Model Driven Software Engineering 2.0
Manifest zur erfolgreichen Anwendung der MDSE

Andreas Foltinek, IMACS GmbH

1. Bestandsaufnahme zur MDSE
Immer komplexere Strukturen, Funktionen und das Zusammenspiel technischer Systeme bestimmen unseren Alltag. Die einhergehende fortwährende Aufgabenverlagerung von Mechanik- und Elektronikkomponenten auf die Software rücken diese immer stärker in den Fokus der Innovationen und erhöhen damit deren Bedeutung und Aufwendungen. Leider wird dieser Entwicklung nicht immer Rechnung getragen und der Software lastet - auch auf Leitungs- und Führungsebene - immer noch oft der Aspekt eines notwendigen Beiwerks an. Hinzu kommt, dass ein allgemeines Bewusstsein eines immer möglichen und vermeintlich kostenlosen Software-Updates in den Köpfen besteht und dieses Verständnis durch Nutzer und Konsumenten, die dies mittlerweile als selbstverständlich akzeptieren, noch gefördert wird.

All diese Umstände führen allzu oft zu einer gewissen Lockerheit im Umgang mit Software und den damit verbundenen Arbeitsweisen bei deren Entwicklung. Das, was im Bauwesen, dem Maschinenbau und der Elektrotechnik seit über 25 Jahren zur Normalität gehört - dem Arbeiten mit Modellen und deren direkte Verwendung für die "Realisierung" von Produkten - wird in der Softwareentwicklung selbst, oft noch argwöhnisch betrachtet und zögerlich angewendet.
Im Hinblick auf die aktuell diskutierten Trends wie Industrie 4.0, IoT oder das autonomes Fahren, in denen Computersysteme ohne menschliches Zutun Daten austauschen, miteinander agieren, sich synchronisieren, lenken … werden sich obige Anforderungen noch einmal vervielfachen. Auf Basis der klassischen Programmierung wird man diesem Trends nicht gewachsen sein.

Es soll jedoch auch nicht verschwiegen werden, dass Ende des letzten Jahrtausends die MDSE bereits einen Hype erlebte und die Riege der Entwickler durchaus nach neuen Wegen suchte und diese auch vereinzelt beschritt. Die damals allgemein favorisierten Umsetzungswege hatten sich in der Praxis jedoch als oft wenig effektiv und teilweise sogar kontraproduktiv erwiesen. Wie bei jeder neuen Methodik musste reflektiert und sich anhand der Erfahrungen neu besonnen werden. Diese "verbrannte Erde" gilt es angesichts der aktuellen großen SW-Herausforderungen wieder zu kultivieren und die Modellbasierung als Methode selbst zu einem selbstverständlichen Arbeitsstandard zu machen.

2. Intension der gemeinsamen Initiative
Auch nach über 15 Jahre Model Driven Development im Software Engineering bleibt diese also hinter ihrem Einsatzpotential noch deutlich zurück. Was ist der Grund?
Um diese zu beantworten haben wir uns zusammengefunden und Resümee gezogen. Wir, das sind verschiedene Akteure, die in den letzten Jahren extensiv mit dem Thema MDSE in Berührung standen. Bezeichnender ist in dem Zusammenhang, dass, nach der Zeit der Findung, mittlerweile alle diese bekannten Akteure der Methoden- und Werkzeug-Szene über den Weg und die Maximen gleich denken. Wie so oft in der industriellen Evolution ist dies ein Zeichen, dass eine gewisse Reife erreicht wurde.

Man muss dabei erwähnen, dass diese Protagonisten allesamt diese Methoden nicht nur abstrakt vertreten, sondern in eigenen Projekten und Produkten auch langjährig konkret

erfolgreich einsetzen. Dabei hat sich über die Jahre, aus anfänglich sehr akademischen Ansätzen, eine naturgemäß praxisorientiertere Vorgehensweise gebildet und etabliert. Leider konnten sich diese Ansätze, auch aufgrund der eingangs erwähnten, negativen Erfahrungen der vergangenen Jahre noch nicht großflächig etablieren. Warum?

Diese Frage haben wir uns gestellt und konnten feststellen: Es funktioniert durchaus, wenn bestimmte Aspekte berücksichtigt und richtig eingeführt werden. Leider geschieht dies häufig nicht und die Erwartungen stellen sich nicht ein.
Aus diesem Grund haben wir ein Manifest fixiert und 7 Thesen aufgestellt die definieren, welche Aspekte MDSE wirkungsvoll ausmachen. Es soll der Community dazu dienen sich über das Thema auszutauschen und es weiterzuentwickeln.

3. Sieben Thesen und deren praktische Umsetzung

Im Folgenden werden die einzelnen Thesen vorgestellt. Kurze praxisbezogene Erläuterungen verdeutlichen die Hintergründe und erläutern Best Practice Empfehlungen als auch Negativbeispiele und Irrwege. Sie sollen als Anregung aber auch zur Reflexion und Bewertung der eigener, bisher praktizierten Ansätze und Vorgehensweisen dienen.

1) INTEGRIERE STAKEHOLDER – durch, für die Domäne passende Abstraktionen, Notationen und Sichten.

- Modelle sollen für Mitwirkende und Verantwortliche passend verständlich und übersichtlich sowie erkenntnisgewinnend sein.
- Die Verwendung dieser soll das Projektverständnis erhöhen und vereinfachen und Missverständnisse und Verkomplizierungen vermeiden.

2) STREBE LANGLEBIGE MODELLEN AN - durch die richtigen Abstraktionen, Separierung von Belangen und angemessene Style Guides.

- Modelle sollen von der Idee und den Anforderungen bis zur Generierung der Outcomes (z.B. Quell-Code, Hilfsmitteln, Dokumentationen) sich entwickeln und einzig maßgeblich sein.
- Modelle bzw. Modellteile sollen derartig gestaltet sein (abstrakt, eindeutig), dass eine Wiederverwendung anderweitig ohne Veränderung möglich ist.

3) ERSTELLE VALIDIERBARE, TRANSFORMIERBARE UND AUSFÜHRBARE MODELLE - durch semantisch wohldefinierte Sprachen für funktionale und nichtfunktionales Aspekte

- Modelle sollen in einer derartigen Sprache beschreiben sein, dass diese maschinell eindeutig in andere Formen (z.B. Sichten, Auszüge, ...) überführt werden können, ohne dabei Informationen zu verändern.
- Der Umfang sollte dabei möglichst umfassend und detailliert sein um damit Prüfungen oder direkt ausführbaren Code oder Simulationen zu erzeugen.

4) FRONT-LOADING - verwende Modelle für frühzeitigen Erkenntnisgewinn

- Modelle sollen von möglichst von Beginn ab, z.B. in der Findung von Ideen und Anforderungen zum Einsatz kommen.
- Vorteilhafterweise bieten diese verschiedene Sichtweise bis hin zu realistischen Simulationen

5) VERMEIDE DUPLIKATION UND UNNÖTIGE WIEDERHOLUNGEN - durch Automatisierung und Integration von Modellen verschiedener Aspekte

- Modelle sollten möglichst jede Art von Informationen an jedoch nur einer Stelle beinhalten. Eine Vervielfältigung/Mutation sollte rein durch Referenzierung/Vererbung erfolgen.
- Modelle und deren Anwendungsart (Profile) sollen konsequent eine Redundanz von Informationen vermeiden durch z.B. Visualisierung und damit Wiederverwendung.

6) MACHE MODELIERUNG LEICHT ZUGÄNGLICH - mittels skalierbarer, benutzerfreundlicher, und leicht erlernbarer Werkzeuge und Infrastruktur

- Modellbasierte Werkzeuge (Editoren, Versionierung, Generatoren) sollten verständlich und leicht einsetzbar sein, sowie anderweitige, nicht modellbasierte Aspekte/Informationen abbilden können.
- Komplizierte, unpassende (bzgl. Domäne), übertrieben umfangreiche Werkzeuge bzw. Werkzeugketten sollten vermieden werden.

7) ETABLIERE EINE MODELLIERUNGSKULTUR - durch Ausbildung, Training und Integration mit dem Entwicklungsprozess

- Die modellbasierte Entwicklung soll als die Standardmethode verstanden, und der Umgang damit auch von Führungsebenen entsprechend favorisiert und ausreichend geschult und unterstützt werden.

4. Betrachtung der Beziehung Entwickler-Denkweise-Tool

Auch bei kritischer Betrachtung erscheinen die genannten Thesen durchaus griffig und richtig und können ein Garant für die erfolgreiche Anwendung der MDSE sein. Umso mehr stellt sich die Frage warum doch so oft noch klassisch gearbeitet wird. So gern embedded Softwareentwickler selbst komplexe Dinge und Systeme neu erschaffen, so herrscht doch, entgegen ihrer Kollegen aus der IT, bzgl. der eigenen Arbeitsweise und Werkzeugwahl ein gewisser konservativer Hang. Der über die Jahre liebgewonnene Editor mit projektglobaler, kontextbezogener Eingabe-Vervollständigung scheint "Tool genug" zu sein. Umgesattelt wird oft weniger aus eigenem Antrieb sondern nur, wenn es gar nicht mehr anders geht (20% Gain - 80 % Pain - Regel).

Leider fehlt es aber auch, und das im Gegensatz zu allen anderen Ingenieurswissenschaften, in der Branche der embedded Softwareentwicklung an weitläufig anerkannten Standards und damit der Gewissheit "etwas allgemein etabliert Richtiges" zu tun. Viele modellbasierte Werkzeuge im Maschinenbau und der E-Technik bilden in der Basis Standards ab und ermöglichen deren leichten Einsatz. Diese fehlen in der Embedded Branche bzw. hat jedes Entwicklerteam oft über Jahre gewachsene eigene Standards.
Da Werkzeugentscheidungen immer mehr nach der Prämisse "nur Nichts falsch machen" anstelle von "was nutzt uns am meisten" getroffen werden, haben es die Entscheider schwer. Welche modellbasierte Methode soll man denn verwenden, ohne dass man als Teamleiter bei einer Bauchlandung später zur Rechenschaft gezogen wird? "Bevor ich etwas falsch mache, bleibe ich lieber bei der aktuellen Arbeitsweise, das hat bisher auch halbwegs geklappt".

In der praktischen Anwendung werden mit modellgetriebenen Ansätzen häufig lediglich die statischen Aspekte eines Systems adressiert, z.B. durch die Darstellung von Klassen und deren Beziehungen. Sehr häufig wird auch lediglich die grafische Darstellung genutzt, die dann fast immer formal unkorrekt ist. Diese Ansätze helfen eventuell das schlechte Gewissen zu beruhigen, aber nicht Komplexität zu beherrschen, denn die wirk-

lichen Probleme die durch Komplexität entstehen (Emergenz, Hidden Links, Funktion) werden nicht adressiert.
Eine formal korrekte, dynamische Darstellung der Kommunikation zum Beispiel durch Sequenz Diagramme, eine zeitliche Darstellung der Zusammenhänge durch Timing Diagramme verbunden mit einer frühen Simulation des Systems sind da schon eher geeignet. Dieses und weiteres Potential des MDSE wird in der Praxis noch zu selten ausgenutzt. Das führt leider häufig zu dem Schluss, dass MDSE den erwarteten Nutzen nicht geliefert hat.

Ob UML, SysML, DSL oder eigene Ansätze: Uns Akteuren ist mittlerweile bewusst – und da sind wir uns einig - dass es "die Wahrheit" in dem MDSE nicht gibt. Jede Methode hat Ihre Schwerpunkte und Vorzüge und kann die jeweiligen Anforderungen und Aufgaben mitunter exzellent lösen. Die Modellbasierung hat so viele Gesichter wie Anwendungsgebiete. Einen universellen Werkzeugkasten gibt es nicht bzw. würde dieser stets nur Halblösungen bieten. Dafür ist die Thematik der embedded Softwareentwicklung einfach zu komplex und vielschichtig.

5. Fazit

Unsere Erfahrung zeigt: Es liegt nicht am MDSE an sich, sondern daran wie es umgesetzt und eingesetzt es wird. Dass das Model Driven Development geeignet ist, die eingangs beschriebenen Herausforderungen und Anforderungen unserer Zeit zu managen zeigt sich in den vielen Projekten, in denen MDSE konsequent Erfolge bringt.
"Die beste Methode" in der MDSE gibt es nicht. Stets zeigt der entsprechende Anwendungsfall und die Zielsetzung verschiedene Schwerpunkte auf, die es zu finden und anzupacken gilt. Wohl aber gibt es "die (sieben) besten Vorgehensweisen" bei der Umsetzung einer modellbasierten Methode bzw. mit dieser umzugehen.

Vom Kleinwagen über Traktoren und Lieferwagen zu Rennboliden gibt es eine Vielzahl an Automobilen, die alle Ihren besonderen Zweck hervorragend erfüllen. Keiner wird das Automobil verteufeln, nur weil er mit einem Sportwagen auf einem Feldweg steckenbleibt oder er durch falsche oder ungeübte Fahrweise einen Unfall verursacht. Vielmehr wird jeder einsehen das passende Fahrzeug zu verwenden UND ... zuerst richtig Fahren zu lernen.

Die Zeit ist reif für MDSE! Nutzen auch wir in der Embedded Software-Entwicklung flächendeckend die Methoden, die für Maschinen und Anlagen seit Langem zum Standard zählen.

6. Ausblick, Ziele der Initiativem und weitere Aktivitäten

Im Rahmen der MDSE-Initiative werden gemeinsam folgende Aktivitäten forciert:

- Bewusstseinsschärfung für die Wichtigkeit von (embedded) Software auch über betriebliche Hierarchien hinweg.
- Aufbau einer Kultur der modellbasierten (embedded) Softwareentwicklung mit entsprechenden Grundsätzen und Best Practice Ansätzen.
- Erstellen von Publikationen.
- Veranstalten von Kongressen (Mesconf) und Teilnahme bei einschlägigen Events
- Forcieren von globalen, lokalen bzw. branchenorientierten Treffen
- Animation zur Beteiligung an der MDSE in verschiedenen Ebenen

Dies alles wollen wir nicht allein tun und fordern daher alle Interessierten dazu auf teilzunehmen und uns ihre Belange (z.B. Erfahrungen, Wünsche, Unterstützungsgesuche) mitzuteilen und sich an Diskussionen zu beteiligen. Fordern Sie uns, uns zu fördern!

Autor

Andreas Foltinek studierte Elektrotechnik an der Universität Stuttgart.
Bereits vor seinem Studium sammelte er als "Jugend forscht" Teilnehmer und durch zahlreiche Projektentwicklungen für Firmen im Automatisierungs- und MSR-Bereich umfangreiche Erfahrung und kann auf mittlerweile über 30 Jahre Embedded Hard- und Softwarepraxis zurückblicken.
1994 gründete er die Firma IMACS GmbH und ist seither dort als Geschäftsführer für Forschung und Entwicklung verantwortlich. Als geistiger Vater eines UML basierten CASE-Tool-Systems gilt seine Leidenschaft den Methoden zur Softwareentwicklung sowie der Erhöhung der Einsetzbarkeit und des Abdeckungsgrades von Softwaremodellen und Code-Generatoren.
Schwerpunktmäßig beschäftigt er sich aktuell mit neuen Tool-Konzepten sowie der Beratung von Kunden beim Einsatz von OOP UML, MDD, deren praktischen Umsetzung und der hardwareseitigen Integration.

Kontakt

Internet: www.imacs-gmbh.de
Email: andreas.foltinek@imacs-gmbh.de