

Bild: © V. Yakobchuk – Fotolia.com

# Wie hoch ist die Performance?

Eine integrierte Performance-Analyse verbessert den Entwicklungsprozess von Steuergeräten im Rahmen einer **MODELLBASIERTEN ENTWICKLUNG** mit UML/SysML. Dafür sind einige notwendigen Anpassungen wie die Erstellung eines UML-Profiles und die Anpassung der Code-Generierung erforderlich.

**A**ls international tätiger Automobilzulieferer sieht sich die Firma Hella KGaA Hueck & Co. einer zunehmenden Komplexität bei der Entwicklung von Produkten im Bereich Licht und Elektronik gegenüber. Zur Bewältigung dieser Komplexität, aber auch zur Beschleunigung und Absicherung des Entwicklungsprozesses, greift das Unternehmen zunehmend auf Methoden der modellbasierten Entwicklung zurück.

Ein modellbasierter Entwurf ist insbesondere dann effektiv, wenn der Entwickler bei seinen Entwurfsentscheidungen durch (automatische) Analysen unterstützt wird. Dies sollte auch schon in möglichst frühen Phasen der Entwicklung geschehen, so dass die getroffenen Entscheidungen frühzeitig, also bereits während der Systemanalyse und dem Systemdesign, verifiziert werden können. Dadurch muss der Entwickler sich nicht mehr allein auf seine Erfahrung verlassen.

Wenn wir hier von Analysen spre-

chen, dann bedeutet das zunächst die Sicherstellung der Konsistenz des Gesamtmodells (beispielsweise die korrekte Verwendung von Schnittstellen). Für Anwendungen im Bereich Embedded-Systeme, speziell im Automobilbereich, ist aber auch die Analyse bezüglich der Ressourcen-Auslastung und der Einhaltung

**Wenn zur Analyse ein Echtzeitsimulator zum Einsatz kommt, dann kann er direkt auf das Modell zugreifen, in dem die Entwurfsentscheidungen dokumentiert sind.**

von Echtzeitanforderungen von Bedeutung. Nur so können frühzeitig Engpässe gefunden werden und sinnvolle Zeitbudgets für maximale Ausführungszeiten festgelegt werden.

Im weiteren Verlauf des Entwicklungsprozesses werden die vorhandenen Modelle verfeinert. Dadurch werden

wiederum Analysen auf einem höheren Detaillierungsgrad möglich und notwendig. So sind im Software-Design eher Fragen zur Laufzeit innerhalb der gewählten Software-Architektur (Jitter, Zyklus- und Laufzeiten sowie Prioritäten von Tasks, Laufzeiten einzelner Funktionen, Ressourcenzugriffe) oder die Analyse von Nebenläufigkeiten (Race Conditions) von Interesse.

Für einen effektiven Einsatz im Entwicklungsprozess müssen die oben beschrie-

benen Entwicklungsphasen zunächst auf eine modellbasierte Vorgehensweise abgebildet werden. Da die Modellierung und Analyse des zugrunde liegenden Modells aber häufig von unterschiedlichen spezialisierten Werkzeugen durchgeführt wird, ist die nahtlose Integration der verwendeten Werkzeuge nicht nur

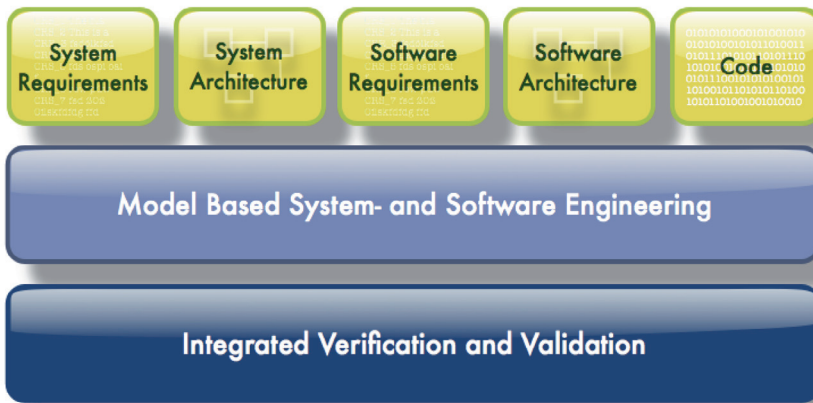


Bild 1: Modellbasierte Entwicklung ermöglicht ein gesamtheitliches Vorgehen.

aus Sicht des Entwicklungsprozesses sinnvoll, sondern bietet auch ein hohes Optimierungspotenzial. Durch die Integration verringert sich der Aufwand zur Durchführung einer Analyse deutlich. Kommt beispielsweise zur Analyse der oben beschriebenen Fragen im Bereich Laufzeiten und Ressourcen-Auslastungen ein Echtzeitsimulator zum Einsatz, so kann dieser direkt auf das Modell zugreifen, in dem die Entwurfsentscheidungen dokumentiert sind. Ein manuelles Übertragen der Informationen in das Werkzeug entfällt. Bild 1 verdeutlicht dieses Prinzip.

Dieser Beitrag zeigt in exemplarischer Art und Weise auf, wie die Integration zwischen dem Echtzeitsimulator chronSIM

und dem Modellierungswerkzeug Rhapsody erfolgt.

**Modellierung**

Als SysML/UML-basiertes Werkzeug bietet Rhapsody die Möglichkeit, die Struktur und das Verhalten des Systems mit den bereitgestellten Diagrammen und Notationselementen zu modellieren. So können Verteilungs- und Blockdiagramme zur Darstellung der Systemarchitektur verwendet werden. Zur Modellierung der Software-Architektur werden Objekt- und Kompositions-Strukturdiagramme eingesetzt, während für das Verhalten Zustands-, Sequenz- und Aktivitätsdiagramme zum Einsatz kommen. In späteren Phasen wird aus dem Modell

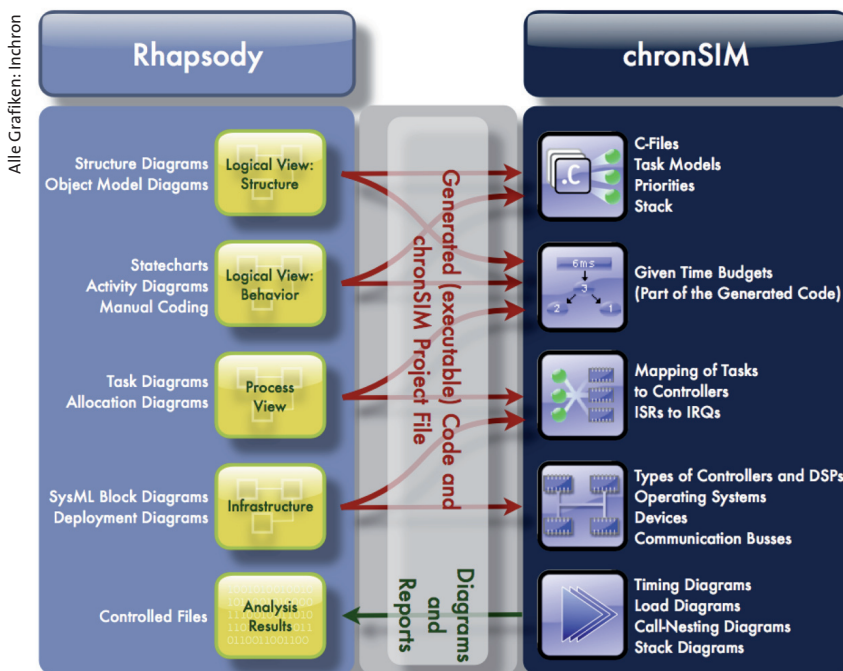


Bild 3: Abbildung von Rhapsody Modellelementen auf chronSIM.

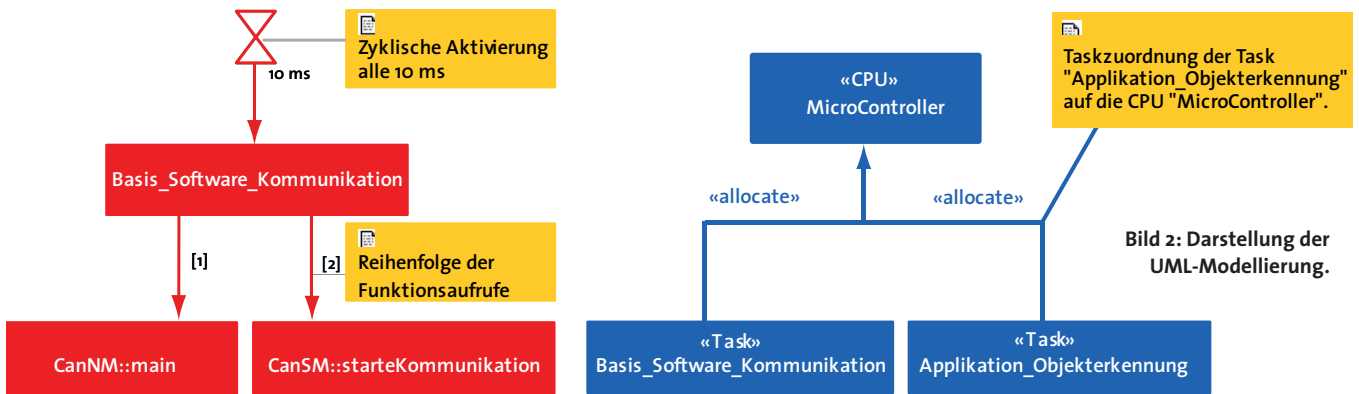


Bild 2: Darstellung der UML-Modellierung.

teilweise Code generiert, wobei einzelne Algorithmen manuell implementiert und hinzugefügt werden.

Für die Integration zwischen Rhapsody und chronSIM war es notwendig, zusätzliche Modellierungsmöglichkeiten bereitzustellen. Hierzu wurde der UML-Erweiterungsmechanismus des Profils genutzt. In diesem sind entsprechende Stereotypen mit Eigenschaftswerten (tagged values) gespeichert. Mit den vorhandenen und den neu hinzugefügten Diagrammen und Modellelementen ist es nun möglich, alle für eine Simulation notwendigen Informationen wie Tasks, Ressourcen oder Interrupts zu spezifizieren.

Die Modellierung dieser Betriebssystem-Elemente erfolgt mit Hilfe eines angepassten UML-Aktivitätsdiagramms, in dem asynchrone und nebenläufige Kommunikationsmechanismen spezifiziert werden können. Eine Task wird zum Beispiel als eine stereotypisierte Aktion dargestellt, die zusätzliche Informationen zur Aktivierung, Laufzeit, Priorität und der Unterbrechbarkeit beinhaltet (Bild 2). Die anderen Elemente (Ressourcen, Interrupts etc.) werden dementsprechend modelliert. Sie bilden zusammen die Prozesssicht auf das System (Bild 3).

Die Informationen aus der logischen Sicht dienen dazu, automatisch Task-Mo-

delle beziehungsweise C-Code – falls das Modell detailliert genug ist – für die Simulation zu generieren. Hierfür wurde der C-Codegenerator des Werkzeugs Rhapsody entsprechend angepasst, der nun zwischen der Generierung von abstrakten Task-Modellen und der C-Codegenerierung umschaltbar ist. Dies erlaubt bereits frühzeitig mit einer Abschätzung (Task-Modell), aber auch später mit dem realen Code, eine Simulation durch-

### In Zukunft soll ein eingeschränkter Abgleich der Spezifikation in beide Richtungen möglich sein – auch von chronSIM zu Rhapsody.

zuführen. Die generierten Dateien beinhalten alle für eine Simulation notwendigen Informationen.

Die Ergebnisse der Simulation werden in verschiedenen Diagrammen, wie beispielsweise zeitlich annotierten Sequenz-, Auslastungs- oder Zustandsdiagrammen umfassend dargestellt. Diese werden anschließend ins UML-beziehungswise SysML-Modell als Grafik eingefügt (Controlled File), damit der Architekt sie dann zusammen mit seinem Modell in einem Werkzeug betrachten und auswerten kann. Er erhält dadurch beispielsweise die CPU-Auslastung oder die maximale Antwortzeit des Systems (Bild 4).

Der hier beschriebene Ansatz ist ein erster Schritt hin zu einer integrierten Performance-Analyse. Bisher ist es möglich, die in Rhapsody modellierten Informationen zum Simulationswerkzeug chronSIM zu übertragen und die Analyseergebnisse in Form von Grafiken im (Rhapsody-)Modell abzulegen und zu versionieren. Dies ist insbesondere zur Generierung von Reports nützlich und sinnvoll.

### Zukunft

In Zukunft soll es auch möglich sein, einen eingeschränkten Abgleich der Spezifikation in beide Richtungen, also auch von chronSIM zu Rhapsody zu ermöglichen. Ziel ist die Übernahme von Änderungen, welche während der Echtzeitanalyse an dem Modell innerhalb von chronSIM vorgenommen werden, automatisch in das Rhapsody-Modell zu übertragen. Dadurch wäre die Analyse und

Optimierung unterschiedlicher Konfigurationen des Systems (z.B. unterschiedliche Prioritäten und Zykluszeiten von Tasks) direkt in chronSIM möglich. Die dort vorgenommenen Entwurfsentscheidungen

werden abschließend mit dem Rhapsody-Modell synchronisiert. Das Forschungsprojekt SPES 2020 arbeitet derzeit an der vollständigen Integration einer Performance-Analyse in den Entwicklungsprozess. ←

*Ulrich Nickel war bei der Hella KGaA Hueck & Co. verantwortlich für Prozesse, Methoden und Tools im Bereich Anforderungsmanagement und Systemarchitektur.*

*Jan Meyer ist wissenschaftlicher Mitarbeiter am s-lab (Software Quality Lab) der Universität Paderborn und arbeitet zusammen mit der Hella KGaA Hueck & Co. im Bereich der modellbasierten Entwicklung.*

*Tapio Kramer, Marketing Manager bei Inchron, ist zwar nicht Autor, wohl aber Vermittler dieses Beitrags. Inchron hat nämlich für Hella ein chronSIM-Profil erstellt.*

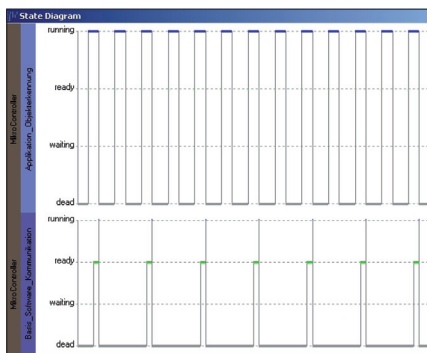


Bild 4: Zustands- und Auslastungsdiagramm aus chronSIM.

infoDIRECT [www.all-electronics.de](http://www.all-electronics.de)

Link zu Hella und Inchron 346AEL0310