1 Übungsaufgabe 2: Konfigurierbares StuBSmI

Nachdem ihr in der vorherigen Übung erste Erfahrungen mit AspectC++ gesammelt habt, sollt ihr nun im Rahmen dieser Aufgabe eure StuBSmI-Implementierung mit einer konfigurierbaren Erweiterung in Form von Aspekten versehen.
Darüberhinaus wurde in der Vorlesung die Entwicklung von Software in Software-Produktlinien vorgestellt. Im zweiten Teil dieser Aufgabe sollt ihr eure bestehenden Implementierungsartefakte mit Hilfe von `Kconfig` als Software-Produktlinie darstellen und sie somit konfigurierbar machen.

1.1 Erweiterung des Aufrufstatistikmoduls

Zuerst sollt ihr das in der vorherigen Aufgabe entstandene Aufrufstatistikmodul so erweitern, dass nicht mehr pro Objektinstanz sondern stattdessen pro Memberfunktion (statisch, über alle Instanzen hinweg) gezählt wird.

1.2 Integration des Aufrufstatistikmoduls in StuBSmI

Das Aufrufstatistikmodul soll nun in StuBSmI integriert werden. Die Ausgabe soll der Kern periodisch in ein „Fenster“ auf dem CGA-Screen schreiben, dessen Größe und Position konfigurierbar sind. In diesem Fenster soll eine sortierte Ausgabe der n am häufigsten aufgerufenen Funktionen angezeigt werden. In der Konfiguration soll auch eine Auswahl zwischen einem absoluten Aufrufzähler und verschiedenen „Aufrufe-pro-Zeiteinheit“-Ausgaben angeboten werden, sowie ein periodisches Zurücksetzen der Zähler in einem einstellbaren Zeitintervall.
Es bietet sich an, den Statistikaspekt nicht in *alle* Funktionen einzuweben, sondern „uninteressante“ Funktionen wie beispielsweise die Ausgabeoperatoren zu ignorieren.

Hinweise:

- Bei der Verwendung von statischen, lokalen Variablen in Funktionen begegnet ihr möglichererweise unerwarteten Fehlermeldungen zu den Funktionen `__cxa_guard_{acquire,release}`. Ein Workaround dafür findet sich via Google auf `osdev.org` (auch auf der Übungsseite verlinkt).

1.3 Merkmalmodell für die Übungsaufgabe

Die unterschiedlichen Verhaltensweisen und Parameter der Implementierung sollt ihr nun in ein Merkmalmodell (*feature model*) überführen.
Die Umsetzung erfolgt in der Sprache des Linux-Konfigurationswerkzeugs `Kconfig`.

1.4 Das Linux-Konfigurationswerkzeug: `make menuconfig`

Die Entwicklung mit Produktlinien sieht vor, dass auch nicht-Domänenexperten spezifische Lösungen generieren können. Für den Linux-Kern wurde dazu ein spezielles Werkzeug, `Kconfig`, entwickelt, welches es den Benutzern erlaubt, den Kern zu konfigurieren. Dieses Werkzeug wird in den Linux Quellen mit den Kommandos `make menuconfig` aufgerufen. `Kconfig` stellt sicher, dass die vom Benutzer getroffene Auswahl im Variantenmodell gültig ist. Im Fall des Linux-Kerns werden dann aus dieser Auswahl einbindbare *Header*-Dateien und *Make*-Fragmente erzeugt, die dann den Build-Prozess so beeinflussen, dass die ausgewählte Variante erzeugen wird.
Diese Werkzeugkette sollt ihr nun auch für eure Lösung nutzen. Allerdings sind die durch `Kconfig` erzeugten Header alleine noch nicht sehr nützlich, da ihr eure variablen Features nicht mit Hilfe von C-Präprozessordirektiven implementiert habt. Ein Feature wird stattdessen durch Hinzufügen von Aspekten implementiert. Die durch KConfig beschriebenen Features müssen also noch auf Dateimengen abgebildet werden, die beim Übersetzungsvorgang berücksichtigt werden.
Hierzu stellen wir euch das Werkzeug `BST-config`. Damit könnt ihr ein Feature aus der Kconfig-Beschreibung durch Verwendung eines Familienmodells auf eine Menge von Implementierungsdateien abbilden. Diese Dateimenge, die zu einer Auswahl von Features gehört, kann nun mit Hilfe von `BST-config` in einen eigenen Ordner kopiert werden, in welchem ihr dann die konkrete Variante eurer Software übersetzen könnt.

→ Modelliert die soweit vorhandene und implementierte Variabilität in Form eines Variantenmodell als Merkmaldiagramm (am Besten auf Papier).

→ Implementiert dieses Modell mit dem Linux-Konfigurationsprogramm `Kconfig`.

→ Integriert das Linux-Konfigurationsprogramm in eure vorhandenen Makefiles.

→ Arbeitet euch in das zur Verfügung gestellte Konfigurationssystem `BST-config` ein, um mit Hilfe der mit `Kconfig` erzeugten Konfiguration ein Projektverzeichnis zu konstruieren, in dem nur die Aspekte zur Übersetzung verwendet werden, die für die selektierte Konfiguration relevant sind.

Hinweise:

- Eine bereits vorübersetzte ausführbare Version des Linux Konfigurationswerkzeugs Kconfig liegt in `/proj/i4bs/tools/bst-config/bin`. Er wird wie folgt gestartet: `mconf features.fm` und erzeugt/bin dann aus eurer Feature-Auswahl eine `.config` Datei.
- Ebenfalls unter `/proj/i4bs/tools/bst-config/bin` findet ihr das Werkzeug `transform.pl`. Mit diesem könnt ihr einzelne Features auf Dateimengen abbilden. Der Aufbau des Familienmodells, das die Abbildung von Features in Kconfig auf Dateien festlegt, und die Bedienung des Werkzeugs sind in `../doc/bst_transform.pdf` dokumentiert. Darüberhinaus sind dort auch zwei Beispiele zu finden, an denen ihr den Aufbau der Konfigurationsdateien und das Zusammenspiel mit Kconfig nachvollziehen könnt.

Abgabe: am 20.06.2013