Sicherstellen semantischer Integrität im Big Data Umfeld – Problemstellung und Lösungsansatz

T. Hoppe[1,2], H. Eisenmann[2], A. Viehl[1], O. Bringmann[1,3], C. Hennig[1,2]

[1]FZI Forschungszentrum Informatik
Haid-und-Neu-Str. 10-14
76131 Karlsruhe
{hoppe,viehl,hennig}@fzi.de

[2]Airbus DS GmbH
88039 Friedrichshafen
harald.eisenmann@astrium.eads.net

[3]Universität Tübingen
Sand 13
72076 Tübingen
bringmann@informatik.uni-tuebingen.de

Abstract: In diesem Beitrag wird die Notwendigkeit des konsequenten Einsatzes von IT-gestützten Modellen im Systems Engineering, exemplarisch aus Sicht der Raumfahrtindustrie, motiviert und dabei auf aktuell existierende Anforderungen und Probleme eingegangen. Aus wissenschaftlich-technischer Sicht wird dabei auf Problemstellungen aus dem Bereich Big Data, insbesondere auf problematische Aspekte zur Etablierung einer konsistenten Semantik auf der Modellierungsebene und zur Laufzeit, fokussiert. Es wird ein Ausblick auf einen entwickelten Ansatz zur Addressierung der identifizierten Herausforderungen skizziert, welcher auf einem methodischen Vorgehensmodell und relevanten technischen Lösungsbausteinen basiert.

1 Motivation

Im Ingenieursbereich haben Modelle vielseitiger Art und Weise eine lange Tradition. Dazu zählen unter anderem Zeichnungen auf Papier, abstrahierte Miniaturnachbauten, bis hin zu voll funktionsfähigen Simulationsmodellen. In den letzten Jahrzehnten verzeichnen insbesondere IKT-gestützte Modelle einen rasanten Anstieg bei der Nutzung im Systems Engineering. Dies ist vor allem einer steigenden Unterstützung durch entsprechende Modellierungssoftware zu verdanken. Diese ermöglicht sowohl eine effektive und effiziente Erstellung von Modellen als auch eine einfache Anpassung bei notwendigen Änderungen. Außerdem wird eine Wiederverwendung und teilweise projektspezifische Anpassung ermöglicht. In diesem Kontext bilden Modelle vereinfachte Nachbildungen eines Systems, die der Lösung einer bestimmten Aufgabe dienen. Diese Aufgaben können bspw. kommu-

nikativer, analytischer oder demonstrativer Natur sein. Die eingesetzte Modellierungstechnologie hängt von den Anforderungen an das Modell selbst, der Erstellungskomplexität eines Modells, sowie weiteren Faktoren wie kommerziellen Aspekten ab.

Modellbasierte Lösungen, die den gesamten Entwicklungsprozess umfassen, sind heutzutage für jede Ingenieursdisziplin separat zu finden. Damit ist jeder Schritt von der Anforderungsspezifizierung, über Analyse, Design, Entwicklung, Produktion und Verifikation umfassend durch softwarebasierte Modelle unterstützt. Die Erfahrungen, die mit modellgetriebenem Engineering (engl. Model-Driven Engineering) im Softwarebereich gemacht wurden, sind in das klassische Systems Engineering transferiert worden. Daraus ist das modellbasierte Systems Engineering (engl. Model-Based Systems Engineering (MBSE) [FGS07]) entstanden. Für die Umsetzung von MBSE sind folgende Schritte notwendig:

1. Alle Modelle, die während des Systementwicklungsprozesses zum Einsatz kommen, müssen identifiziert und spezifiziert werden.
2. Für jedes Modell müssen die Schnittstellen mit dem Entwicklungsprozess und innerhalb der Domäne spezifiziert werden.
3. Um einen Informationsaustausch zwischen dem domänenspezifischen Modell und dem domänenübergreifendem Modell, dem sogenannten Systemmodell, zu ermöglichen, müssen die Schnittstellen miteinander verbunden werden.

Solch ein generelles Systemmodell bildet die Grundlage sowohl für den Informationsaustausch zwischen den einzelnen Domänen als auch für überblickhaftes Informationsmanagement auf Systemebene während des gesamten Systemlebenszyklus. Weiterhin muss das Systemmodell die Modellierung aller benötigten Abstraktionsniveaus und Entwurfsphasen unterstützen und von jeder am Entwicklungsprozess beteiligten Disziplin, den Kunden und den Zulieferern akzeptiert werden. Die einzelnen Disziplinen haben sich durchgesetzte, reife Werkzeuge, die alle domänenspezifischen Aspekte des Systems abbilden können und auch den Datenfluss innerhalb der Disziplin unterstützen. Eine Lösung, die den gesamten MBSE-Anwendungsbereich vom Erfassen der Anforderungen bis zur Auslieferung eines Systems umfasst, erweist sich als erheblich komplexer.

In diesem Beitrag werden die konkreten Herausforderungen von MBSE in der Raumfahrt vorgestellt und ein möglicher Lösungsweg skizziert. Dazu stellt das folgende Kapitel die aktuellen Probleme mit MBSE aus Sicht der Raumfahrtindustrie detailliert dar. Im dritten Kapitel wird ein Lösungsansatz präsentiert, mit dem die beschriebenen Probleme angegangen werden können. Das vierte Kapitel fasst die Probleme und den Lösungsansatz zusammen.

2 Aktuelle Probleme mit MBSE aus Sicht der Raumfahrtindustrie

In Teilen der Raumfahrtindustrie wird MBSE bereits angewendet [EMdK09], [dKEB10], [Eur12]. Allerdings müssen die folgenden Herausforderungen noch angegangen werden.

Diese liegen an der Schnittstelle der Themen Cyber Physical Systems und Big Data.

- Eine Vielzahl am Systementwicklungsprozess beteiligten Stakeholdern, wie Auftraggeber, direkt am System arbeitenden Engineering-Disziplinen, und Zulieferern
- Verwendung eigener Frameworks für die domänenspezifische Modellierung der Stakeholder
- Massiver Daten– und Wissensaustausch zwischen den beteiligten Stakeholdern
- Hohe Komplexität der Datenmodelle, die es erlauben, interdisziplinäre Abhängigkeiten zu erfassen und zu überprüfen
- Projektspezifische Anpassungen im Kontext geringer Stückzahlen innerhalb eines Projektes

In der Raumfahrt sind viele der zum Einsatz kommenden Systeme Prototypen, da die Einsatzzwecke oftmals sehr vielfälter Natur sind. Weiterhin werden die anfallenden Arbeiten oftmals geographisch voneinander getrennt erledigt. Als Beispiel sei die Rosetta Mission genannt, an der insgesamt 50 Firmen aus weltweit 14 Ländern beteiligt waren. Allein in Europa waren etwa 1000 Personen mit der Entwicklung und dem Bau von Rosetta beschäftigt. Es mussten 11 verschiedene wissenschaftliche Instrumente in die Raumsonde integriert werden, wozu charakteristische Systemparameter wie Dimensionen, Massen, Energiebedarf und Interfaces wie Befestigungspunkte und Kommunikationsprotokolle zwischen dem jeweiligen Hersteller eines Instrumentes und dem Hersteller der zentralen Komponenten der Raumsonde ausgetauscht werden mussten [FHT04].

Spezifisch für die Raumfahrt ist weiterhin, dass Tests unter realen Bedingungen, wie sie im Weltraum vorherrschen, nicht oder nur sehr eingeschränkt durchführbar sind. Weiterhin können kleinste Fehler bereits zum Scheitern einer Mission führen. Deshalb müssen die Systeme sehr fehlertolerant und teils redundant ausgelegt sein. Daraus resultiert ein noch höheres Maß an Komplexität.

2.1 Semantik auf Modellierungsebene

Die konzeptuelle Datenmodellierung dient als Grundlage für den Entwurf des im vorherigen Kapitels vorgestelltem Systemmodells. Dabei wird der Umfang der zu verwaltenden Daten und deren Abhängigkeiten untereinander von allen Beteiligten in einem konzeptuellen Datenmodell (engl. Conceptual Data Model (CDM)) erfasst. Das CDM fungiert dabei auch als Grundlage für einen Abgleich zwischen den Modellierern des CDMs und den beteiligten Personen aus den Engineeringdomänen. Weiterhin dient das CDM als Referenzimplementierung aller Werkzeuge, die später mit dem Systemmodell arbeiten werden [KOB07], [Oli07].

Bei, wie in der Raumfahrt üblich, projektspezifischen Zusammensetzungen von Domänen, Zulieferern und Kunden ist es deshalb essentiell, dass das CDM sowohl klar definierte

Grundbausteine vorgibt, als auch projektspezifische Anpassungen unterstützt, damit nicht für jedes Projekt ein neues CDM entworfen werden muss.

2.2 Semantik zur Laufzeit

Die Semantik, die in den Modellen der einzelnen Disziplinen, Zulieferern und Kunden steckt, muss zwischen diesen ausgetauscht werden. Dafür wird, wie im vorausgegangenen Abschnitt beschrieben, das CDM verwendet. Damit ergeben sich eine Vielzahl an Problemen, die im Folgenden erläutert werden:

Diskrepanz zwischen modellierter und implementierter Semantik Mit dem aktuellen Stand der Technik kann die vollständige Semantik der Daten nicht im CDM definiert werden, da entweder die verwendete Modellierungssprache semantisch nicht reich genug ist oder keine Generierungsmöglichkeiten für die definierten Bedingungen zur Verfügung steht. So ist Ecore beispielsweise semantisch schwach, bietet aber mit dem Eclipse Modeling Framework [SBMP08] eine starke Codegenerierungsplattform. Faktenbasierte Ansätze wie Object Role Modeling [HMM08], [Hal11] bieten zwar eine semantisch reichhaltigere Modellierung, aber die Transformation dieser Modelle in eine ausführbare Applikation ist bisher nur sehr eingeschränkt unterstützt [Hop14], [SV07], [PVN12]. Manchmal stehen auch Performanceansprüche an die zu generierende Applikation einer Transformation im Weg. Resultierend daraus wird die Semantik, die nicht aus dem CDM in die Implementierung übertragen werden konnte, direkt in der Implementierung umgesetzt. Sie ist somit nur implizit in der Implementierung vorhanden und nicht explizit im Modell spezifiziert.

Für den Nutzer eines MBSE–Werkzeugs ergibt sich somit eine Diskrepanz zwischen der Semantik, die bezogen auf das CDM erwartet wird, und der, die tatsächlich implementiert ist.

Datenaustausch zwischen domänenspezifischen Werkzeugen Viele Disziplinen setzen Standardsoftware für den modellbasierten Entwurf ein. Diese stammt von unterschiedlichen Herstellern, setzt auf verschiedenen Technologien auf, realisiert andersartige Paradigmen oder weist sonstige semantische Unterschiede auf. Aufgrund der Vielzahl von Disziplinen und externen Partnern im Raumfahrtbereich muss ein massiver Datenaustausch zwischen einer Vielzahl von heterogenen Werkzeugen stattfinden. Eine Übersicht über eine solche Infrastruktur ist schematisch in Abbildung 1 gegeben.

Der Datenaustausch ist allerdings durch die fehlende Anpassbarkeit der Standardsoftware problematisch. Deshalb wird in Teilen der Raumfahrtindustrie ein zentrales System eingesetzt und der Datenaustausch mit jedem domänenspezifischen Werkzeug muss in Form einer Modelltransformation vom applikationsspezifischen in das zentrale Datenmodell definiert werden. Bei der Umsetzung dieser Transformationen kann es zu semantischen Fehlinterpretationen einzelner Aspekte eines der beteiligten Datenmodelle kommen. Dadurch kann ein Import der Daten in das zentrale System und ein Export zum gleichen Werkzeug unproblematisch sein. Allerdings kann ein Import von Daten aus einem Werkzeug und

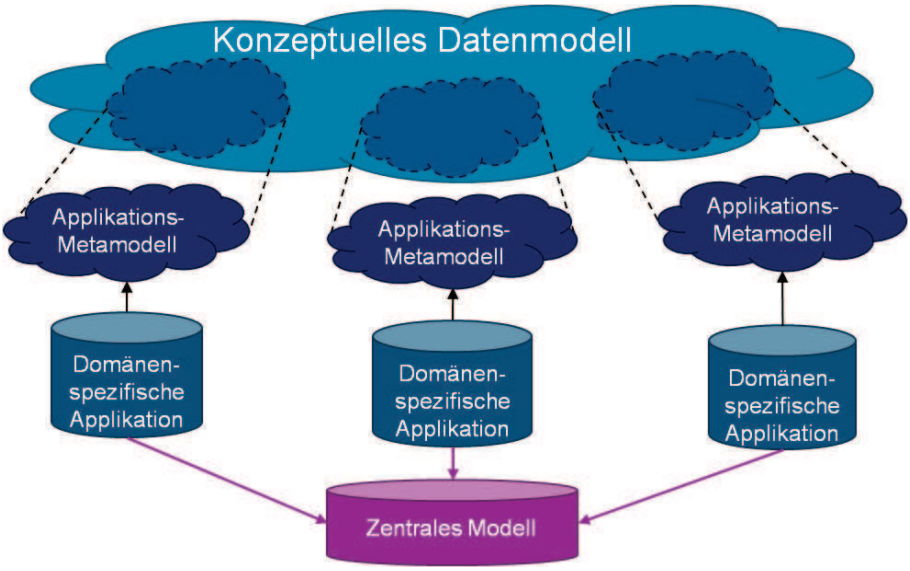

Abbildung 1: Anbindung domänenspezifischer Standardsoftware an eine zentrale Applikation. Jede der Domänen hat ihr eigenes Datenmodell welches vom jeweiligen Werkzeug implementiert wird. Die Summe aller systemübergreifend relevanten Daten ist im konzeptuellen Datenmodell festgehalten und bildet die Grundlage für das zentrale Model.

ein Export zu einem anderen Werkzeug in falsch interpretierten oder strukturierten Daten enden.

Ein Parameter ist mehr als ein Wert Die Komplexität der Datenstrukturen lässt sich anhand der zu betrachtenden Aspekte eines Parameters veranschaulichen. Dieser durchfährt im Ingenieursbereich oftmals eine Entwicklung von einer geschätzten Annahme hin zu einem Wert, der auf Daten basiert und immer weiter verfeinert wird, bis er letzendlich gemessen werden kann. Hierbei müssen sowohl Grenzwerte als auch Zugriffsrechte verwaltet werden. Des Weiteren müssen die einzelnen Werte eines Parameters jederzeit für alle bisherigen Phasen des Systemlebenszyklus zugreifbar sein. Außerdem werden Parameter teilweise kommentiert, um beispielsweise Beobachtungen, Wertherkunft oder Auffälligkeiten zu annotieren. Insbesondere Werte, die von Zulieferern stammen, müssen auf ihre Kompatibilität mit den in der Gesamtarchitektur festgelegten Grenzen überprüft werden und eventuelle Abweichungen müssen in beiderseitigem Übereinkommen korrigiert werden.

Datenverfeinerung während des Entwicklungsprozesses In der Raumfahrtindustrie ist der Systementwicklungsprozess in Phasen aufgeteilt, die jeweils einen fest definierten Systemabstraktionsgrad beschreiben. Die Daten einer Phase werden in der darauffolgenden Phase verfeinert. Im Bedarfsfall muss aber auch auf die Daten aus zurückliegenden Phasen zurückgegriffen werden, um beispielsweise die Auswirkungen von Änderungen zu

ermitteln. Außerdem können die Daten aus früheren Phasen unter Umständen für mehrere Projekte verwendet werden, wohingegen die Daten der späten Phasen projektspezifisch sind. Während sich die Daten von Phase zu Phase weiterentwickeln, müssen auch die Validierungsbedingungen angepasst werden, denn diese sind zum Teil ebenso phasenabhängig.

Die Ermittlung, welche Daten einer Phase in der nächstfolgenden auf welche Art und Weise verfeinert wurden, ist angesichts der Mannigfaltigkeit der verwendeten Datenmodelle und deren hochkomplexer innerer Struktur mit einer Vielzahl von Bedingungen keine triviale Aufgabe. Insbesondere dann, wenn eine Referenz aus der einen Phase in eine Vielzahl von Referenzen in der nächsten Phase aufgebrochen wird, um beispielsweise die Unterschiede der einzelnen Instanzen zu modellieren.

Projektspezifische Anpassungen des Datenmodells zur Laufzeit Im Raumfahrtbereich werden häufig nur Prototypen und Kleinstserien mit geringen Stückzahlen gefertigt. Dementsprechend sind kosteneffiziente, projektspezifische Anpassungen des Datenmodells zur Laufzeit notwendig. Dabei wird ein gewisser Rahmen durch das implementierte Datenmodell vorgegeben, welcher dann sowohl erweitert als auch beschnitten werden kann. Dabei beziehen sich die Erweiterungen auf Klassen, Attribute, Referenzen und auch die Vererbungshierarchie.

Multidomänendaten erfordern eine angepasste Visualisierung Die Daten, die aus einer einzelnen Disziplin stammen, können sehr gut durch die disziplinspezifischen Modelle dargestellt werden, die eine Analyse der Daten ermöglichen. Allerdings erfordert die Integration von Daten aus einer Vielzahl von Disziplinen eine passende Visualisierung. Diese sollte sowohl eine Einschränkung der darzustellenden Daten erlauben als auch ein automatisiertes Erstellen des passenden Layouts beinhalten.

Ontologische Ansätze zur Wissenserweiterung Die Datenintegration aus verschiedenen Quellen ermöglicht auch weitere Überprüfungen der Daten insbesondere bezüglich der Konsistenz zwischen den einzelnen Quellen. Somit kann beispielsweise eine Instanz im Datenmodell, die einen Wassertank im mechanischen Design darstellt, mit einer anderen Instanz im Datenmodell, die den gleichen Wassertank im thermischen Design repräsentiert, verbunden werden.

Aufgrund dieser Verbindungen ergeben sich sowohl Notwendigkeiten als auch Ausschlüsse, die im Datenmodell spezifiziert werden müssen, da zum Beispiel Aktoren und Sensoren per Definition disjunkt sein müssen. Ontologische Konzepte, die den Typ eines Objektes anhand der ihm zugewiesene Attribute bestimmen, sind Teil aktueller Forschungen [Hen12]. Sie wurden bisher unter anderem aufgrund der Komplexität, die sich aus der Vielzahl von möglichen Kombinationen ergibt, aber noch nicht implementiert.

Zentrales versus dezentrales Datenmanagement Für den Umgang mit den Daten zur Laufzeit ergeben sich zwei unterschiedliche Ansätze. Zum einen könnten die Daten durch eine zentrale Applikation verwaltet werden und zum anderen könnte jedes im MBSE–

Prozess involvierte Werkzeug die Verwaltung seiner Daten selbst organisieren. Zur Datenverwaltung zählen in beiden Fällen die Versionierung, das Definieren und Ausführen von Validierungsbedingungen, das Verwalten von projektspezifischen Anpassungen, das Bereitstellen von Daten für eine andere Applikation und Weiteres.

Bei einem zentralen Ansatz, wie er im Raumfahrtbereich aktuell angewendet wird, müssen die einzelnen Funktionen nur einmal realisiert werden und arbeiten auf dem zentralen Datenmodell. Des Weiteren werden Adapter benötigt, die die applikationsspezifischen Daten zwischen der spezifischen Form und der des zentralen Datenmodells transferieren. Diese realisieren den Austausch mit den Werkzeugen der involvierten Disziplinen und Partner. Bei einem dezentralen Ansatz müssen die Funktionen für jedes Datenmodell jeder beteiligten Applikation realisiert werden. Weiterhin muss der Datenaustausch zwischen allen involvierten Werkzeugen wechselseitig durch Adapter realisiert werden.

Der zentrale Ansatz bietet den Vorteil, dass neue Werkzeuge oder Partner nur eine Schnittstelle mit dem zentralen Datenmodell haben und nicht wie beim dezentralen Ansatz mit jedem Partner haben, mit dem Daten ausgetauscht werden müssen. Weiterhin bietet der zentrale Ansatz den Vorteil einer zentralen Datenverwaltung, wodurch das Abrufen von Daten nicht an die jeweiligen Quellen gebunden ist. Ebenfalls kann sichergestellt werden, dass alle Daten früherer Entwicklungsphasen gegebenenfalls wieder zur Verfügung stehen.

3 Entwicklung eines Lösungsansatzes

Die im vorherigen Kapitel beschriebenen Probleme sind miteinander verzahnt, so dass die unabhängige Betrachtung einzelner Lösungen für jedes der identifizierten Probleme nicht zu einer zufriedenstellenden Gesamtlösung führt. Vielmehr müssen die unterschiedlichen Dimensionen der Problemstellung identifiziert und die jeweiligen Teilbereiche analysiert werden. Die dadurch gewonnenen Erkenntnisse dienen zum einen dem Problemverständnis und zum anderen bilden sie die Grundlage für die Ausarbeitung einer Lösungsstrategie.

Um die beschriebenen Probleme anzugehen werden vier Dimensionen betrachtet:

Probleme In dieser Dimension werden die aktuellen Unzulänglichkeiten bei der Umsetzung von MBSE detailliert erfasst.

Funktionen Diese Dimension umfasst alle notwendigen Funktionen für die Umsetzung von MBSE.

Forschungsstand In dieser Dimension soll der aktuelle Stand der Forschung und Technik in Bezug auf die zu realisierenden Funktionen und die auftretenden Probleme eingeordnet werden.

Testumgebung Diese Dimension umfasst das Vorgehen, um einzelne Blöcke von Problemen in der Tiefe zu analysieren und mögliche Lösungen zu evaluieren.

Für jeden der zu untersuchenden Teilbereiche müssen vier zentrale Punkte mithilfe der Testumgebung herausgearbeitet werden:

1. Der aktuelle Stand der Technik in der Industrie muss erfasst werden.
2. Es müssen ein oder mehrere Lösungsansätze umgesetzt werden und miteinander verglichen werden.
3. Dabei ist es wichtig, die Lösungen sowohl isoliert als auch im Gesamtkontext zu betrachten, um mögliche Abhängigkeiten respektive spezielle Konfigurationen ausfindig zu machen, unter denen eine bestimmte Lösung schlechter oder besser funktioniert.
4. Die vorhanden Lösungen müssen im Kontext einer Gesamtlösung gegeneinander abgewogen werden.

Die im vorherigen Kapitel beschriebenen Probleme können in mehrere Blöcke aufgeteilt werden, die zwar nicht unabhängig voneinander sind, aber jeweils ihren individuellen Schwerpunkt haben. Diese Blöcke werden im Folgenden detailliert beschrieben, wobei die vorher genannten Dimensionen im jeweiligen Kontext erläutert werden.

3.1 Analyse des Status quo

Das Fundament für die Entwicklung von Lösungen für die in Kapitel 2 beschriebenen Probleme wird mithilfe einer Testumgebung gelegt, die den derzeitigen Stand widerspiegelt. Darauf aufbauend können dann Lösungen für die identifizierten Probleme entwickelt und evaluiert werden.

Um den derzeitigen Stand abzubilden, muss ein zu definierendes Datenmodell entworfen und implementiert werden. Des Weiteren müssen alle notwendigen Funktionalitäten, die für die Durchführung von MBSE notwendig sind, umgesetzt werden. Außerdem müssen die Schnittstellen zu externen Werkzeugen simuliert werden, um später die Qualität des Datenaustausches bewerten zu können.

Diese Testumgebung dient weiterhin dazu Evaluierungskriterien zu definieren, sodass die Lösungen, die entwickelt werden, auch bewertet werden können und somit untereinander vergleichbar sind. Dieser Prototyp dient weiterhin dazu, einen repräsentativen Testdatensatz zu entwickeln. Die Evaluierungsergebnisse, die mit dieser Testumgebung basierend auf dem zu definierenden Referenzdatensatz erstellt wurden, bilden die Grundlage und stellen eine untere Schranke dar, die von allen Lösungen mindestens erreicht werden muss. Nur dann sind diese als besser zu bewerten als der aktuelle Stand.

3.2 Semantisch starke konzeptuelle Datenmodelle zur Laufzeit

Das Ziel dieser Testumgebung ist, die Auswirkungen eines semantisch starken konzeptuellen Datenmodells auf die Applikation zu untersuchen. Dabei sind auch die Auswirkungen auf Codegenerierungsmechanismen zu analysieren. Die Durchsetzung der im Datenmodell definierten Bedingungen ist der Schlüssel zum Erfolg für dieses Szenario. Bestimmte Realisierungen könnten aus Performancegründen nicht praktikabel sein, da sich alles im Big Data Umfeld, mit einer großen Anzahl an Bedingungen und Instanzen, auf denen diese ausgewertet werden müssen, bewegt. Mit den gewonnenen Daten soll unter anderem evaluiert werden, was geeignete Technologien und Beschreibungsformen für semantisch starke Bedingungen sind und inwiefern sich neue Implementierungen bestehender Technologien finden lassen, die hinreichende Performance bieten. In diesem Hinblick sind z.B. die partielle oder ereignisbasiert Auswertung von Bedingungen interessant.

3.3 Grenzen von Datenmodellanpassungen zur Laufzeit

Wie in Kapitel 2 beschrieben, sind projektspezifische Anpassungen des Datenmodells innerhalb der Applikation essentiell. In dieser Testumgebung soll analysiert werden, wo die Grenzen für solche Anpassungen liegen und wie diese umgesetzt werden können.

Dazu muss eine Modellierungsplattform für die Anpassungen zur Verfügung gestellt werden, um eine grafische Repräsentation der Anpassungen für Diskussionsrunden und für die Dokumentation zu ermöglichen. Dabei muss auch sichergestellt werden, dass die Anpassungen konform zum statischen Teil des Datenmodells sind. Anschließend muss für die definierten Anpassungen eine Codegenerierung angeboten werden, um das vorhandene Werkzeug mit den definierten Anpassungen zu erweitern.

Des Weiteren soll das Verfeinern von Daten in dieser Testumgebung behandelt werden. Dabei steht vor allem das Abbilden von Daten einer Phase des Systementwicklungsprozesses auf eine vorherige Phase im Mittelpunkt. Dadurch hat man jederzeit Zugriff auf die Daten aller vorausgegangenen Phasen und kann Auswirkungen von Veränderungen analysieren.

3.4 Integration von Applikationen bei zentralen und dezentralen Ansätzen

Diese Testumgebung behandelt zwei Aspekte. Zum einen soll der Datenaustausch analysiert und verbessert werden und zum anderen soll der derzeitige zentrale Ansatz mit einem dezentralen Ansatz verglichen werden.

Der Datenaustausch zwischen einem domänenspezifischen Werkzeug und der zentralen Applikation erfordert eine bidirektionale Transformation der Daten zwischen domänenspezifischen Datenmodellen und zentralem Datenmodell. Dabei ist es essentiell, dass keines der Datenmodelle Raum für Interpretationen lässt, da dies sonst bei Nutzung der Daten

durch eine dritte Applikation zu Fehlern führt. In diesem Zusammenhang sollte auch der Einsatz eines Standards wie OSLC [Wor10] für die Datenübertragung untersucht werden. Außerdem sollte die Software, die die Transformation durchführt, dahingehend erweitert werden, dass jederzeit ermittelt werden kann, welche Daten mit welchem Werkzeug ausgetauscht wurden. Dies würde feingranulare Änderungsmitteilungen für die angebundenen Werkzeuge ermöglichen. Damit könnten Simulationen vereinfacht werden, indem unter Umständen diese nur für Teilbereiche eines Systems neu ausgeführt werden müssen.

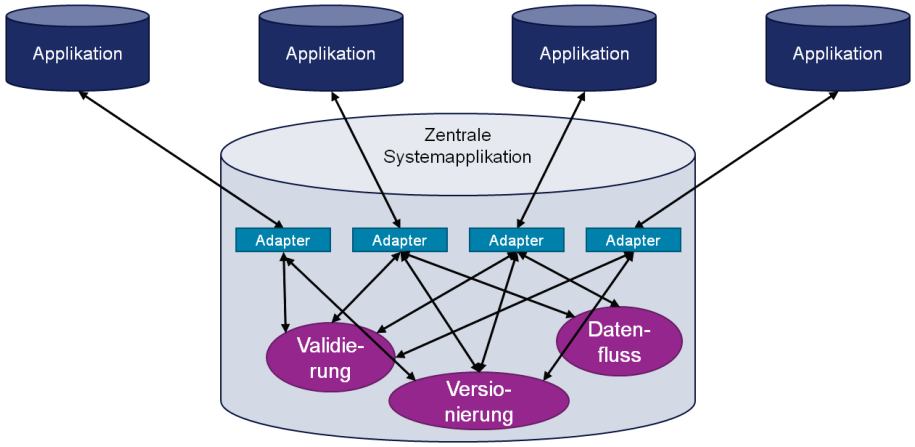

Abbildung 2: Ein zentraler Ansatz für MBSE wie er im Moment verwendet wird. Die Werkzeuge sind mit der zentralen Applikation verbunden und die Funktionalitäten sind beispielhaft abgebildet.

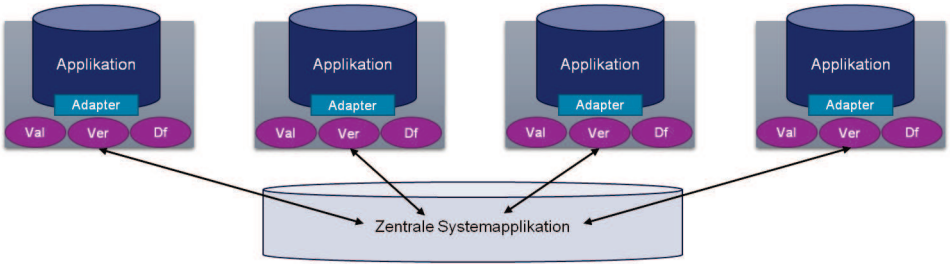

Abbildung 3: Ein dezentraler Ansatz, der die Funktionalitäten spezifisch für jedes Datenmodell eines Werkzeugs umsetzt und die zentrale Applikation nur ein Datencontainer ohne weitere Funktionalitäten ist. Die beispielhaft dargestellten Funktionalitäten sind Validierung (Val), Versionierung (Ver) und Datenfluss (Df).

Der zweite Teil dieser Testumgebung widmet sich der Frage, welcher Ansatz besser ist – zentral oder dezentral. Dazu muss zuerst ein dezentraler Ansatz umgesetzt werden. Anschließend können die Vor– und Nachteile beider Ansätze miteinander verglichen werden. Der zentrale Ansatz ist in Abbildung 2 und der dezentrale Ansatz ist in Abbildung 3 illustriert.

3.5 Datenorganisation im Hinblick auf den Datenfluss

In dieser Testumgebung steht das Management der Daten im Vordergrund. Als erstes soll untersucht werden, wie sich der Datensatz in mehrere Teile aufbrechen lässt. Dies ist essentiell, da ein Datensatz, der aus einer einzelnen Datei bestehen würde, nicht handhabbar ist, da er viel zu groß für den Speicher wäre. Dabei können verschiedene Aspekte betrachtet werden. Dazu zählen eine Aufteilung bezüglich der Produktstruktur, des Datenflusses oder der Art der Daten (Sensor, Aktor, Konfiguration, Realisierung,...).

Ein weiterer Punkt, der in dieser Testumgebung betrachtet werden soll, ist das gemeinsame Nutzen von Attributen durch mehrere Objekte. Dabei sollen die Vor– und Nachteile zweier möglicher Strategien herausgearbeitet werden: Zum einen könnte jedes Attribut nur virtuell geteilt werden. Zum anderen könnte der Wert für jeden, der darauf zugreift, dupliziert werden und Änderungen müssten synchronisiert werden.

Des Weiteren soll das Management der Metadaten beleuchtet werden. Das sind die Daten, die für die Durchführung des Prozesses notwendig aber nicht Teil des zu entwickelnden Systems sind.

3.6 Visualisierung heterogener Daten

Das Vereinigen von Daten aus verschiedenen Werkzeugen erfordert auch neue Methoden der Darstellung, welche in dieser Testumgebung näher betrachtet werden sollen.

Klassische tabellarische Ansichten ermöglichen nur geschulten Nutzern neue Zusammenhänge in den Daten zu erkennen. Für einen gezielten Erkenntnisgewinn sind Modelle viel besser geeignet. Die Darstellung der Daten in den einzelnen Werkzeugen kann sehr unterschiedlich sein. Doch wie soll die Visualisierung für die vereinigten Daten aussehen? Gibt es Grenzen? Welche Visualisierungsmethoden und –stile sind hilfreich? Diese Fragen müssen beantwortet werden um passende Visualisierungen zu finden.

Dabei sollte dem Nutzer auch die Möglichkeit geboten werden, die darzustellenden Referenzen und Attribute einzugrenzen, um bestimmte Aspekte eines Systems bewusst auszublenden oder hervorzuheben. Für die automatische Erzeugung der Visualisierung dieser Modelle ist ein automatisierter Layoutmechanismus essentiell.

4 Zusammenfassung

Modellbasiertes Systems Engineering (MBSE) wird in Teilen der Raumfahrtindustrie bereits erfolgreich eingesetzt. Die auftretenden Probleme und Schwächen sind allerdings nicht von der Hand zu weisen und müssen angegangen werden, um die Vorteile, die MBSE bietet, besser ausnutzen zu können.

Es gibt bereits Ansätze, die die Probleme auf konzeptueller Ebene angehen [Hen12],

[Hop14]. Allerdings treten viele Schwachstellen erst zur Laufzeit zu Tage. Die zentralen Aspekte sind:

- Auftritt einer semantischen Diskrepanz zwischen einem konzeptuellen Modell und der implementierenden Applikation
- Datenaustausch mit domänenspezifischen Standardwerkzeugen im Big Data Umfeld
- Mehrschichtige und komplexe Parameter
- Projektspezifische Anpassungen des Datenmodells zur Laufzeit
- Visualisierung heterogener Daten aus verschiedenen Quellen
- Wissenserweiterung und logische Ausschlüsse basierend auf vorhandenen Daten
- Zentrales versus dezentrales Datenmanagement

Um die genannten Probleme nachhaltig zu lösen reicht eine isolierte Betrachtung einzelner Probleme nicht aus, da diese miteinander verzahnt sind. Deshalb wird in diesem Beitrag ein Ansatz vorgestellt, der das Gesamtproblem in einzelne Teilbereiche herunterbricht, die jeweils im Gesamtkontext betrachtet werden. Zentrales Element des Lösungsansatzes sind kontrollierbare Testumgebungen, die eine Evaluierung des Ausgangsproblems und der entwickelten Lösungen ermöglichen. Dabei gilt es zunächst den aktuellen Stand zu erfassen und die Probleme zu verstehen. Anschließend können die einzelnen Teilbereiche analysiert und potentielle Lösungen entworfen sowie evaluiert werden. Dabei müssen auch untereinander auftretende Abhängigkeiten von Lösungsansätzen analysiert werden.

Literatur

[dKEB10] Hans Peter de Koning, Harald Eisenmann und Massimo Bandecchi. Evolving standardization supporting model based systems engineering. 2010.

[EMdK09] Harald Eisenmann, Juan Miro und Hans Peter de Koning. MBSE for European Space-Systems Development. *INCOSE Insight*, 12(4):47–53, 2009.

[Eur12] European Space Agency (ESA), http://vsd.esa.int/. *Virtual Spacecraft Design*, 2012.

[FGS07] Sanford Friedenthal, Regina Griego und Mark Sampson. INCOSE Model Based Systems Engineering (MBSE) Initiative. *INCOSE MBSE Track*, 2007.

[FHT04] Horst Fiebrich, JE Haines und Ferdinando Tonicello. Power System Design of the Rosetta Spacecraft. *AIAA Paper*, 5535:2004, 2004.

[Hal11] T. Halpin. Fact-Orientation and Conceptual Logic. In *Enterprise Distributed Object Computing Conference (EDOC), 2011 15th IEEE International*, Seiten 14–19. IEEE, 2011.

[Hen12] Christian Hennig. Evaluating Ontologies for Use in Model-Based Systems Engineering. Diplomarbeit, Technische Universität München, 2012.

[HMM08] T.A. Halpin, A.J. Morgan und T. Morgan. *Information modeling and relational databases*. Morgan Kaufmann, 2008.

[Hop14] Tobias Hoppe. Fact-Based Modeling for Eclipse Applications. Master's thesis, Hasso-Plattner Institut an der Universität Potsdam, 2014.

[KOB07] John Krogstie, Andreas Lothe Opdahl und Sjaak Brinkkemper. *Conceptual Modelling in Information Systems Engineering*. Springer, 2007.

[Oli07] Antoni Olivé. *Conceptual modeling of information systems*. Springer, 2007.

[PVN12] B. Piprani, S. Valera und G. M. Nijssen. Metamodel for Fact Based Information Model Registration. Bericht, Metadata Standards ISO/IEC JTC1 SC32 WG2, 2012.

[SBMP08] D. Steinberg, F. Budinsky, E. Merks und M. Paternostro. *EMF: eclipse modeling framework*. Addison-Wesley Professional, 2008.

[SV07] S. Sosunovas und O. Vasilecas. Tool-supported method for the extraction of OCL from ORM models. In *Business Information Systems*, Seiten 449–463. Springer, 2007.

[Wor10] OSLC Core Specification Workgroup. OSLC core specification version 2.0. *Open Services for Lifecycle Collaboration, Tech. Rep*, 2010.