# Review SimuCom Plattform

SimuCom ist das Simulationsframework des PCM. Es bemüht sich die Konzepte das Meta-Modells möglichst genau so abzubilden, wie sie definiert sind. Es unterteilt sich in zwei Teile: eine Modell-2-Text Transformation, die konkrete Instanzen des PCM in Code umwandelt, der den zweiten Teil konfiguriert: die SimuCom Plattform. Diese Plattform bietet alle Dienste, die generisch sind, und die nicht vom Generator erstellt werden müssen. Damit folgt das Design dem von M.Völter vorgeschlagenen MDD Vorgehen mit Plattform und Generat. Außerdem wurden einige der Patterns eingesetzt, die Generat und Plattform koppeln sollen. So gibt es z.B. Template-Methods, die vom Generator gefüllt werden und abstrakte Oberklassen, die das Generat beerbt und mit dem restlichen Methoden auffüllt.

Die Plattform teilt sich auf in zwei Eclipse-Plugins:

1. simucom.variables: Enthält alle Teile der Plattform, die sich mit der Simulation der Parametercharactersierungen von Heiko befassen. Das Plugin wird auch vom Prototype Mapping genutzt, das eine „Simulation“ auf echter Hardware durchführt.
2. Simucom: Der eigentliche Kern der PCM Simulation. Enthält Klassen für simulierte Ressourcen, für Scheduler, für Workload Treiber, usw.

Simucom.variables benutzt als Kernalgorithmus s.g. StackFrames. Die Idee stammt aus dem Compilerbau und wird zur Realisierung von Scopes aus Programmiersprachen benutzt. Scopes sidn dabei die Sichtbarkeitsbereiche von Variablen, in Java und C also z.B. alles zwischen zwei geschweiften Klammern. Der Compiler generiert nun für jeden Scope ein Stackframe, dass ein Speicherblock auf dem Stack ist, der die lokalen Variablen des aktuellen Scopes enthält. Weiterhin enthält er auch einen Pointer auf seinen Vaterstackframe, denn die Variablen des umliegenden Frames sind zusätzlich zu den Variablen des Frames zugänglich.

Diese Frames werden nun, genau wie im Compilerbau benutzt, um Heikos Characterisierungen umzusetzten. Alle Werte von Characterisierungen eines Scopes liegen im aktuellen Stackframe.

Wichtig im variables Packet ist der PCMStoExEvaluationVisitor, der von der Simulation benutzt wird, um den AST eines beliebigen StoEx Ausdrucks zu durchlaufen und anhand des aktuellen Stackframes auszuwerten. Steht also z.B. im Stackframe unter a.BYTESIZE der Wert 10, so ergibt die Auswertung von 100\*a.BYTESIZE dann 1000. Dabei prüft der Visitor die Typen der Ausdrücke auf Konformität. Der von Heiko eingeführte ANY Typ entspricht dabei dem Java Object Typ, bei dem erst zur Laufzeit ermittelt wird, welcher Typ tatsächlich vorliegt. Ansonsten entspricht der EvaluationVisitor dem üblichen Expression Evaluieren aus dem Compilerbau.

Simucom bietet im Paket resources Klassen für die Simulation von Resourcen mit drei Scheduling Strategien: ProcessorSharing, ein RoundRobin ohne Zeiten für Kontextswitch und mit unendlich kleinen Zeitscheiben, FIFO und DELAY. Im package usage sind die Basisklassen der Workload-Treiber mit ihren jeweiligen Template-Methoden zu finden. Im Basis-Paket findet sich die abstrakte Oberklasse einer jeden Simulation. SimuCom.model enthält das Simulationsmodell, wie es für Desmo-J nötig ist, um den aktuellen State der Simulation zu halten. SimuCom.sensor enthält das ursprüngliche Simulations-Sensorframework. Dieses wurde bisher nicht ausgetauscht, sondern vielmehr über seine Observer zugänglich gemacht. Als Observer hängt dann im Moment die aktuelle Fassung

Fassung des Sensorframeworks daran. Damit konnte eine Migration des Codes bisher verschoben werden, um später im Sinne einer sanften Migration diesen Teil auszutauschen.

Ziel des Reviews ist es nun, diese beiden Plugins zu untersuchen. Weitere Dokumentation findet sich in den jeweiligen Klassen, Testfälle (wenn auch noch nicht in ausreichender Menge) finden sich immerhin in simucom.variables.tests für den StoExVisitor.