Praktikumsbeschreibung

Überblick:

Patientenmonitore sind technisch hoch-komplexe Systeme. Deren (z.T. sicherheits-kritischen) Funktionen hatten wir bereits im funktionalen Viewpoint kennengelernt und diese auch modelliert. Eines dieser Funktionen ist das *Alarming*, welches bei kritischen Veränderungen der Körpertemperatur des Patienten diese besonders hervorhebt.

Um das System weiter zu „engineeren“, lernen wir in diesem Praktikum eine *logische Architektur* des Systems zu erstellen. Die logische Architektur teilt das System in die zwei logischen Subsysteme auf: Den **Vitals Processor** und das **Display** auf. Wir werden für diese logischen Subsysteme zudem Verhalten modellieren und dann auch im Werkzeug Cameo simulieren. Die Simulation zeigt dann die Alarmfunktionalität.

Technische Realisierung des Patientenmonitorsystems der Firma Philips

Praktikum 4

In diesem Versuch modellieren Sie im Werkzeug *Cameo Systems Modeler* eine logische Architektur und Verhalten, welche Teile des Systems darstellen und simulieren.

Ziel:

Die Erstellung eines logischen Architekturmodell (Struktur- und Verhaltensmodell) des Patientenmonitors, dass eine Funktionalität der Funktionsarchitektur, nämlich die Alarmfunktionalität, darstellt. Dieses Modell kann simuliert werden, um zu überprüfen, ob die geforderte Funktionalität richtig modelliert wurde.

Idee:

Der **Patientenmonitor** („Patient Monitor“ in der Vorgabe) selbst soll aus zwei (logischen) Sub-Systemen bestehen (vgl. Abbildung oben):

1. Den sogenannten **„Vitals Processor“**, welcher rohe Messwerte einschließt und diese auf Auffälligkeiten überprüft. Das Modell, dass wir in diesem Praktikum vom **Vitals Processor** erstellen, soll das folgende Verhalten simulieren:
    a. Einlesen der Temperatur die vom Patienten kommt.
    b. Senden eines Alarmsignals, wenn die Temperatur den Grenzwert über- oder unterschreitet.
2. Das **Display**: In diesem Praktikum soll es eine Farbeinstellung haben, die den Alarmzustand widerspiegelt. Wir erwarten folgendes Verhalten:
    a. Reagiert auf Alarmsignale vom **Vitals Processor**.
    b. Im Alarm-Zustand ist die Farbe rot, sonst ist sie blau.

Im *operationalen Kontext* des Patientenmonitors steht der **Patient,** der regelmäßig *Signale* zu Verfügung stellt, welche Daten zur Körpertemperatur beinhalten. Dieses ist über das Property **temperature** modelliert.

Aufgabe 1: Strukturmodellierung der Logischen Architektur

Laden Sie sich die Vorlage aus dem ILIAS-Ordner herunter und öffnen Sie diese. Schauen Sie in den Ordner für die Logische Architektur und öffnen Sie das BDD. Führen Sie folgende Schritte aus:

- Dekomponieren Sie die logische Komponente: **Patienten Monitor** in die oben genannten Subsysteme (ähnlich wie bereits im Praktikum der Funktionsarchitektur geschehen).
- Führen Sie auch eine Dekomposition der Ports durch. Das **TemperatureSignal** des Patienten soll dem **Vitals Processor** zur Verfügung gestellt werden.
- Führen Sie neue Ports zwischen den Sub-Systemen ein, die später Alarmsignale senden bzw. empfangen sollen und verbinden diese in den notwendigen IBDs.

Aufgabe 2: Verhaltensmodellierung des Displays

Modellieren Sie das Verhalten vom **Display**:

- Erstellen Sie ein *State Machine Diagram* für das **Display**.
- Definieren Sie geeignete Zustände in dieser State-Machine für das angedachte Verhalten (s.o.)
- Ziehen Sie sinnvolle Transitionen, vorerst unbeschriftet, zwischen den Zuständen.
- Spezifizieren Sie nun Signal-Trigger an Transitionen.
- Simulieren Sie das **Display**. Schauen Sie sich an, wie ihr Automat auf Ihre Signale reagiert.

Aufgabe 3: Vorbereitung zur Datenübertragung zw. log. Komponenten

- Legen Sie ein neues SysML-Packages mit dem Namen „**Interfaces**“ an.
- Erstellen Sie darin ein BDD mit dem gleichen Namen
- Ziehen Sie alle Signale, die sie bisher (für bspw. die Verhaltensmodellierung der Alarmsignale angelegt haben) in das Package und in das Diagramm.
- Erstellen Sie die nötigen *Interface Blocks* (inkl. der *Flow Properties* für den Alarm-„Fluss“)
- Typisieren Sie alle Ports der logischen Architektur. Typisieren Sie auch alle Ports des Kontexts; Die Zugehörigen Interface-Blöcke sind gegeben. Denken Sie daran, dass die Richtung vom „Flow“ richtig eingestellt ist (Property im Proxy-Port: *isConjugated*).
- Erstellen Sie ein IBD (wo nötig), um alle notwendigen Ports entsprechend zu verbinden.
- Erstellen Sie einen „Item Flow“, indem Sie das passende Signal auf die Verbindung ziehen.

Aufgabe 4: Erweiterung

- Erstellen Sie auch für den **Vitals Processor** ein Verhalten (*State Machine Diagram*), der auf die Temperatursignale des Patienten wie folgt reagieren soll:
  - Wenn die Temperatur < 35°C oder >37.5°C soll ein Alarmsignal gesendet werden.
  - Wenn die Temperatur normal ist, soll kein Alarm angestellt sein.
  - Hinweis: Hier reicht ein einziger Zustand mit zwei Transitionen auf sich selbst.

Aufgabe 5:

Laden Sie Ihr Modell in den ILIAS-Abgabeordner hoch.