Thema, Ziele: Algorithmische Schätzungen

Aufgabe 1: Rapid Prototyping

Für welche Themen ist Rapid Prototyping geeignet? Was sind die Vor- und Nachteile von Rapid Prototyping?

Wie lässt sich Rapid Prototyping in den Projektlebenszyklus einordnen und wie sieht die Projekthierarchie aus? Geben Sie auch ein geeignetes Vorgehensmodell an falls nötig.

Aufgabe 2: Function Point Analysis

Aufgrund den im letzten Praktikum erstellten groben Anforderungen mit Hilfe eines Use-Case Diagramms sollen nun ein Functional Point Analyse durchgeführt werden.

Wie viele Functions Points erhalten Sie aus für die Funktionalität der in unserem Projekt zu erstellenden Software?

Als Ausgangslage empfehle ich meinen Use Case Lösungsvorschlag vom letzten Praktikum zu verwenden, damit die Lösungen dieser Aufgabe nicht zu gross abweichen.

Zur Vereinheitlichung der Lösung sind hier die zu schätzenden Elemente aufgelistet:

- Datenelemente
  - ILF VoltageDividerWidget inkl. ESeries
    - 2 RETs
      - VoltageDividerWidget
      - ESeries
    - ? DETs:
      - U1,
      - U2,
      - ?,
      - aktuell ausgewählte Serie

- Transaktionselemente
  - Parameter konfigurieren (worst case Betrachtung mit E-Reihe setzen)
    - 2 FTRs
      - VoltageDividerWidget
      - ESeries
    - ? DETs:
      - ?
  - Widerstand berechnen
    - 2 FTR
      - VoltageDividerWidget
      - ESeries
    - ? DETs:
      - ?

Vorgehen:

1. Bestimmen Sie die fehlenden, mit einem Fragezeichen gekennzeichneten Elemente.
2. Bestimmen Sie die Komplexität der jeweiligen Daten- und Transaktionselemente.
3. Ermitteln Sie die Anzahl Function Points für die zu entwickelnde Anwendung.

Aufgabe 3: Stack GUI implementieren

Implementieren Sie mittels Qt ein Stack GUI. Die Problem-Domain ist bereits implementiert und ist im Vorlage Ordner „Vorlage/A1“ zu finden.

Zu implementierende GUI Darstellungen/Funktionalitäten sind

- *push*, *pop* und *peek* Buttons
- Eingabe einer Zahl als *integer* Datentyp
- Es kann davon ausgegangen werden, dass der User die Eingabe immer korrekt macht → keine Eingabeüberprüfung
- Information über vollen und leeren Stack (Zustände müssen ständig sichtbar sein)

Die *Stack* Klassenvorlage ist als Template implementiert. Der *moc* unterstützt derzeit keine Klassen-Templates in Zusammenhang mit *QObject* und *Signal/Slots*. Als Workaround wurden die *Signals/Slots* in die Basisklasse *StackSignals* der *Stack* Klasse verschoben werden.