CES Softwareentwicklungspraktikum

Analyse- und Entwurfsdokument

Lena Blum, Alexander Fischer und William Hulin

Matr.-Nr. 302253, 303979 und 293858
email:
[lena.blum|alexander.fischer|william.hulin]@rwth-aachen.de

Inhaltsverzeichnis

Kapitel 1

Vorwort

1.1 Aufgabenstellung und Struktur des Dokuments

Aufgabenstellung

Im Rahmen des Softwareentwicklungspraktikums (CES_SS2012) soll eine Software zur Simulation eines Stehaufkreisels erstellt werden. Die Simulationssoftware muss sowohl den reibungsfreien, als auch den reibungsbehafteten Fall korrekt simulieren knnen.
Als Programmiersprache soll C++ verwendet werden. Der Quellcode soll derart strukturiert und kommentiert sein, dass spätere Modifikationen und Erweiterungen durch Dritte möglich sind.

1.2 Projektmanagement

Protoyping (MATLAB/ FORTRAN)	Alexander
Dokumentation	Lena
Coding:	
Parameterset, Solver, Solution, Rkv56Parset, Rkv56, DESolution,<<interface>>RightSide, RHS, Rkv56Modified	Alexander
<<interface >>OutputInterface, OutputToolbox, Main, ExceptionHandlingModule, MathException, NonCriticalME, CriticalME, ParameterException	William
GUI	Lena

1.3 Lob und Kritik

Kapitel 2

Analyse

2.1 Anforderungsanalyse

2.1.1 Benutzeranforderungen

Das von Herrn Professor Gauger gestellte Simulationsproblem umfasst die Erstellung einer Software zur Simulation eines Stehaufkreisels.
Die Simulation muss sowohl den reibungsbehafteten, als auch reibungsfreien Fall korrekt simulieren.
Im Speziellen wird ein Runge-Kutta56-Verfahren mit adaptiver Schrittweitensteuerung unter Betrachtung einer Erhaltungsgre (*conserved quantity*) zur Simulation des Problems verwendet.
Wahrscheinlich wird das Rkv56 Verfahren durch ein BDF-Verfahren oder eine C++ Implementierung eines speziellen Krylow-Verfahrens ersetzt.
https://computation.llnl.gov/casc/software.html
Die Realisierung der Simulation findet in C++ statt.

Die Bedienung sowie das ausgeben der Simulationsergebnisse muss durch eine grafische Benutzeroberfläche (*GUI*) möglich sein.
Die Simulationsergebnisse können in einer *ASCII*-formatierten Datei zur weiteren Verarbeitung und Auswertung exportiert werden.
Durch den modularen Aufbau ist die Wartbarkeit und spätere Modifikationen oder Erweiterungen durch externe Mitarbeiter gewährleistet.
Das Kernproblem besteht im Lösen der Rechten Seite des folgenden Differentialgleichungssystems:

$$\ddot{\theta}(I + ma^2 \sin^2\theta + kma \sin\theta(R - a\cos\theta)(-\dot{x}_c \sin\phi + \dot{y}_c \cos\phi - (R - a\cos\theta)\dot{\theta}))$$
$$= \underbrace{-(I_3 - I)\dot{\phi}^2 \sin\theta\cos\theta}_{=0} - I_3\dot{\phi}\sin\theta\dot{\psi} + (g + a\dot{\theta}^2\cos\theta)(-ma\sin\theta - km(R - a\cos\theta)$$
$$(-\dot{x}_c \sin\phi + \dot{y}_c \cos\phi - (R - a\cos\theta)\dot{\theta}))$$

$$\ddot{\phi} I \sin\theta = -\underbrace{(2I - I_3)}_{=I}\dot{\phi}\dot{\theta}\cos\theta + I_3\dot{\theta}\dot{\psi}$$
$$- km(g + a\cos\theta\dot{\theta}^2 + a\sin\theta\ddot{\theta})(a - R\cos\theta)(\dot{x}_c\cos\phi + \dot{y}_c\sin\phi + (a\dot{\phi} + \dot{\psi}R)\sin\theta)$$

$$\ddot{\psi} I_3 = -I_3(\ddot{\phi}\cos\theta - \dot{\phi}\dot{\theta}\sin\theta)$$
$$- km(g + a\cos\theta\dot{\theta}^2 + a\sin\theta\ddot{\theta})(R\sin\theta)(\dot{x}_c\cos\phi + \dot{y}_c\sin\phi + (a\dot{\phi} + \dot{\psi}R)\sin\theta)$$

$$m\ddot{x}_c = -km(g + a\cos\theta\dot{\theta}^2 + a\sin\theta\ddot{\theta})(\dot{x}_c + (a\dot{\phi} + \dot{\psi}R)\sin\theta\cos\phi + (a\cos\theta - R)\sin\phi\dot{\theta})$$

$$m\ddot{y}_c = -km(g + a\cos\theta\dot{\theta}^2 + a\sin\theta\ddot{\theta})(\dot{y}_c + (a\dot{\phi} + \dot{\psi}R)\sin\theta\cos\phi + (R - a\cos\theta)\cos\phi\dot{\theta})$$

2.1.2 Anwendungsfallanalyse

Anwendungsfälle (Statik: Anwendungsfalldiagramme; Dynamik: Aktivitätsdiagramme; Textuelle Beschreibungen laut Vorlage

Systemanforderungen

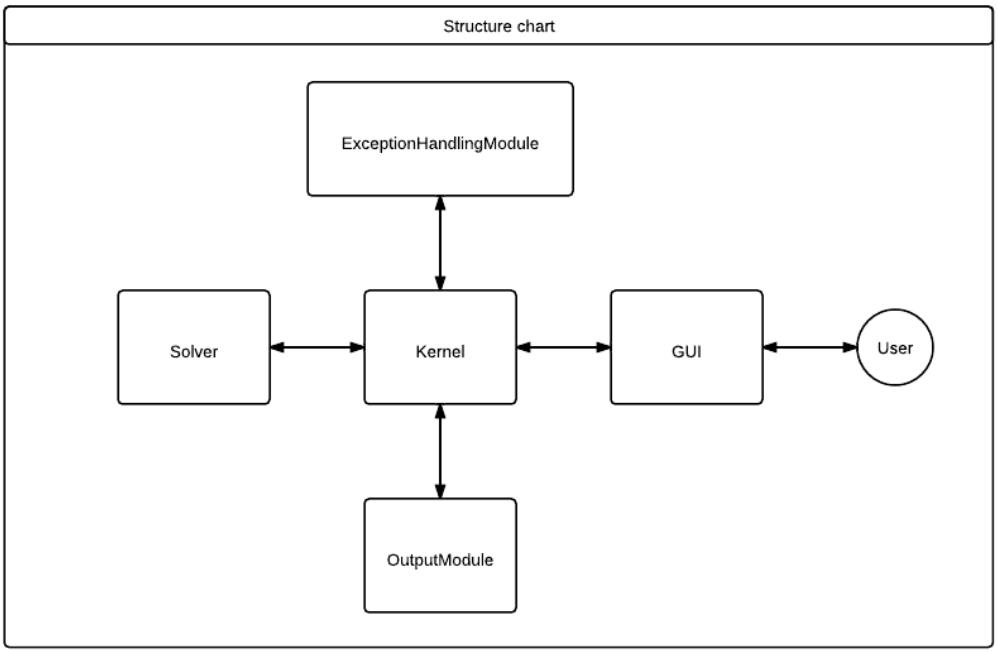
Dem Anwender ist es mglich die Simulationsparameter k (Reibung) sowie $\dot{\psi}$ (rad/s) über eine grafische Eingabemaske festzulegen. Wenn während der Simulation ein Fehler auftritt wird der Anwender über ein Popup-Fenster benachrichtigt. Nach Durchlauf der Simulation bekommt der Anwender die Simulationsergebnisse - $\theta, \psi, \phi, x_c, y_c, \dot{\theta}, \dot{\psi}, \phi$ - in Form von *LineCharts* in eine *GUI* eingebettet angezeigt.
Die auf der *GUI* ausgegebenen Plots können als Bilddatei exportiert werden, ebenso besteht für den Anwender die Möglichkeit, die errechneten Werte in Form von Tabellen zu speichern.

Kommt es während der Laufzeit zu einem kritischen Fehler (ein Fehler, der das korrekte Fortführen des Programmes undmöglich macht) wird der Anwender über ein Popup-Fenster benachrichtigt und das Programm beendet.

Kapitel 3

Entwurf

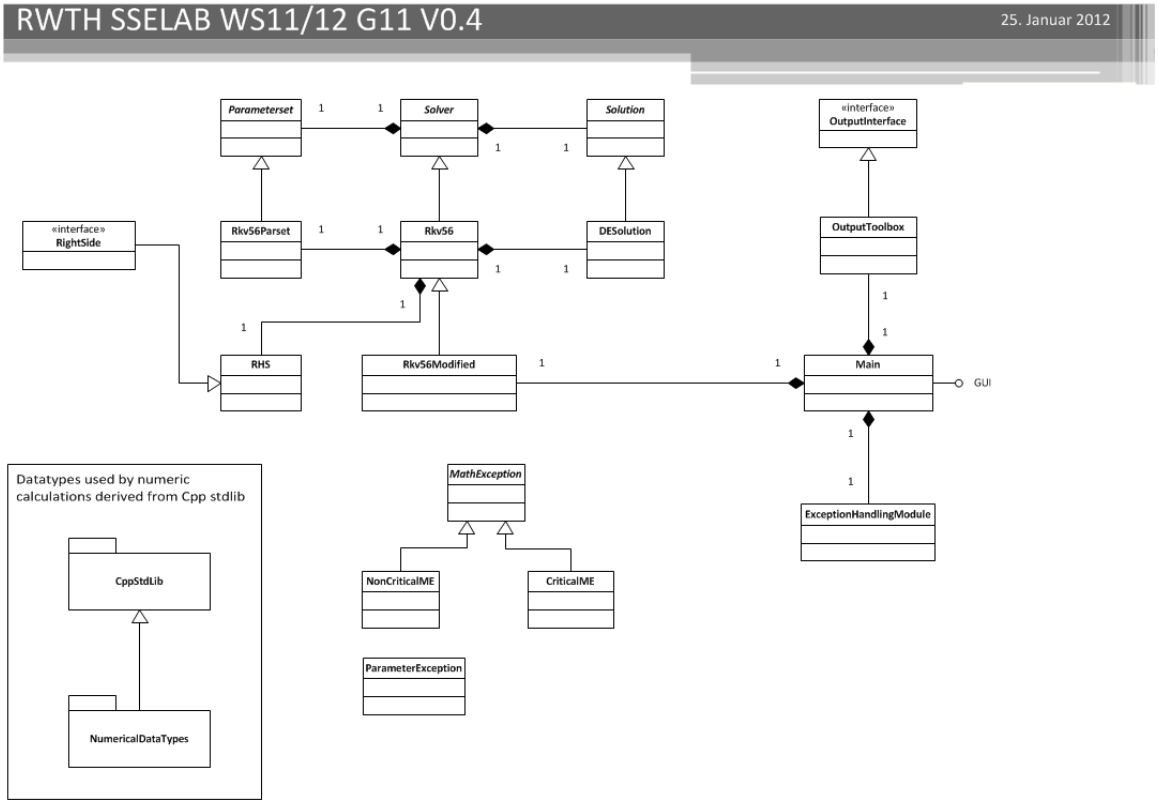
3.1 Grobentwurf: Subsysteme



3.1.1 Statik

3.1.2 Dynamik

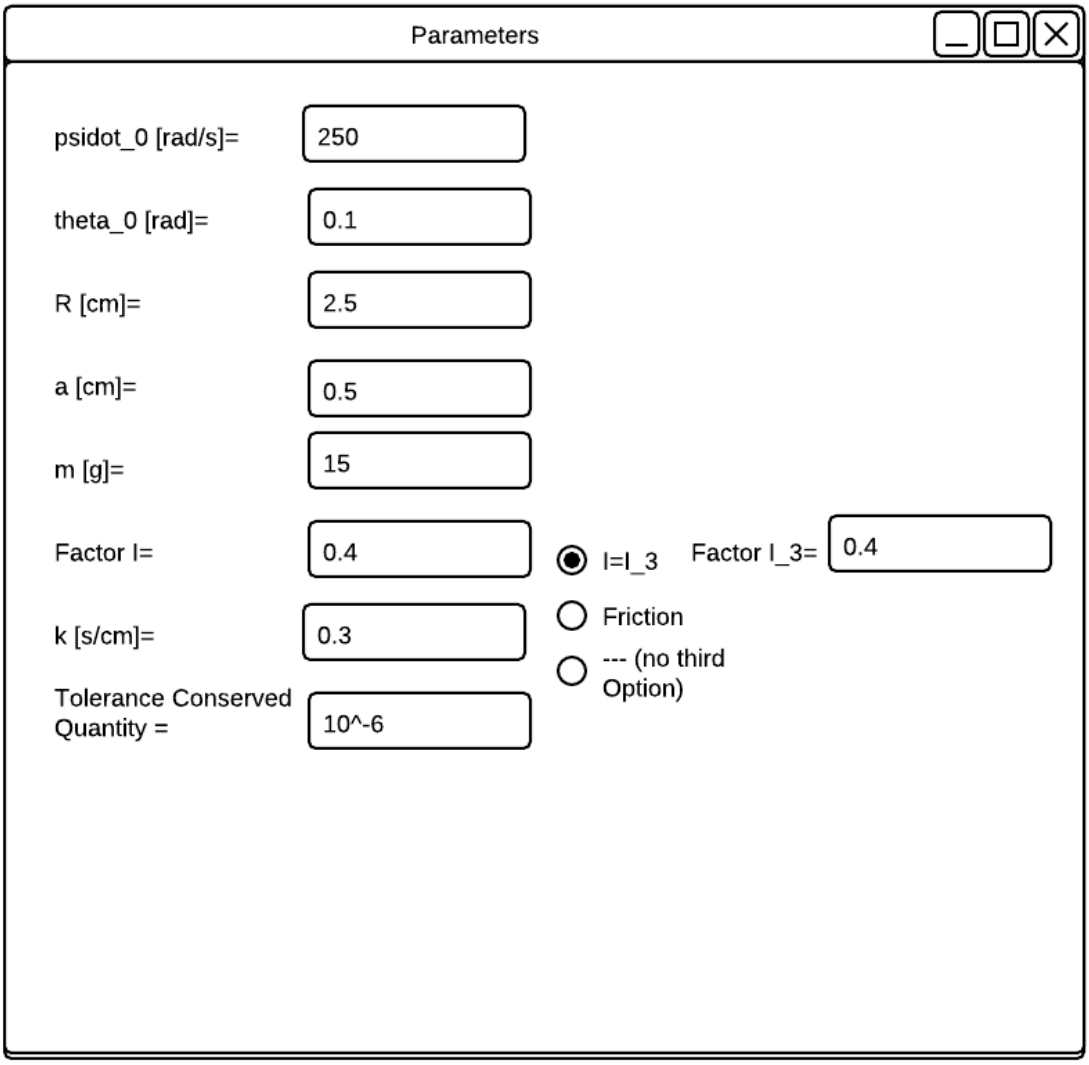
3.2 Detailentwurf: Klassen

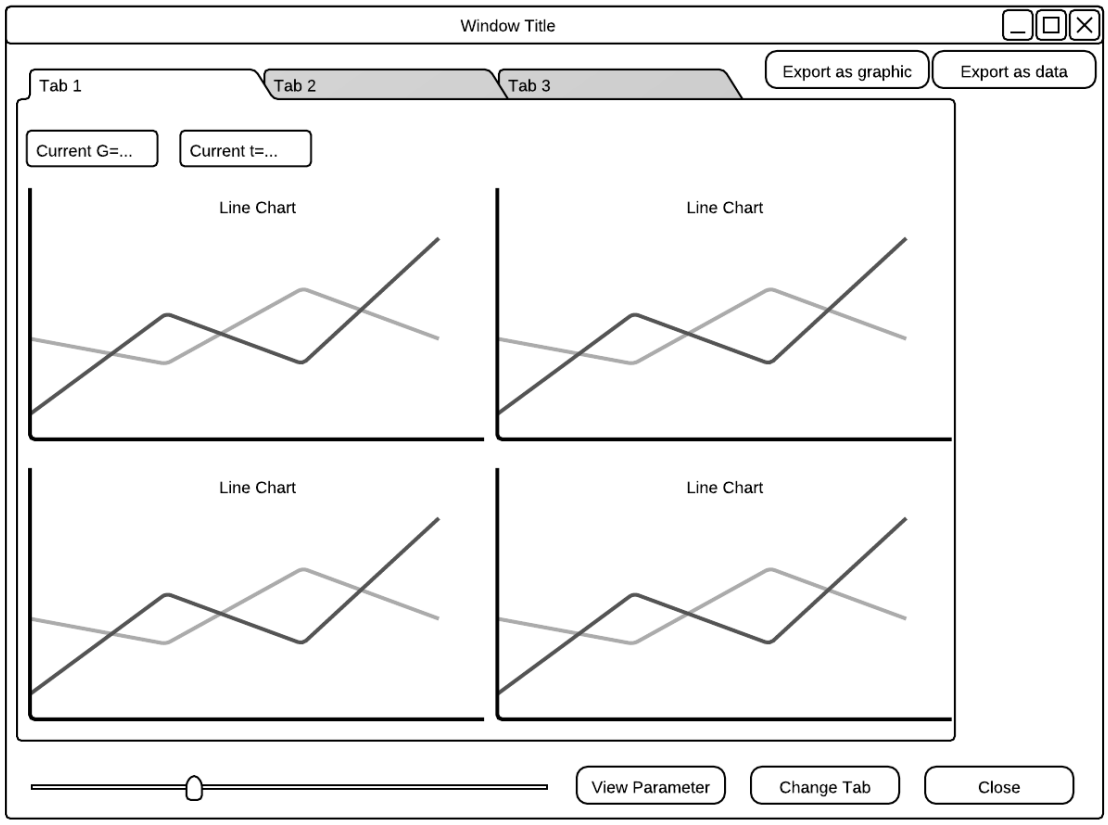


3.2.1 Statik

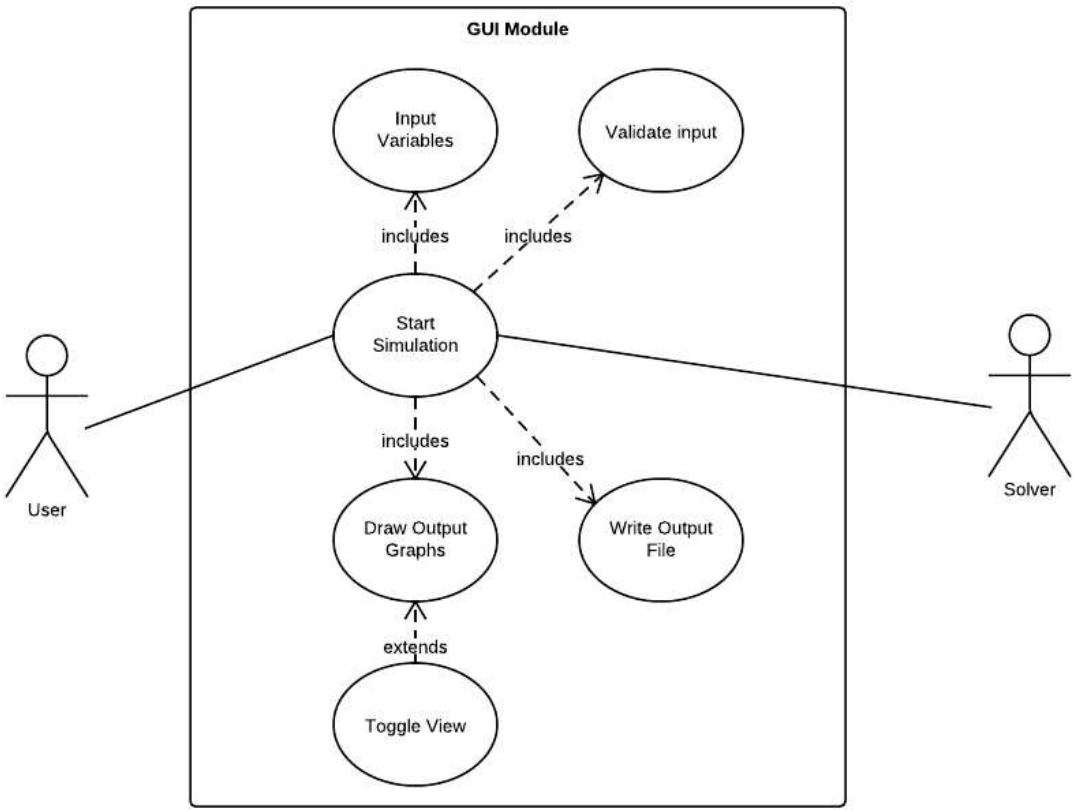
3.2.2 Dynamik

3.3 Graphical User Interface




Window Title
Tab 1
Tab 2
Tab 3
Export as graphic
Export as data
Current G=...
Current t=...
Line Chart
Line Chart
Line Chart
Line Chart
View Parameter
Change Tab
Close

3.4 Use-Case-Diagramm



Kapitel 4

Benutzerdokumentation

Kapitel 5

Entwicklerdokumentation

Anhang A

Quellcode