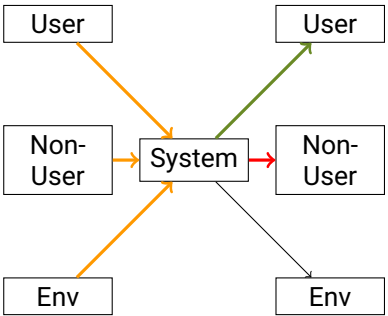
Ausgangspunkt

- Historisch: *Dependability* und *Security* als getrennte Communities
- In eingebetteten Systemen: Ungenutze Synergien unnötiger Kostenfaktor
- In sicherheitskritischen Systemen: Formaler Nachweis notwendig
- Frage nach Formalisierungs- und Operationalisierungsmethode zur Darstellung dieser Synergien

Dependability- und Securitybegriffe nach Avižienis et al. (2004)

- *Dependability* und *Security* als Überbegriff
- *Service* als korrektes Verhalten eines *Systems* gegenüber seinen Nutzern
- Bedrohungen
  - *Faults - „Adjudged or hypothesized cause of an error“*
  - *Errors - „Part of the [...] state of the system that may lead to its subsequent service failure“*
  - *Failures - „Event that occurs when the delivered service derivates from correct service“*
- Attribute
  - *Availability - „Readyness for correct service“*
  - *Reliability - „Continuity of correct service“*
  - *Confidentiality - „Abscence of unauthorised disclosure of information“*
  - ...
- Maßnahmen zur Sicherstellung der Attribute

Modell von Jonsson und Olovsson (1992)

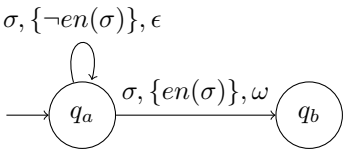


→ *Fault introduction*
- unabsichtlich, Bedienfehler (**User**)
- absichtlich, Ausnutzung von Schwachstellen (**Non-User**)
- ohne direkten Bezug, kosmische Strahlung (**Environment**)

→ *Delivery of Service*
- gegenüber authorisierten Nutzern (**User**)
- *Availability/Reliability*

→ *Denial of Service*
- gegenüber unauthorisierten Nutzern (**Non-User**)
- *Confidentiality*

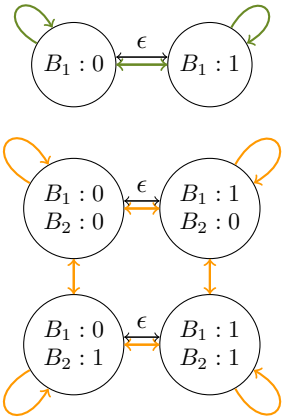
Beschreibung eines *Systems* und dessen *Service*



- $\mathcal{A} = (Q, \Sigma, \Omega, \delta, \lambda, q_0)$
  - Ausgabe am Zustandsübergang
  - keine akzeptierenden Zustände
- Guards[1] an Zustandsübergängen
  - binäre Entscheidung über Auswirkungen von $\sigma \in \Sigma$
  - Annahme: Ausgabe $\omega \in \Omega$ enthält Informationen, ob erfolgreich oder nicht
- Sprache $\mathcal{L}$ des Automaten:
  - Folgen $(\Sigma \times \Omega)^*$, ergeben sich aus Ausführung
  - Beschreibung von Äquivalenz
  - *Service* des Systems

[1] Dijkstra: Guarded Commands, Nondeterminacy and Formal Derivation of Programs (1975)

Spezifikation: *Service* aus Nutzersicht (Beispiel)

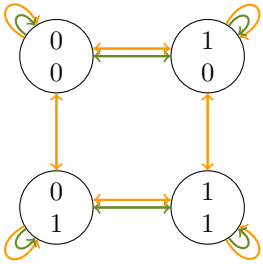


- V : Variablenmenge des Systems
  - $V_R \subset V$: Berechtigungen des Nutzers
- Q : mögliche Belegungen der Elemente aus V
- Σ : Operationen auf V
  - $\Sigma_R \subset \Sigma$: Ändern der Berechtigungen
  - Guards zur Umsetzung der Berechtigungen
  - ϵ-Übergänge für Nebenläufigkeit, keine Ausgabe
- Ω : Abbildung des Zustands auf Variable und Valuation

Spezifikation $\mathcal{A}_S$: *Service* aus Systemsicht (Beispiel)

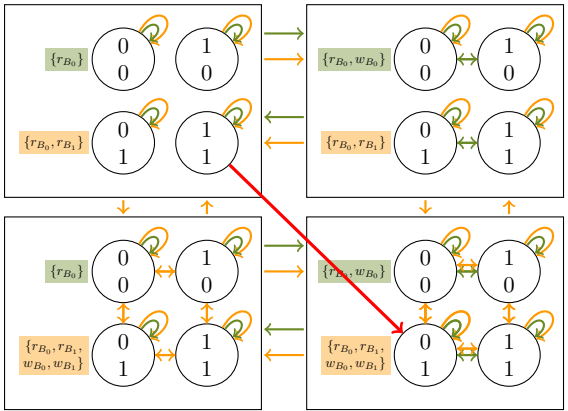
$u_1 = \{r_{B_0}\}$

$u_2 = \{r_{B_0}, r_{B_1}\}$



- Komposition zu Gesamtspezifikation
  - gleicher Name $\Rightarrow$ selbes Objekt
  - keine ϵ-Übergänge
  - Erweiterung von Σ und Ω zu Vektoren als Ein- und Ausgabekanäle für Nutzer
- Annahme: Spezifikation beschreibt den *Service* korrekt.

Implementation $\mathcal{A}_I$: Ursprung von *Faults*



- Zusätzlicher Automat zur Darstellung von *Faults*
- *Faults*: Eingaben $\Sigma' \cup \Sigma$:
  - extern: neue Eingaben σ_F
  - intern: fehlende Eingaben, falsche Zustandsüberführung oder Ausgabe
- *Error*: Resultat einer *Fault*-Eingabe
- *Failure*: Abweichung als Ausgabe an den Nutzer sichtbar

Operationalisierung von *Delivery* und *Denial of Service*

$$\exists\ \alpha \in \mathcal{L}(\mathcal{A}_S), \beta \in \mathcal{L}(\mathcal{A}_I) :$$
$$\pi_\Sigma(\alpha) = \pi_\Sigma(\beta), \pi_\Omega(\alpha) \neq \pi_\Omega(\beta)$$
$$\omega_i \in \pi_\Omega(\alpha) \neq \omega_i' \in \pi_\Omega(\beta)$$

- *Faults* durch $\Sigma' \cup \Sigma$ als Eingabealphabet an beide Automaten: $\sigma_j, j \leq i$.
- *Failure* entsteht durch: $\omega_i \neq \omega_i'$
- Verletzung von *Delivery of Service*: $\pi_u(\omega_i) \neq \epsilon, \pi_u(\omega_i') = \epsilon$
- Verletzung von *Denial of Service*: $\pi_u(\omega_i) = \epsilon, \pi_u(\omega_i') \neq \epsilon$

Interpretation der Attribute

- *Availability* und *Reliability*
  - Folgt aus *Failure* bzgl. *Delivery of Service*
  - Verhalten aus Sicht des **Users** gestört
  - Für *Reliability* → Zeitmaß nötig
- *Confidentiality* als illegale Ausgaben
  - Folgt aus *Failure* bzgl. *Denial of Service*
  - Nutzung von Operationen ohne Berechtigung abbildbar (**Non-User**)
  - Copy & Paste-Semantik über geteilte Variablen nur komplex darstellbar → Prozesskalküle
- *Integrity*
  - Befähigung bzw. Erlaubnis des Nutzers zur Durchführung von Operationen
  - *Fault prevention* am Fehlerautomaten