

Modellaustausch mit SMI

Ronald Ruzicka
Simutech
Atzgersdorferstraße 32
A-1230 Wien

Kurzfassung: Dieser Beitrag zeigt den heutigen Stand der Modellaustausch- und Modellkopplungsproblematik auf und entwickelt daraus eine Diskussion der Prinzipien, der Vor- und Nachteile des Modellkopplungswerkzeugs SMI im Vergleich zum Konzept von VHDL-A. Ein Beispiel demonstriert die Möglichkeiten von SMI in der Praxis.

Abstract: *This contribution gives a glimpse onto the current state of model exchanging and model coupling. It shows a discussion of the principles, advantages and disadvantages of the model coupling tool SMI compared to the concepts of VHDL-A. An example demonstrates the features of SMI.*

1. Motivation

Die Spezialisierung der Fachleute im Bereich Simulation und Simulationsanwendungen, die Komplexität von zu simulierenden Vorgängen und der allgemeine Trend zu vernetztem Denken überhaupt führte in den letzten Jahren dazu, dass Simulationsstudien in der überwiegenden Zahl der Fälle heutzutage einerseits interdisziplinär, andererseits in Arbeitsgruppen erfolgen müssen.

Diese Arbeitsgruppen erstrecken sich in vielen Bereichen auch weit über Firmengrenzen hinaus: etwa die Mitarbeiter der Herstellerfirma einer Steuerung und jene eines Betriebes, der diese Steuerung in eigene Produkte inkludiert. Zwischen den Mitgliedern dieser (nur mehr im weitesten Sinne) „Arbeitsgruppen“ werden Simulationsteilmodelle und entsprechende Dokumentation zu Kommunikationsmitteln, denn die Zeit für intensivere Gespräche ist nicht vorhanden. Als Kommunikationsmittel aber sind die Modelle wesentliche Bausteine des Arbeits- und Produktionsprozesses.

Andererseits stellen die Teilmodelle auch Teile des Auftrags dar. So werden z.B. gewisse Autobauteile nur mehr gemeinsam mit Simulationsmodellen ausgeliefert (meist auf Kundenwunsch).

Beim Anlagenbau etwa ist dies schon länger Usus: Simulationen werden aufgrund der Originalplanungsunterlagen (CAD) erstellt und gemeinsam mit diesen zur Beurteilung der Planungsentscheidungen dem Kunden vorgelegt.

Aus all diesen Argumenten lässt sich der allgemeine Bedarf an Werkzeugen zur Integration von Modellen unterschiedlicher Herkunft erklären.

Modellaustausch ist nun prinzipiell kein Problem: meist kann der einigermaßen erfahrene Simulierende auch Modelle ihm unbekannter Simulatoren verstehen und deuten. Die

Probleme entstehen erst, wenn er versucht die Modelle zu simulieren, mit eigenen zu integrieren und vorliegende Simulationsergebnisse zu reproduzieren.

2. Modellkopplung versus Simulatorkopplung

Prinzipiell bieten sich zwei Wege der Simulation nach erfolgtem Modellaustausch an: Simulatorkopplung und Modellkopplung.

Unter Simulatorkopplung versteht man die - meist Online, d.h. zeitlich parallel ablaufende - Verschaltung von Simulationsmodellen samt Rechenvorschriften (Algorithmen) und -mechanismen. Dies bedeutet, dass i.a. hier bereits bestehende *Simulatoren* miteinander verschaltet werden (siehe etwa [NIEM91], [SCHW91]).

Das Konzept der Simulatorkopplung wirft für den allgemeinen Fall in der Praxis große Probleme auf, ähnlich den Problemen bei der parallelen Simulation [RUZI93]: die Synchronisation der Simulatoren, das Fehlen geeigneter paralleler Algorithmen zur Integration, System-inherente numerische Fehler [RUZI96].

Deshalb geht der Trend in Richtung Modellkopplung.

Modellkopplung bedeutet, dass Modelle unterschiedlicher Herkunft, also etwa mithilfe *unterschiedlicher* Simulatoren entwickelt, zu einem Gesamtmodell verwoben werden. Dieses Gesamtmodell wird dann unter Zuhilfenahme *eines* Simulators (mit einheitlichen Algorithmen) simuliert.

Die Probleme bei der Modellkopplung bereits vorliegender Modelle liegen darin, dass aufgrund der unterschiedlichen Herkunft meist eine Übersetzung der Modelle vorzunehmen ist. Der Vorteil wiederum ist, dass der Kopplungs-overhead - in diesem Fall also die Übersetzung - schon weitgehend Offline, also vor der Simulation, durchgeführt werden kann (siehe Bild 1).

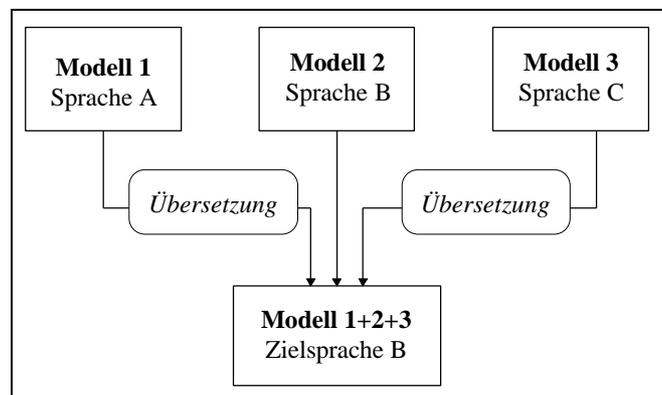


Bild 1: Das Prinzip der Modellkopplung

3. Modellaustausch

Modellaustausch ist sicher dann am einfachsten möglich, wenn die einzelnen Teilmodelle in ein und derselben Modellumgebung entwickelt wurden. Solche Umgebungen müssen sehr umfangreiche Fähigkeiten aufweisen, um alle Teilbereiche einer interdisziplinären Simulation behandeln zu können.

Proprietäre Systeme (z.B. ACSL [MGAI91], Simulink [MATH93], Spice) sind - trotz aller Bemühungen der Hersteller, die in immer neuen Add-Ons und Toolboxes gipfeln, - jeweils nur in Teilbereichen „gut“.

Eine bestechende Idee war nun, aus den Fehlern bisheriger Simulationssysteme zu lernen und eine neue Simulationssprache zu entwickeln, die alle Bedürfnisse moderner Simulationsmodelle abdeckt. Im Wesentlichen sind hier zwei Ansätze zu erwähnen:

DSblock, entwickelt von der DLR, stellt eine neue Modellierungssprache zur Verfügung, die - mit entsprechenden Übersetzern aus einigen anderen Simulationssprachen ausgestattet - eine Gesamtsimulation ermöglicht [OTTE95]. DSblock ist jedenfalls erneut ein proprietärer Ansatz.

Das hingegen nicht proprietäre *VHDL-A* (AHDL) Konzept [IEE93] macht sich die hohe Installationsbasis der VHDL Simulatoren im Bereich Elektronik zu nutze und erweitert VHDL in obigem Sinne, in einer genormten Ausprägung! Bezugnehmend auf Bild 1 wäre VHDL-A eine Sprache B, erweitert um diverse Fähigkeiten, also etwa ein B'.

4. SMI

Einen anderen Weg beschreitet *SMI*, das Simulation Module Interface [RUZI96], auch als *ComponentWare for Simulation* bezeichnet. Dieses System stellt eine offene Schnittstellenbeschreibung zur Verfügung, wie ein sogenanntes *SMI-Modul* auszusehen hat (Bild 2), bzw. welche Informationen es nach außen weitergeben muss.

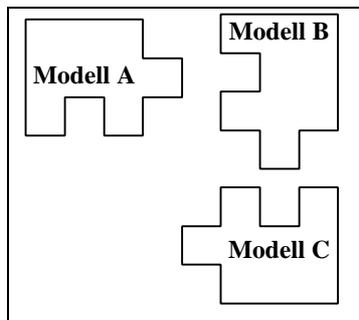


Bild 2:
Teilmodelle als SMI-Module

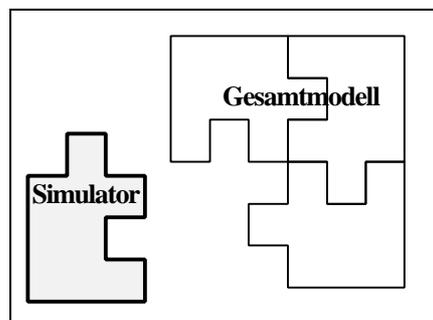


Bild 3:
Gemeinsame Simulation von Modulen in SMI

Ein SMI-Modul ist im Konzept *SMI Step 1* eine fertig übersetzte Laufzeitbibliothek (unter MS-Windows also eine DLL) mit genormten exportierten Funktionsaufrufen. Diese stellen Informationen über das Modell, wie Konstanten, Zustandsgrößen, Dimensionen, Eingänge, Ausgänge, Gleichungen (soweit möglich: in sortierbarer Form!), Ereignissen und anderes zur Verfügung.

Idealerweise kann jedes Simulationsmodell in ein SMI-Modul umgewandelt werden und jeder SMI fähige Simulator diese Module simulieren (siehe Bild 3).

Die Module enthalten keine Rechenalgorithmen und sind somit dem Zielsimulator in derselben numerischen Offenheit (als simple Gleichungen) zugänglich, so als hätte man die Teilmodelle in einer gemeinsamen Zielsprache (z.B. VHDL-A) implementiert.

5. VHDL-A und SMI: Vor- und Nachteile

SMI ist insbesondere dann sehr praktisch, wenn Modelle schon codiert (übersetzt) vorliegen, etwa als MEX-Files von Simulink. Andererseits kann sich der Modellaustausch auf ausführbare Module beschränken, was ein hohes Maß an Investitionsschutz bietet, da die Modelltexte nicht notwendigerweise mitgeliefert werden müssen.

Von Vorteil ist hierbei auch, dass keine neue Simulationssprache erlernt werden muss, da die Module einfach auf informatischem Wege (d.h. automatisch) in SMI-Module verwandelt werden und auch bereits vorhandene Simulatoren als Zielsystem verwendet werden können (soweit diese SMI unterstützen; alle Simulatoren mit offenen C-Schnittstellen können als SMI-Simulatoren fungieren, z.B. SIMUL_R, Simulink, MatrixX/SystemBuild).

VHDL-A weist gerade aus diesen Gründen und aufgrund seiner Komplexität, den Anspruch auf Vollständigkeit mit sich bringt, Defizite in der Benutzerakzeptanz auf; dies eben nicht zuletzt auch wegen der Notwendigkeit für den Benutzer, eine neue Sprache erlernen zu müssen. Die Weiterverwendung bereits vorhandener Modelle und die Einbindung von Hochsprachen (C, Fortran, Ada) stellen weitere, noch nicht zufriedenstellend gelöste Probleme dar.

Des weiteren ist es für die Werkzeughersteller um einiges schwieriger, Übersetzer von vorgegebenen Simulationssprachen in VHDL-A zu erstellen, als fertige Teilmodelle für den Benutzer transparent in ein DLL-Mäntelchen zu verpacken und damit in ein SMI-Modul umzuwandeln. Die Schwierigkeit für SMI-Konverterhersteller liegt darin, dass für fertig compilierte Modelle einer Simulationssprache oft nur unzureichend Dokumentation vorhanden ist, bzw. Hilfestellungen seitens der Simulatorhersteller kaum zu bekommen sind.

Einen Nachteil von VHDL-A versucht man neuerdings dadurch zu verringern, dass man den Sprachumfang reduziert („VHDL-A lite“) - und dabei allerdings auch an Universalität einbüßt. Auch die vorgesehene Möglichkeit der Verschlüsselung bietet einen Ansatzpunkt zur Geheimhaltung von Quelltexten.

6. Ein Beispiel

Als Beispiel sei hier eine Anwendung erwähnt, bei der ein in SIMUL_R (CSSL-Sprache, siehe [RUZI88]) bereits vorliegendes Modell einer Modellstrecke mit einem speziellen in Simulink entwickelten Regler verknüpft wird.

Als Zielsimulationsprache wurde hier wieder SIMUL_R verwendet, da es sich gut zur SMI-Simulation eignet. Alles läuft unter MS-Windows ab.

Die Vorgangsweise (mit Symbolbildern) ist in Bild 4 beschrieben. Die beiden Übersetzungsaktionen laufen für den Simulierer völlig transparent ab und beschränken sich auf das Starten des jeweiligen Konverters unter Angabe der Eingabedatein.

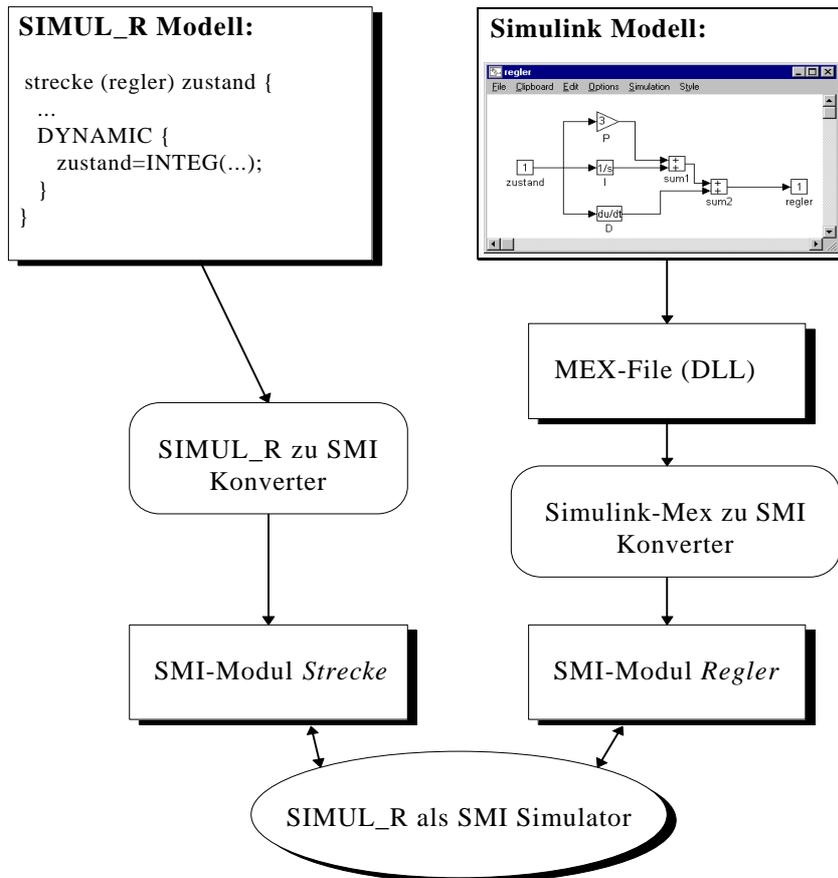


Bild 4: Vorgangsweise zur SMI Modellkopplung.

7. Conclusio

Mit VHDL-A und SMI liegen zwei prinzipiell unterschiedliche Lösungen mit derselben Lösungskompetenz zum in der Basis selben Problem vor. Jedoch haben beide ihre bevorzugten Anwendungen.

Geht man bis auf weiteres davon aus, dass Simulationsmodelle auch noch in anderen Simulationssystemen entwickelt werden - und Vielfalt hat sich bisher immer noch am Markt als Vorteil für den Endanwender erwiesen -, dann bietet SMI hier eine gute Lösung.

Die Entwicklung eines „Esperanto“ für Simulationsmodelle in Form von VHDL-A scheint sicher für die Zukunft und Neuentwicklungen interessant zu sein und wird in Teilbereichen der Industrie auch zu den erwünschten Resultaten führen.

8. Referenz

- [NIEM91] M. Niemeyer: *Das Simulatorkopplungs-System SiCS*, 7. Symposium Simulationstechnik Hagen, September 1991; Fortschritte in der Simulationstechnik Band 4, Vieweg
- [SCHW91] P. Schwarz, P. Trappe: *Einbeziehung des Multi-Level-Simulators KOSIM in ein Simulatorkopplungsprojekt*, 7. Symposium Simulationstechnik Hagen, September 1991; Fortschritte in der Simulationstechnik Band 4, Vieweg 1991
- [RUZI93] R. Ruzicka: *SIMUL_R PARALLEL - hardware-in-the-loop Simulation mit Transputern unter Windows*, 8. Symposium Simulationstechnik Berlin 1993; Fortschritte in der Simulationstechnik Band 6, Vieweg 1993
- [RUZI96] R. Ruzicka: *Simulatorkopplung durch Übersetzung, SMI - Simulation Module Interface*, 10. Symposium Simulationstechnik Dresden, September 1996; Fortschritte in der Simulationstechnik Band 10, Vieweg 1996
- [MGAI91] MAG Inc., *Advanced Continuous Simulation Language*, ACSL Reference Manual Version 10.0, Concord MA, 1991.
- [MATH92] The MathWorks Inc., *SIMULINK User's Guide*, Natick, Massachusetts, 1992
- [OTTE95] M. Otter, *The DSblock model interfac for exchanging model components*, Proceedings Eurosime Conference 1995, Elsevier, Wien
- [IEEE93] IEEE VHDL subPAR 1076.1: *Analog extensions to VHDL. Design Objective Document (DOD)*. Version 1.1, März 1993
- [RUZI88] R. Ruzicka: *SIMUL_R - eine Simulationssprache mit speziellen Befehlen zur Modelldarstellung und -analyse*, Informatik Fachberichte 179, 5. Symposium Simulationstechnik Aachen, Springer, 1988