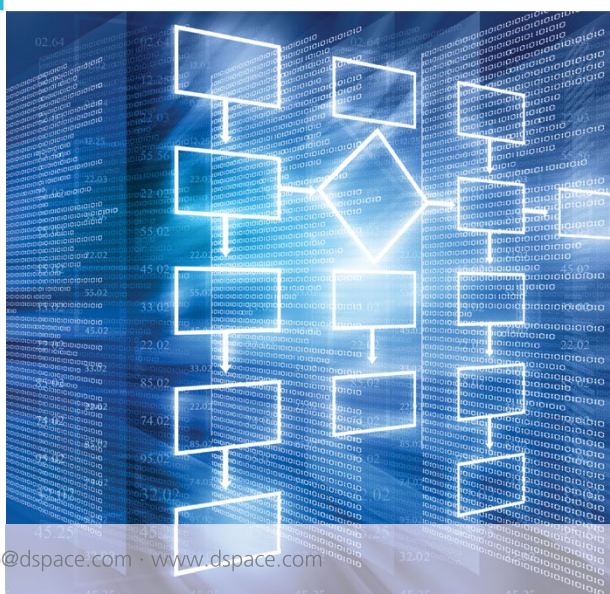
Die Lösung des Komplexitäts-Puzzles

Die Entwicklung umfangreicher Funktionalitäten in großen Arbeitsgruppen mit einer hohen Zahl von Software-Entwicklern unter Einbeziehung unterschiedlicher Entwicklungspartner bringt eine Reihe neuer Herausforderungen an den Entwicklungsprozess mit sich. Dabei spielt der Aufbau einer zuverlässigen Werkzeugkette für eine effiziente, modellbasierte Software-Entwicklung eine wesentliche Rolle. Simulink/TargetLink sowie die Werkzeuge der Firma Model Engineering Solutions bieten hierfür genau die richtigen Lösungen.

Komplexität und Konsistenz umfangreicher Modelle bei der verteilten Entwicklung beherrschen



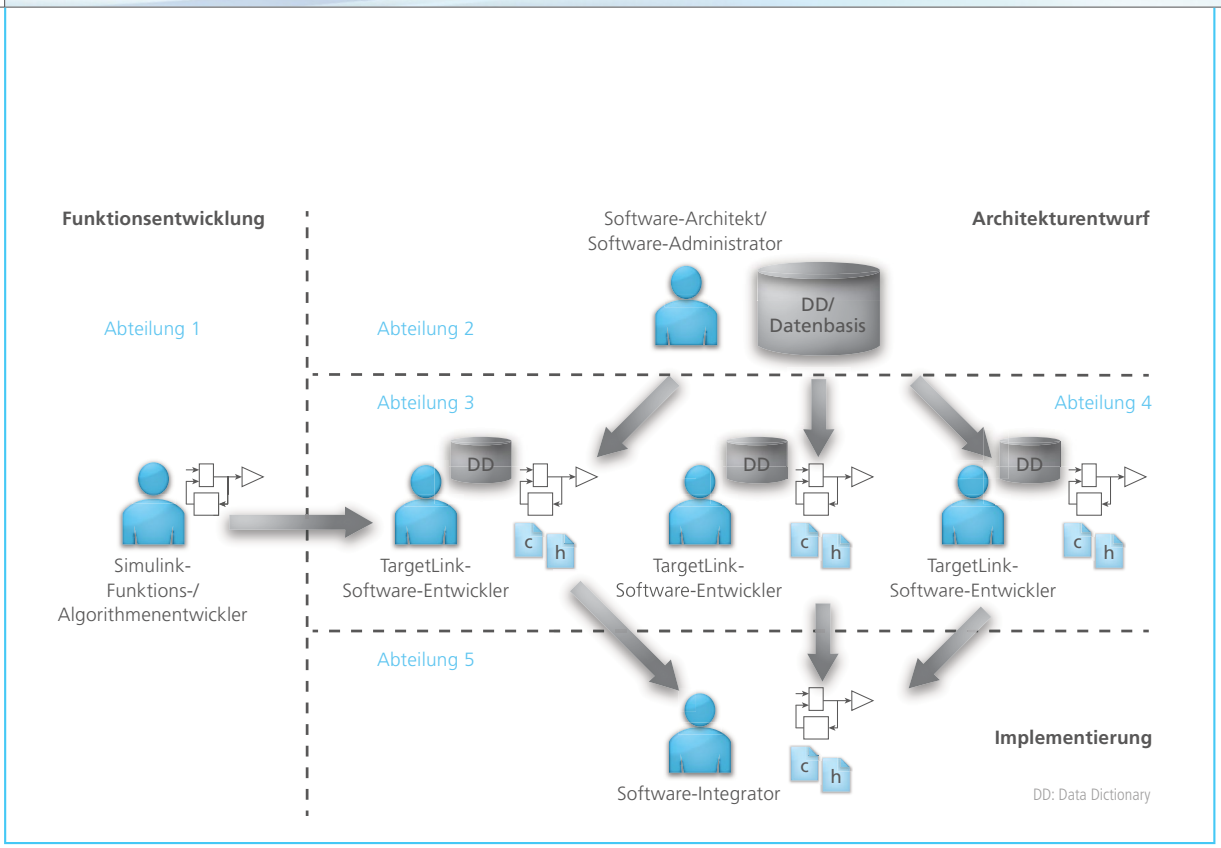


Abbildung 1: Verteilte Entwicklung in großen Teams. Das Modell wird in den Domänen Architekturentwurf und Funktionsentwicklung definiert und in der Teilfunktionsentwicklung weiter angereichert. Die Arbeitsergebnisse werden zusammengefasst, geprüft und letztendlich auf einem Steuergerät implementiert.

Herausforderung: Verteilte Entwicklung

Ausgehend von den Simulink®/TargetLink®-Modellen als ausführbare Spezifikationen der Software-Funktionen, werden neben dem Code weitere Artefakte wie A2L-Dateien, AUTOSAR-XML-Dateien oder Software-Dokumentation generiert. Beschränkt sich der Entwurf und die automatische Code-Generierung auf einzelne Software-Komponenten und Funktionen, fällt meist erst bei der Integration auf, dass es Inkonsistenzen zwischen einzelnen Komponenten gibt. Es fehlen oft Mechanismen für die Prüfung vorangehender Entwicklungsschritte. Durch die zunehmende Komplexität der Funktionen im Fahrzeug und der dadurch verteilten Entwicklungsumgebung aus vielen Arbeitsgruppen und einer großen Anzahl an Software-Entwicklern verschärft sich diese Problematik. Sowohl die Entwicklungsmechanismen als auch eine auf Simulink/TargetLink abgestimmte Werkzeugkette müssen angepasst werden, um modulare, verteilte Entwicklung großer Funktionalitäten effizient zu realisieren.

Methode	Vorteil
Modellierungsrichtlinien	■ Konsistenz ■ geringere Fehleranfälligkeit ■ weniger Überarbeitungsschleifen
Wiederverwendung (Bibliotheken, referenzierte Modelle)	■ Modulare Entwicklung ■ Übersichtlichkeit durch Modellorganisation und Hierarchie ■ Verringerter Entwicklungsaufwand durch Wiederverwendung gleicher Modellteile
Single-Source-Spezifikationen	■ Vereinfachter Austausch zwischen Entwicklungsteam-Mitgliedern durch Software- und Schnittstellenspezifikation im TargetLink Data Dictionary
Inkrementelle Code-Generierung	■ Zeitersparnis bei Reviews ■ Geringere Code-Generierungszeiten ■ Vereinfachte Software-Integration und leichtere Software-Tests
Code-Generierung aus dem DD	■ Generierung geteilt genutzter Variablen in eine Datei
Diff&Merge-Mechanismen durch das TargetLink-DD und Model Compare	■ Nachvollziehbarkeit und Verfolgung von Änderungen an Schnittstellendefinitionen und im Modell
Komplexitätsanalyse mit M-XRAY	■ Hinweis auf geeignete Modellpartitionierung

Konsistenz und verringerte Fehleranfälligkeit durch Modellierungsrichtlinien

Simulink/Stateflow bietet eine Vielzahl von Modellierungsmöglichkeiten, von denen nicht alle für eine effiziente Seriencode-Generierung geeignet sind. Insbesondere wenn viele Entwickler an der Software-Entwicklung mitwirken, empfiehlt sich der Einsatz von Modellierungsrichtlinien, die das Risiko fehlerhafter Modelle deutlich verringern. Die Anwendung der Richtlinien minimiert Überarbeitungsschleifen, vereinheitlicht Modellierungsstile, vereinfacht das Testen und schafft eine gemeinsame Referenz für die Durchführung von Reviews.

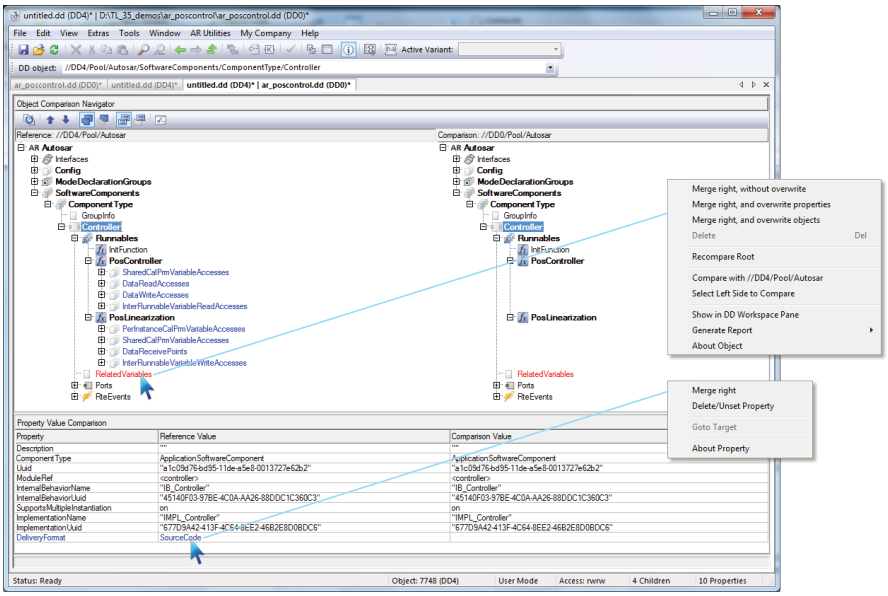


Abbildung 2: Vergleich unterschiedlicher Software-Spezifikationen

Auch der Modell- bzw. Funktionsaustausch zwischen unterschiedlichen Entwicklungsteams wird vereinfacht. Unterstützende Werkzeuge zur automatischen Richtlinienprüfung wie der MES Model Examiner® dienen der Prüfung auf die Einhaltung und der Korrektur von Verstößen gegen die Richtlinien.

Partitionierung und Wiederverwendung von Modellen

Wesentlicher Bestandteil im verteilten Entwicklungsprozess ist das Single-Source-Prinzip. Dieses Prinzip beinhaltet weitestgehend die Verwendung des gleichen Modells in verschiedenen Entwicklungsphasen vom Design der Regelung bis hin zur Integration. Diese Mechanismen werden in Simulink/TargetLink realisiert durch:

- Simulink-Bibliothek-Mechanismen zur Wiederverwendung von Modellteilen, die jeweils mehrfach instanziiert werden, oder
- Model-Referencing-Mechanismen zur Einbindung von Modellen in andere Modelle.

Vereinfachter Austausch und Administration

In großen Entwicklungsteams werden Aufgaben wie die Funktionsentwicklung, Software-Architektur und Administration, Software-Entwicklung und Integration selten von gleichen Personen durchgeführt. Vielmehr greift eine Vielzahl der Teammitglieder auf gleiche Informationen zu (Abbildung 1). Insbesondere die Spezifikationen werden zwischen einzelnen Funktionsentwicklern ausgetauscht, bearbeitet und gespeichert und müssen somit konsistent gehalten werden. In TargetLink gibt es hierfür speziell das TargetLink Data Dictionary (TL-DD), das unterschiedliche Austauschformate wie beispielsweise XML, AUTOSAR XML standardmäßig unterstützt. Die Datenobjekte im Modell und im DD sind miteinander verlinkt, so dass einerseits der Algorithmus von den Daten sauber getrennt wird und zum anderen die Synchronität zwischen den Daten im Modell und DD sichergestellt wird.

Leistungsfähige inkrementelle Code-Generierung

Die inkrementelle Code-Generierung ist eine weitere zentrale Methode der verteilten, modellbasierten Entwicklung. Dabei wird für abgegrenzte Software-Einheiten getrennt von-

einander inkrementell Code generiert. Die Auswirkungen von Änderungen an einer kleinen Funktion auf die gesamte Software-Funktion werden eingegrenzt, da die Software-Einheiten inhaltlich unabhängig voneinander sind. Der Code muss nur für die inhaltlich geänderten Einheiten erneut generiert werden, während der Rest unverändert bleibt. Manuelle Reviews sind weniger zeitintensiv und Code-Generierungszeiten werden durch die Zerlegung in kleinere Einheiten deutlich reduziert. Damit ist eine effizientere und schnellere Entwicklung großer Funktionalitäten möglich.

Code-Generierung aus dem DD

Die Code-Generierung aus dem DD erfolgt ebenfalls unabhängig vom konkreten Modell, wobei übergreifende oder geteilt genutzte Variablen direkt aus dem DD in eine Datei generiert werden. Anwendung findet dieses Verfahren in folgenden Szenarien:

- Erzeugung einer Datei, die alle modellübergreifenden Variablen enthält, z.B. Interface-Variablen, und Festlegung der Rechte auf diese globalen Ressourcen
- Generierung aller Kalibrierparameter unterschiedlicher Funktionen in eine Kalibrierparameterdatei
- Erzeugung von Variablen in einer Datei, die in automatisch generiertem oder handgeschriebenem Legacy-Code verwendet werden

Effiziente Diff&Merge-Mechanismen

Die Nachvollziehbarkeit von Änderungen ist unverzichtbar im Software-Entwicklungsprozess, insbesondere wenn an der Entwicklung unterschiedliche Abteilungen und Zulieferer beteiligt sind. Veränderte Software-Artefakte werden zwischen Software-Architekten, Funktionsentwickler und Integrator ausgetauscht. Eine zuverlässige Werkzeugkette zur Ermittlung der Unterschiede ist äußerst wichtig. Im TargetLink Data Dictionary sind Mechanismen vorhanden, um die Versionsstände zu vergleichen und Unterschiede anzuzeigen (Abbildung 2). Bei der Kontrolle können diese Unterschiede zurück ins Modell verfolgt werden, um deren Auswirkungen auf die Modellierung festzustellen. Für den Vergleich von Modellen an sich bietet z.B. Model Compare von dSPACE komfortable und umfangreiche Funktionalitäten. Mechanismen im DD aktualisieren Schnittstellendefinitionen automatisch, um bei Änderungen die Konsistenz herzustellen (Abbildung 2).

Abbildung 3: Analysebericht von M-XRAY

Struktur: Hauptteil

Name	Komplexität	*	Name	Komplexität	*	Name	Komplexität	*	Name	Komplexität
Fuelsys	96		EGO sensor	33						
			MAP sensor	33						
			Engine speed	33						
			Engine gas dynamics	33		Mixing & Combustion	59		System lag	9
						Throttle & Manifold	42		Intake Manifold	67
									Throttle	350
			Fuel rate controller	58		Airflow calculation	780			
						Fuel calculation	75		Switchable Compensation	200

Validierung der Modellarchitektur

Für die verteilte Entwicklung mit TargetLink ist eine Aufteilung der Modelle in Teilfunktionen wie Subsysteme unerlässlich. Die Komplexität der einzelnen Subsysteme sollte nicht zu hoch sein, um beispielsweise die Anzahl möglicher Fehler zu reduzieren und um diese lesbarer und wartbarer zu halten. Zudem werden dadurch wichtige Anforderungen von Sicherheitsstandards wie ISO 26262 erfüllt, die eine niedrige Komplexität fordern. Eine automatisierte Überprüfung der Modellkomplexität ist mit dem Add-on M-XRAY® des MES Model Examiner möglich. M-XRAY berechnet und bewertet die Komplexität des Gesamtmodells sowie der einzelnen Subsysteme. Zusätzlich werden alle relevanten Modellmetriken erhoben, die für die qualitative Bewertung von TargetLink-Modellen relevant sind (siehe Exkurs zu Modellmetriken). Der durch M-XRAY erstellte Analysebericht (Abbildung 3) beinhaltet einen strukturellen Überblick, der die hierarchische Struktur des Modells und für jedes Subsystem den Wert der jeweiligen Komplexität zeigt. Durch diese Darstellung kann die Verteilung der Komplexität in einem Modell sehr einfach bewertet werden und besonders komplexe Subsysteme lassen sich leicht identifizieren. ■

Referenzwerte für die lokale Komplexität

Referenzwert	Bewertung
MV ≥ 750	Hoch
MV < 750	Mittel
MV < 300	Niedrig

Zusammenfassung

Zur Beherrschung von Komplexität und Konsistenz in der verteilten Entwicklung steht eine Vielzahl effizienter Methoden zur Verfügung, die sich auch für große Teams eignen.
Zum Erfolgsrezept gehören eine zweckmäßige Modellpartitionierung, die Nutzung von inkrementeller Code-Generierung und der werkzeugunterstützten Änderungsverfolgung. Die Partitionierung kann durch Messungen und Metriken bewertet werden. Die beschriebenen Vorgehensweisen erleichtern die Entwicklung großer Funktionalitäten und ermöglichen einen effizienten und fehlerresistenteren Austausch von entwickelten Teilfunktionen.

Exkurs

Modellmetriken

Metriken ermöglichen es, Modelle zu vergleichen und im Hinblick auf Komplexität und Qualität zu bewerten. Sicherheitsnormen wie die ISO 26262 schreiben die Prüfung der Komplexität für sicherheitsrelevante Modelle verbindlich vor (vgl. ISO 26262, §5.4.7, Tabelle 1).
Darüber hinaus kann die Erhebung von Modellmetriken für die Abschätzung der Aufwände für Tests und Reviews genutzt werden. Durch die kontinuierliche Erfassung der Metrikwerte für unterschiedliche Entwicklungsstände lässt sich die Entwicklung eines Modells überwachen, und besonders komplexe und fehleranfällige Modellteile lassen sich frühzeitig identifizieren.

Für die Messung der Modellkomplexität werden Zählmetriken herangezogen wie Blockanzahl, Modellierungstiefe, Schnittstellenbreite oder auch die Zyklomatische Komplexität. Diese Metriken basieren auf Konzepten aus der Programmierung, sind aber für die Bewertung von Modellen selten geeignet. Aufgrund der Datenflussorientierung von Simulink ist beispielsweise eine Bewertung der Zyklomatischen Komplexität von Modellen nur wenig aussagekräftig.

Als wichtigstes Maß zur Bewertung der Modellkomplexität etabliert sich in der Industrie derzeitig das von der Halstead-Metrik abgeleitete Maß des *Modellvolumens* (MV). Es ermöglicht die Bewertung der Modellkomplexität, da es nicht nur die im Modell enthaltenen Blöcke berücksichtigt, sondern auch die Verknüpfung der Blöcke, deren Gewichtung und eigene Komplexität sowie die Verknüpfung von Blöcken durch Signale untereinander.

Analyse und Bewertung von TargetLink-Modellen durch Modellmetriken können durch den MES Model Examiner® und das Add-on M-XRAY sehr effizient durchgeführt werden. M-XRAY analysiert die Modelle, berechnet das Modellvolumen und alle relevanten Metrikwerte. Die Ergebnisse werden in einer kompakten Übersicht tabellarisch und strukturell anschaulich dargestellt. Somit besteht die Möglichkeit, die Verteilung der Komplexität in einem Modell effizient zu ermitteln und letztendlich niedrig zu halten. ■

Literatur:
Stürmer, I., Pohlheim, H., Rogier, T.: „Calculation and Visualization of Model Complexity in Model-based Design of Safety-related Software", (in German) in Keller, B. et. al., Automotive - Safety & Security, Shaker, pp. 69 82, 2010.