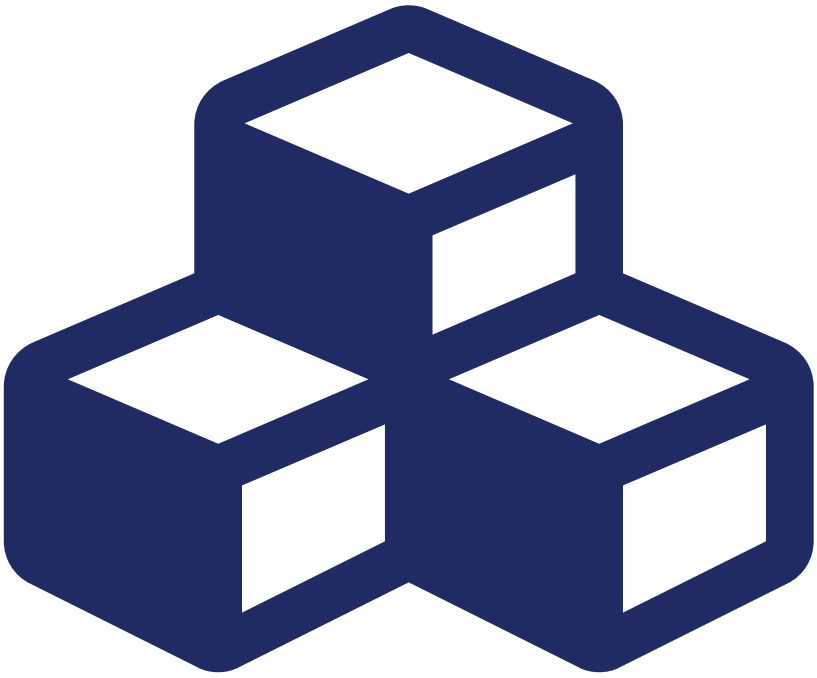
TEIL 2

Modellieren, aber richtig!

Wie Entscheider die richtige Modellierungsstrategie finden und im Unternehmen installieren können.

Whitepaper



Dr. Horst Kargl
SparxSystems Software GmbH
www.sparxsystems.de
vendor@sparxsystems.eu

ENTERPRISE ARCHITECT

Inhalt

Einleitung

Die Entwicklung moderner Software- und durch Software unterstützter Systeme wird immer aufwendiger und komplexer. Viele Unternehmen entscheiden sich daher, ihren Entwicklungsprozess zu überarbeiten. Die Verwendung grafischer Modellierungssprachen und dazu passender Werkzeuge hat schon viele Unternehmen dabei unterstützt, diese Komplexität in den Griff zu bekommen, andere allerdings nicht. Warum?

Um dieser Frage auf den Grund zu gehen, habe ich zwei „White Paper" erstellt, die die einzelnen Aspekte bei der Einführung modellbasierter Software- und Systementwicklung im Unternehmen näher beleuchten:

- Modellieren, aber richtig! Teil 1 - Über die Vorteile und den Nutzen der modellbasierten Software und Systementwicklung
- Modellieren, aber richtig! Teil 2 - Wie Entscheider die richtige Modellierungsstrategie finden und im Unternehmen installieren können

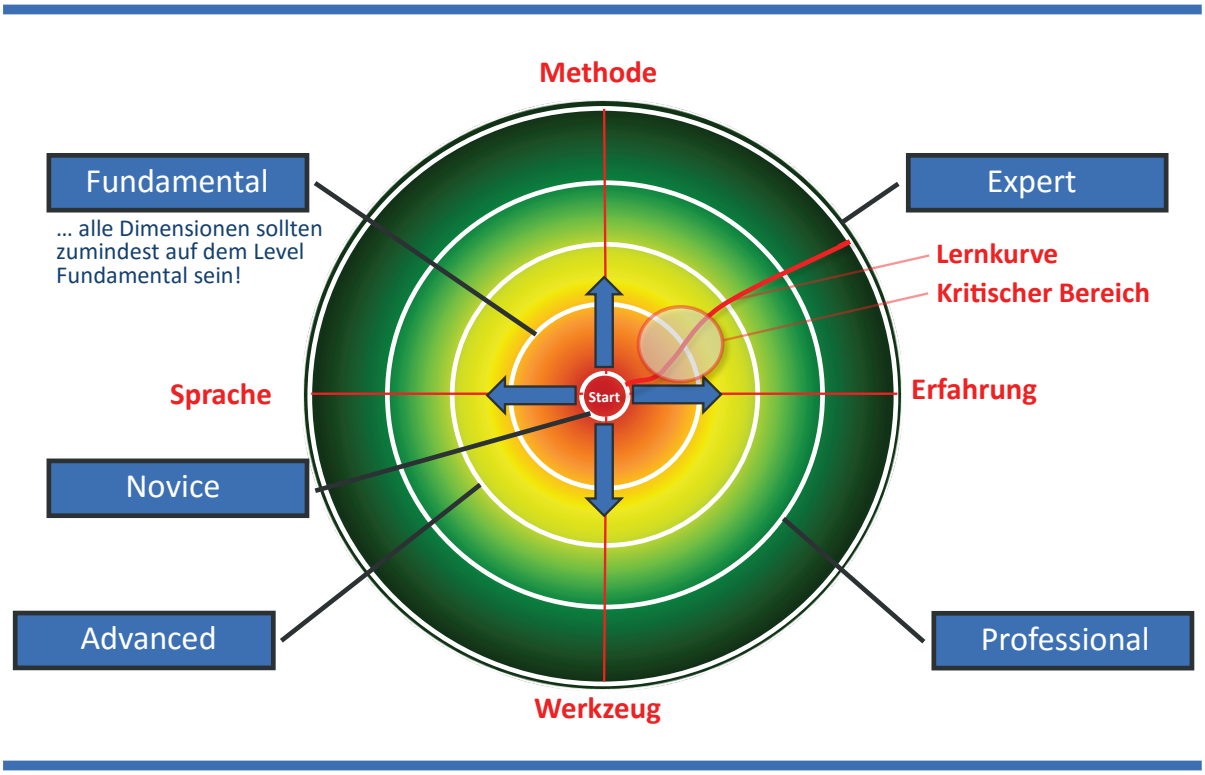
Im hier vorliegenden Teil 2 ist meine Erklärung für das Scheitern bei der Einführung grafischer Modellierungssprachen im Unternehmen, dass man Komplexität einfach nicht wegzaubern kann. Wenn etwas komplex ist, ist es auch das Beschreiben einer Lösung. Dann hilft eine Modellierungssprache nur, wenn man gut damit umgehen kann und sie richtig eingesetzt wird.

Neben der Modellierungssprache selbst existieren viele weitere Dimensionen, die beachtet werden müssen, um mit dem Modellierungsansatz erfolgreich zu sein. Dies gilt vor allem, wenn man im Team an der Beschreibung eines Systems arbeitet.

So ist es etwa ein oft begangener Fehler, dass man seine Mitarbeiter zu früh sich selbst überlässt. Ein dreitägiger C++ Kurs und eine Einführung in die bevorzugte Entwicklungsumgebung macht aus einem Mitarbeiter noch lange keinen Meisterentwickler. Als Beispiel: Um eine kontinuierliche Integration (Continues Integration) zu erreichen, und nach jedem Check-in in das Versionskontrollsystem einen lauffähigen Build am Build-Server zu haben, benötigt man neben den Kenntnissen der Programmiersprache und einer praxistauglichen Architektur auch Codierungsrichtlinien. Man muss also einer Architektur folgen und bestimmte Prozesse einhalten. Beim Modellieren verhält es sich ähnlich.

Damit die Modellierung im Unternehmen aber auch Früchte tragen kann und nicht nur als zusätzlicher Aufwand empfunden wird, sind einige Schritte notwendig, die wir in diesem „White Paper" im Detail diskutieren werden.

Einen Modellierungsansatz im Unternehmen einführen



Wir haben inzwischen viel über Modelle und deren Vorteile gehört und welche Voraussetzungen erfüllt sein müssen, damit wir einen Modellierungsansatz auch gewinnbringend aufsetzen können. Welche Schritte sind nun aber konkret durchzuführen und welche Möglichkeiten bietet Enterprise Architect, um mit der Einführung eines Modellierungsansatzes erfolgreich zu sein?

Mein Tipp: Es sind viele Schritte, die zum Erfolg führen. Ein notwendiger Anfang ist dabei der Kauf eines Modellierungswerkzeugs und die Einübung in dessen Bedienung. Wenn wir von Modellierung sprechen, dann geht es um mehrere Dimensionen: Modellierungssprache, Werkzeug, Methode und zuletzt die Erfahrung im Umgang mit all dem. Jede dieser Dimensionen ist ein eigenes Thema. Um zu funktionieren müssen letztlich alle Dimensionen wie Zahnräder ineinandergreifen. Dies macht den Anfang für Einsteiger anspruchsvoll, weil gleich drei Dimensionen gleichzeitig zu lernen sind.

Die Erfahrung kommt dann praktisch von selbst mit der Praxis. Aus Erfahrung weiß ich, dass nicht jede Person im Team in jeder dieser Dimensionen Expertenstatus erlangen kann. Praktikabler ist es, einen geeigneten Modellierungsansatz für das Unternehmen mit Hilfe eines Experten in einer kleineren Gruppe fortgeschrittenen Modellierer zu erarbeiten.
Das Ergebnis dieses ersten Schrittes ist ein Referenzmodell, das sich mit der Zeit weiterentwickelt und als Grundlage für Schulungen und Anpassungen dient.
Mein Tipp: Wird im Team modelliert, sollte das gesamte Team in all den Dimensionen zumindest auf dem „Advanced Level“ (siehe Grafik) sein.

Wie finde ich heraus, was ich wirklich brauche?

Auch wenn alle Teammitglieder Modellierungs-Experten sind und über langjährige Erfahrung im Umgang mit modellbasierten Ansätzen verfügen, ist es notwendig, sich über die konkrete Vorgehensweise mit dem ge-

wählten Werkzeug Gedanken zu machen.
Um den mit der Modellierung zunächst verbunden Mehraufwand zu rechtfertigen, ist es wichtig, die Einführung gut zu planen und die Mitarbeiter gezielt zu unterstützen. Nur so lassen sich die Vorteile der Modellierung schnell und effizient nutzen.

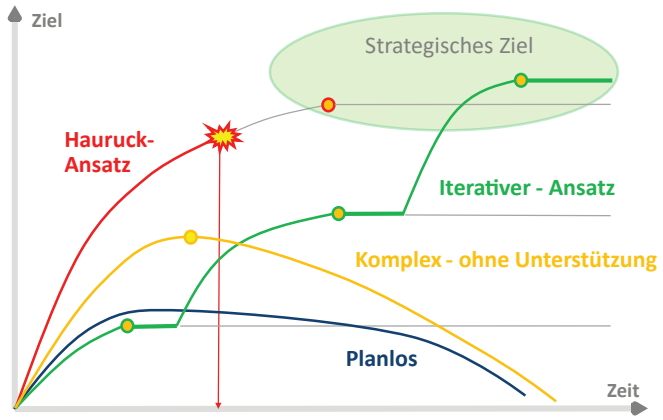
Mögliche Szenarios

Szenario 1: Ohne Planung

Bei der Einführung von Modellierungsansätzen musste ich leider schon oft erleben, dass Mitarbeiter in mehrtägigen Kursen zwar den Umgang mit dem Modellierungswerkzeug und die favorisierte Modellierungssprache gelernt haben, aber anschließend einfach sich selbst überlassen wurden. Selbst wenn die Teilnehmer enthusiastisch waren und schon konkrete Vorstellungen über den Einsatz der Modellierung hatten, ebbt bei diesem Vorgehen die Bereitschaft zur weiteren Entwicklung meist rasch ab. Die Modelle werden dann oft nur noch als „hübsche Bilder“ betrachtet und der erwartete Nutzen der Modellierung stellt sich nicht ein.
In diesem Szenario findet sich im Unternehmen niemand, der das Thema Modellierung strukturiert angeht, konkrete Ziele definiert und die notwendigen Modellstrukturen und Richtlinien (das Referenzmodell und das Metamodell) erarbeitet. So entwickeln sich nur Insellösungen, die zwar oft gut durchdacht aber miteinander nicht kompatibel sind. Typisch ist in diesem Szenario ist, dass ein weitgehend unerfahrenes Team mit großen Einsatz viele sehr detaillierte Modelle erstellt, die aus Zeitmangel nicht aktualisiert und damit wertlos werden. Durch die Desorganisation wird hier die Modellierung als Qual empfunden, niemand kennt sich in den Modellen der anderen aus. Auch wird zu viel Zeit damit verschwendet, für gegebene Probleme ein passendes Modell zu finden.

Szenario 2: Komplex und ohne Unterstützung

In diesem Szenario wird zwar ein Ziel gesetzt und dieses auch mit konkreten Vorgaben und Regeln versehen. Obwohl also klar vorgegeben ist, wie das Ziel erreicht werden soll - wie hat das Modell auszusehen und wann soll welche Information modelliert werden – ist dieser Prozess aber zum gegeben Zeitpunkt zu kompliziert aufgesetzt. Die beteiligten Personen haben keine Zeit, sich in die Struktur einzuarbeiten und arbeiten nicht oft genug mit dem Modell, um die einzelnen Arbeitsschritte und Ansätze zu verinnerlichen.



Dieser Ansatz ist ähnlich zum iterativen, mit einem wesentlichen Unterschied. Hier ist nicht nur das Ziel mit den Regeln vorgegeben, sondern es wird auch ein Belastungstest mit dem Ziel durchgeführt, nötigenfalls den Ansatz bzw. das Werkzeug anzupassen, um ohne großen Aufwand den gewünschten Mehrwert aus den Modellen zu gewinnen.

Hauruck-Ansatz: Ein oft zum Scheitern verurteilter Ansatz ist die Suche nach einem umfassenden Master-Ansatz, mit dem alle Probleme gelöst werden sollen. Dazu wird eine kleine Gruppe mit der Ausarbeitung des Ansatzes beauftragt. Aufgrund fehlender Erfahrung und keiner Erprobung in der Praxis wird der gefundene Modellierungsansatz von den Kollegen nicht akzeptiert, weil sie sich nicht wieder finden und bereits eine eigene, für sie praktische Arbeitsweise entwickelt haben (siehe „Planlos“).
Ein allumfassender Ansatz ist meist sehr kompliziert und erfordert großen Einarbeitungsaufwand der Mitarbeiter (Hohe Lernkurve). Wird die Methode nicht durch Automatismen und Validierungen unterstützt (wie beim Ansatz „Komplex ohne Unterstützung“) ist die Frustration der Mitarbeiter schnell sehr hoch, weil sie den komplizierten Regeln dieses Ansatzes nicht gewachsen sind.
Die oben beschriebenen Probleme sind darüber hinaus eher idealisiert: In der Regel scheitert das Projekt

schon, bevor es überhaupt ausgerollt werden konnte. Und zwar einfach deshalb, weil der Ansatz aufgrund von äußeren Umständen nie fertiggestellt werden kann. Damit wird viel Zeit und Geld vergeudet, ohne jemals einen Nutzen daraus zu ziehen.

Iterativ: Beim Iterativen Ansatz macht man sich (wie beim „Hauruck-Ansatz“ und „Komplex – ohne Unterstützung“) zuvor Gedanken darüber, welches Ziel man erreichen möchte. Dabei unterscheiden wir allerdings in große strategische Ziele und Ziele, die auch innerhalb weniger Monate erreicht werden können. Der iterative Ansatz ist vergleichbar mit dem Ansatz „Komplex ohne Unterstützung“. Der Unterschied liegt in der Konzentration des Modellinhalts auf die allerwichtigsten Fragen, die man mit dem Modell beantworten möchte. Hat man ein kurzfristiges Ziel erreicht, wird dieses in der Praxis etabliert und durch einen Belastungstest evaluiert. Erweist sich der Ansatz als nützlich, kann über das nächste kurzfristige Ziel nachgedacht werden. Sonst wird der aktuelle Ansatz solange angepasst, bis sich ein Mehrwert ergibt.
Dabei ist darauf zu achten, dass der Modellierungsansatz für jeden Beteiligten Nutzen und nicht nur zusätzlichen Aufwand erzeugt.
Konfigurationen und Automatismen erleichtern den Umgang mit dem Werkzeug. Validierungen helfen sicherzustellen, dass keine Modellierungsfehler begangen werden und das Modell nicht zerstört wird. Assistenten dienen dazu, komplizierte Schritte zu vereinfachen.

Ein Iterativer Einführungsprozess

Ein iterativer Ansatz ist also zu bevorzugen, da man schnell Feedback aus der Praxis erhält und gegebenenfalls den Ansatz adaptieren kann. Um nun die langfristigen strategischen Ziele zu definieren und einen Ansatz für die kurzfristigen Ziele zu erhalten, schlage ich den nachfolgenden Zyklus vor. Dieser Prozess wird im Sinne der Agilität möglichst rasch durchlaufen, um die Lösungsansätze für die gesteckten Ziele praktisch überprüfen zu können. Das verhindert eine Entwicklung in die falsche Richtung, da das Team laufend Feedback gibt und sich in den getroffenen Entscheidungen wiederfindet. Dass der iterative Ansatz also von Beginn an vom Team mitgetragen wird, verbessert sowohl die Akzeptanz als auch die Qualität!

- Experten-Workshop: Zuerst sollten die Mitarbeiter mit der größten Modellierungserfahrung und diejenigen, die wissen, welche Fragen mit dem Modell beantwortet werden sollen, an einem Expertenworkshop teilnehmen. Dieser gibt einen Überblick über alle Modellierungsaspekte und -möglichkeiten. Sitzen in einem Experten-Workshop Modellierungsskeptiker bzw. Mitarbeiter, die noch sehr wenige Berührungspunkte mit Modellierung hatten, werden diese oft mehr von den Möglichkeiten abgeschreckt, als dass sie ein großes Potenzial darin sehen könnten. Daher sollten in einem Expertenworkshop primär die „Fahnenträger“ sitzen oder die, die es werden möchten. Als Ergebnis erhält man Grundlagen-Wissen, das im nächsten Schritt beim Erstellen des Beispielprojekts hilfreich sein wird.

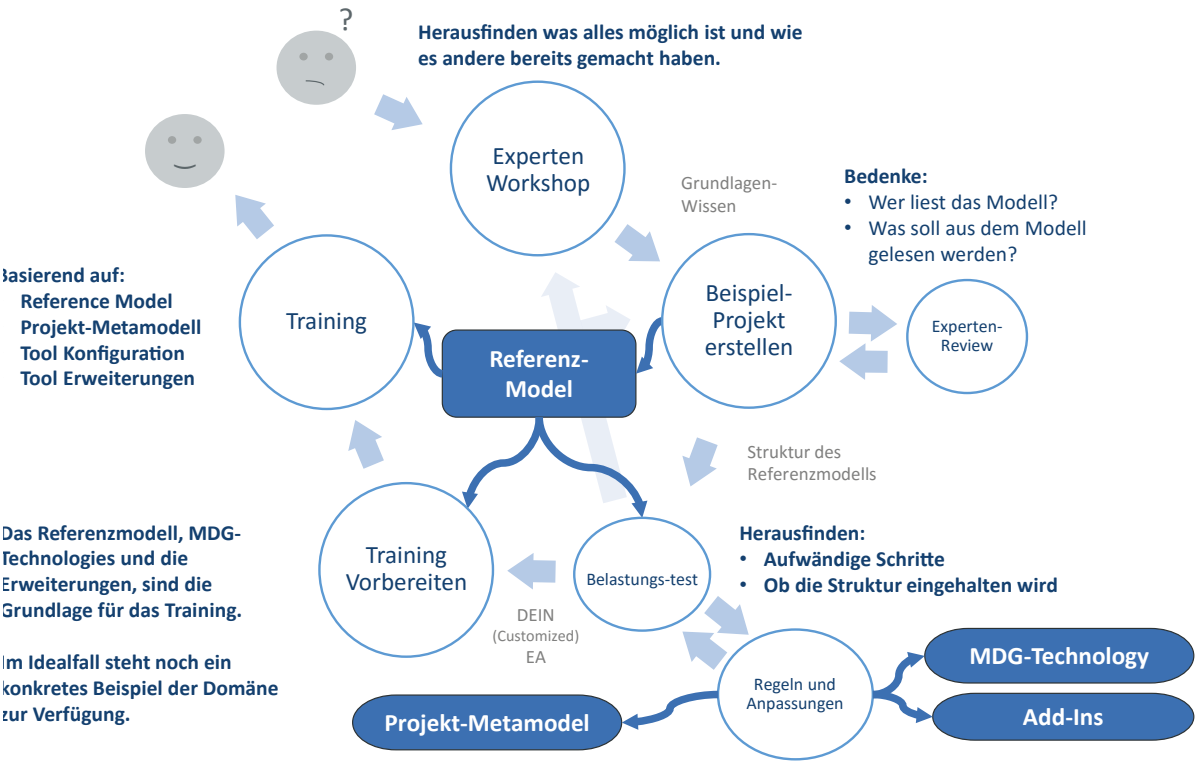
- **Beispielprojekt erstellen**: Basierend auf den gelernten Grundlagen und vermittelten Erfahrungen wird ein kleines, überschaubares Projekt erstellt. Dabei werden alle Informationen bespielhaft abgebildet, die mit dem Modell beantwortet werden sollen.
    - Experten-Review: Die Erstellung des Beispielprojektes kann von einem Experten begleitet werden, um eine optimale Abbildung der Informationen in ein Modell zu gewährleisten, ohne dabei zu viel Zeit durch mehrmalige Iterationen zu verlieren. Dabei sollten auch Aspekte wie die Generierung von Dokumenten, Verwaltung von Modellen und Modell-Versionen, Generierung von Code, etc. Berücksichtigung finden. **Als Ergebnis erhält man ein Beispielmodell (Referenzmodell) und damit das wichtigste Artefakt des Modellierungsansatzes.**

- **Belastungstest**: Um die Arbeit mit dem nun gefundenen Modellierungsansatz reibungsloser und flüssiger zu gestalten, wird in diesem Schritt die Arbeitsweise in einer Art Belastungstest überprüft. Dafür wird der Modellierungsansatz in eine etwas erweiterte Runde getragen, um Feedback über die Vorgehensweise zu bekommen bzw. ob bestimmte Informationen fehlen oder die Arbeitsweise als umständlich empfunden wird.
    - Regeln und Anpassungen: Werden bei Erstellung oder Auslesung des vorgeschlagenen Referenzmodells aufwendige Schritte gefunden, lassen sich diese durch eine eigene MDG-Technologie erleichtern. Sie enthält Konfigurationen, die die Arbeit erleichtern bzw. vereinheitlichen. Darüber hinaus lassen sich auch Automatismen und für den eigenen Ansatz erforderliche Funktionen als Erweiterungen (Add-ins) erstellen.
- Training vorbereiten: Auf der Basis des Referenzmodells sowie des Projekt-Metamodells und der Anpassungen im Werkzeug sollte nun ein Training folgen, in dem genau auf die Bedürfnisse der Gruppe eingegangen wird.
- Training basierend auf dem Referenz-Modell: Das Training basiert auf dem Referenzmodell und dem daraus abgeleiteten Projekt-Metamodell und fokussiert auf die Bedürfnisse der Gruppe. Diese praktiziert dabei die zuvor erarbeitete und bereits überprüfte Arbeitsweise an konkreten Beispielen aus der Domäne.

Das Referenzmodell

Das Referenzmodell umfasst viele Aspekte. Wie der Name schon sagt, handelt es sich um ein Modell-Repositorium, mit dessen Hilfe ein Verständnis über den gewünschten Modellierungsansatz entsteht und das als Basis für weitere Projekte dient. Das aktuell fertiggestellte Referenzmodell enthält alle notwendigen Konfigurationen und Anpassungen, um das gewünschte Ergebnis zu erreichen.

Der grundlegende Gedanke eines Referenzmodells ist es, die oben erwähnten Fragen „Wer liest das Modell?“ und „Was soll aus dem Modell gelesen werden?“ für



ein kleines, überschaubares Beispiel zu beantworten. Bei der Erstellung dieses Beispielmodells leiten wir folgende Informationen ab:

- **Die Methode:** Die Methode legt fest, wann welcher Teil des Modells erstellt wird. Daraus leiten sich die Modellstruktur und die verwendeten Modell-Elemente ab.
- **Die Modellstruktur**: Die Modellstruktur gibt den Rahmen vor, in dem das Modell enthalten ist, dient als Orientierung bei der Navigation durch das Modell und ist die Grundlage für die Generierung von Dokumenten.

- **Die Modell-Elemente**: Wir verwenden eine oder mehrere Modellierungssprachen, aber selten alle Modell-Elemente aus diesen Sprachen. Es ist wichtig zu definieren, wann welche Modell-Elemente für welchen Zweck verwendet werden und vor allem wie diese verbunden werden dürfen. Basierend auf der Modellstruktur wird definiert, welche Modell-Elemente auf welcher Abstraktionsebene wie miteinander verbunden werden.

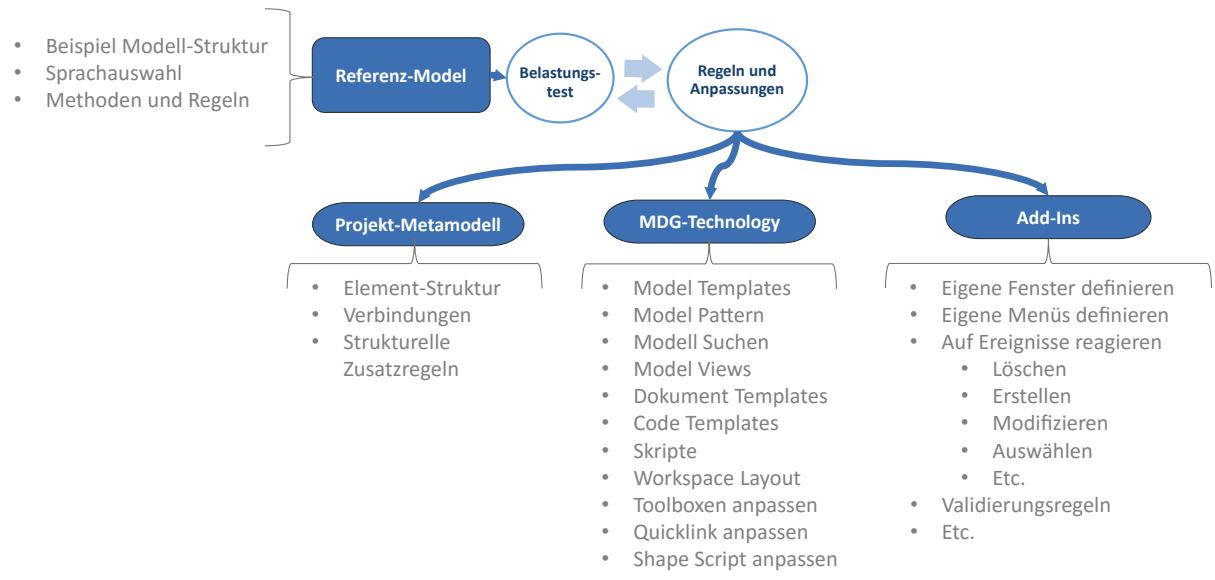
Im nächsten Schritt müssen wir das erstellte Beispielmodell analysieren und verallgemeinern. Wir leiten daraus Regeln ab, mit denen wir ähnliche Modelle wie das Beispielmodell erstellen können. Diese Regeln können wir auch wieder als Modell mit zusätzlichen Einschränkungen definieren. So ein Modell heißt Metamodell. Da es ein Metamodell für unser Projekt ist, nennen wir es „Projekt-Metamodell“.

Das Projekt-Metamodell als Regelbasis

Das Projekt-Metamodell ist also ein Modell, mit dem wir definieren, wie ein Modell aufgebaut sein darf. Jede Modellierungssprache hat ein Metamodell, in dem definiert ist, wie die Sprach-Elemente der Modellierungssprache (z. B. UML/SysML/etc.) zu verwenden sind. Das Projekt-Metamodell, das wir erstellen, ist allerdings ein Metamodell, in dem beschrieben wird, wie wir mit einer oder mehreren Modellierungssprachen unsere Probleme und Lösungen beschreiben. Genaugenommen ist es also eine Sub-Menge aus einem oder mehreren anderen Sprach-Metamodellen, ergänzt mit Regeln und Einschränkungen, wie diese verwendet werden, um ein Projekt wie das Referenzmodell zu erstellen.

Die Regeln, die durch die Modellierungssprache definiert sind, können wir in unserem Projekt-Metamodell noch weiter einschränken bzw. aufweichen. Die wenigen Restriktionen, die standardmäßig im Enterprise Architect implementiert sind, machen dies möglich.

Im Projekt-Metamodell verallgemeinern wir das



Referenzmodell und beschreiben damit die Regeln, wie ein Modell (ähnlich zum Referenzmodell) auszusehen hat. Das Projekt-Metamodell beschreibt, welche Modell-Elemente in einer Baumstruktur (dem Project Browser) verschachtelt werden und mit welchen Beziehungen diese verbunden werden dürfen.
Der Unterschied zwischen dem Projekt-Metamodell und dem Referenzmodell besteht darin, dass im Referenzmodell ein konkretes Beispiel beschrieben wurde und im Projekt-Metamodell das Schema zum Erstellen eines weiteren Beispiels. Im Referenzmodell haben wir z.B. einen SysML Block mit Namen „Kaffeemaschine“, der einen Zustandsautomaten enthält, der alle möglichen Zustände der Kaffeemaschine aus einer logischen Sicht beschreibt. Im Projekt-Metamodell können wir dann ableiten, dass jeder SysML Block durch einen oder auch mehrere Zustandsautomaten beschrieben werden kann (optional) oder beschrieben werden muss.
Haben wir diese Überlegung für alle Teile des Referenzmodells angestellt und dafür ein Projekt-Metamodell entwickelt, können wir eigentlich schon mit dem nächsten Projekt loslegen. Wir wissen, wann wir welche Informationen mit welchen Modell-Elementen abbilden und aufgrund des beispielhaften Referenzmodells, auf welche Abstraktion mit wie vielen Details modelliert werden soll.
Jetzt hängt der Erfolg nur noch davon ab, dass der Ansatz vom ganzen Team verstanden wird und sich alle beteiligten Personen immer an die im Projekt-Metamodell formulierten Regeln halten.

Die Einhaltung des Projekt-Metamodells unterstützen

Enterprise Architect bietet viele Möglichkeiten, um die Arbeit zu vereinfachen und das Team bei der Regelbefolgung zu unterstützen.
Basierend auf dem Referenzmodell und Projekt-Metamodell werden alle unnötigen Menüeinträge, die sich abschalten lassen, auch wirklich deaktiviert. Fehlende Funktionen werden ergänzt. Das Werkzeug kann nun - auf dem Projekt-Metamodell basierend - mögliche nächste Schritte vorschlagen und die Eingabe der Informationen erleichtern. Das erstellte Modell erlaubt jederzeit die Überprüfung der Übereinstimmung mit dem Projekt-Metamodell. Die Überprüfung des Modells auf Korrektheit und Vollständigkeit könnte auch schon beim Erstellen des Modells geschehen, um ein lästiges Nacharbeiten zu vermeiden.

Enterprise Architekt bietet dazu folgende Mechanismen:

- **MDG-Technologies**: Eine Model-Driven-Generation-Technology ist eine Sammlung an Konfigurationen, die einfach verteilt werden kann.
    - Vorhandene Toolboxen anpassen: Für die UML lässt sich die vorhandene Toolbox anpassen, indem unerwünschte Modell-Elemente und -Beziehungen entfernt und fehlende aufgenommen werden.
    - Eigene Toolboxen und Diagramm-Typen erstellen: Mittels UML-Profil lässt sich eine eigene Modellierungssprache mit eigenen Toolboxen und Diagrammen erstellen. Vorhandene Modellierungssprachen lassen sich anpassen und erweitern, auch die grafische Darstellung (das Shape) des Modell-Elements ist adaptierbar.
    - Eigene Quicklink-Menüs (Schnellzeichenpfeil-Menü) erstellen: Der Schnellzeichenpfeil erleichtert bei der Modellierung den Zugriff auf die gewünschten Beziehungen. Fehlende Beziehungsarten im Quicklink-Menü sind hinzufügbar.

- **Add-ins**: Ein Add-in dient der Erstellung eigener User Interfaces, um die Informationen des Modells anders aufbereitet anzuzeigen oder eine alternative Eingabemaske anzubieten. Diese UI Dialoge sind anstelle der Standard-Dialoge des EA anzeigbar. Damit lassen sich auch Wizards erstellen, die den Modellierer bei der Einhaltung komplizierter Prozessen unterstützen. Dank Add-in ist eine Reaktion auf viele Ereignisse im Enterprise Architect möglich und es lassen sich Automatismen im Hintergrund (ohne Benutzerinteraktion) durch-

führen, wenn die Oberfläche des Enterprise Architect verwendet wird.

- **Installationsspezifische Konfigurationen**: Jede Installation des Enterprise Architect bietet individuelle Konfigurationen, die sich auf alle Installationen übertragen lassen.

Konkrete Ansätze werden im Kapitel Regeln und Anpassungen besprochen. Zuvor betrachten wir noch den Schritt des Belastungstests, der uns hilft, den ausgearbeiteten Modellierungsansatz in einem erweiterten Kernteam zu testen und zu verifizieren, ob er auch praxistauglich ist.

Der Belastungstest

Der Belastungstest ist einer der wichtigsten Schritte im Einführungsprozess eines Modellierungsansatzes. Das erstellte Referenzmodell wird dabei von zukünftig beteiligten Personen auf folgende Punkte überprüft:

1. **Fragen sind einfach zu beantworten**: Alle zuvor definierten Fragestellungen sollten mit dem Modell von allen mit dem Modell arbeitenden Personen einfach beantwortet werden können.
2. **Keine weiteren Fragen tauchen auf**: Neben den zuvor definierten Fragen können die beteiligten Personen keine weiteren Fragen identifizieren, die mit dem Modell noch beantwortet werden sollten.
3. **Erstellung und Änderung ist einfach**: Die Erstellung neuer Informationen und die Veränderung vorhandener Informationen verursacht keinen großen Aufwand.

Nun betrachten wir einige Beispiele zu diesen drei Punkten, bei denen Handlungsbedarf besteht.

Zu Punkt 1. : Die Information ist zwar im Modell vorhanden, die Antwort kann allerdings nur schwer aus dem Modell gelesen werden. Die Ursache dafür liegt darin, dass der Betrachter des Modells mit der Komplexität des Werkzeuges noch nicht umgehen kann bzw. weil die Suche nach der Antwort zu aufwendig und fehleranfällig ist.
Lösung: Wir können mit Modell-Suchen, Skripten und Add-ins die Fragestellung per Knopfdruck beantworten lassen. Dazu wird die Modell-Struktur aufgrund des Projekt-Metamodells durchsucht und die Antwort „berechnet“, es sind beliebig komplexe Modellen möglich. Um die „richtige“ Information zu berechnen, ist allerdings die Einhaltung des Projekt-Metamodells unbedingte Voraussetzung.
Um den Betrachter des Modells noch besser zu unterstützen, kann bei der Suche gegebenenfalls auch gleich eine Analyse und Validierung des Modells angestoßen werden. Neben dem gewünschten Suchergebnis erhält der Betrachter die Information, ob das Ergebnis vollständig bzw. unvollständig ist (Regelverletzungen gefunden).
Wird z. B. nach allen noch nicht realisierten Kundenanforderungen gesucht, so werden die im Projekt-Metamodell definierten Verlinkungsmuster analysiert. Die Regel könnte z. B. lauten, dass alle Kundenanforderungen im Paket „Kundenforderungen“ enthalten sein müssen. Ist aber aus der täglichen Praxis bereits bekannt, dass das Referenzmodell oft nicht eingehalten wird, kann gleichzeitig das Modell überprüft werden, ob an anderen Stellen im Modell ebenfalls (Kunden-) Anforderungen gefunden wurden, welche unter Umständen im Suchergebnis enthalten hätten sein sollen.

Das ist natürlich ein Thema für ein schon sehr ausgereiftes Referenzmodell, das bereits einige Belastungstests hinter sich hat. Bedenken Sie, dass Ziel anfangs nicht zu hoch zu stecken und zuerst zu überprüfen, ob diese Automatismen auch notwendig sind und für den Anwender einen Vorteil bringen. Gibt es allerdings bei der Modellierung immer wieder Reibungsverluste, sollten diese Schritte unbedingt in Erwägung gezogen werden.

Zu Punkt 2. : Die benötigten Informationen lassen sich im Modell nicht finden, weil z. B. auf eine Fragestellung vergessen wurde.
Lösung: Wir müssen die fehlenden Fragen formulieren, die mit dem Modell beantwortet werden sollen. Anschließend werden wir diese Information in das Referenzmodell aufnehmen, um daraus eine neue Version des Projekt-Metamodells abzuleiten.

Wenn die neuen Fragestellungen den gesamten Modellierungsansatz infrage stellen, kann ein neuer Experten Workshop verhindern, dass beim Umbau des Referenzmodells eine falsche Richtung eingeschlagen wird.
Wurde das Referenzmodell angepasst, sollte jedenfalls ein Experten Review prüfen, ob die anvisierte Lösung unter den gegebenen Umständen optimal ist.
Der nächste Belastungstest wird schließlich die Praxistauglichkeit der Lösung aufzeigen.

Die Reihenfolge Referenzmodell -> Projekt-Metamodell sollte eingehalten werden, da wir im Referenzmodell schon prüfen können, ob der vorgesehene Lösungsansatz, also der Teil des Modells, mit dem die neue Frage beantwortet werden kann, auch praxistauglich ist.
Beim direkten Anpassen des Projekt-Metamodells würde dieser wichtige Schritt entfallen.
Wird ein sehr aufwendiges Modell erstellt und somit eine komplizierte Lösung anvisiert, wird die Lösung beim Experten Review analysiert und eventuell eine alternative Abbildung gefunden.
Wird keine einfache Lösung gefunden, kann die Komplexität durch Automatismen und Assistenten im Werkzeug entschärft werden.

Zu Punkt 3.: Die Suche nach Informationen ist ja nur eine Seite der Medaille. Die andere Seite besteht in der einfachen Möglichkeit der Manipulation und Erweiterung der bestehenden Modelle. Ist der Aufwand für die Erstellung und Änderung des Modells nach den Vorgaben des Referenzmodells und Projekt-Metamodells zu hoch, besteht dringender Handlungsbedarf. Wird hier nicht gleich gegengesteuert, werden die Modelle nicht nach den Vorgaben erstellt werden können und damit mit hoher Wahrscheinlichkeit nicht die gewünschten Informationen enthalten.
Ein Verstoß gegen Punkt 3 kann auch dazu führen, dass sich die Mitarbeiter nicht mehr trauen, das Modell anzugreifen, weil der regelkonforme Umgang zu kompliziert ist. Damit „stirbt“ der Modellierungsansatz (siehe „Komplex – ohne Unterstützung“).
Lösung: Die Erstellung der Modelle sollte so einfach wie möglich gehalten werden. Das kann einerseits durch einfache Modelle geschehen. Je nach gewünschter Informationsdichte im Modell ist dies allerdings nicht immer möglich. Ist die Realität kompliziert, spiegelt sich dies meist auch in der Abbildung im Modell wieder; d.h. die Modelle und deren Struktur werden ebenfalls eine gewisse Komplexität aufweisen.
Wir können nun zwei Arten der Komplexität unterscheiden: Die des Werkzeugs und die des Modellierungsansatzes (die der Realität betrachten wir ja nicht). Durch eine Anpassung der Oberfläche des Enterprise Architect lässt sich die Komplexität des Werkzeuges etwas reduzieren. Ein grundlegendes Verständnis von Modellierungswerkzeugen muss bei allen am Projekt beteiligten Personen aber trotzdem vorhanden sein. Das direkt am Modell arbeitende Team sollte zumindest „Fundamental“, besser noch „Advanced“ Status besitzen.
Die weitere Komplexität liegt nun in der Information, die abzubilden ist. Durch die Ausarbeitung des Referenzmodells und des Projekt-Metamodells bekommen wir die Möglichkeit für den Einsatz von Assistenten, die bei der Erstellung und Manipulation der Modelle unterstützen und führen. Die Assistenten und Wizards achten auf die Vollständigkeit und Konsistenz des Modells, so können fehlerhafte Modelle gar nicht erst entstehen.
Die Validierung verifiziert das Modell nach dem Projekt-Metamodell und stellt so sicher, dass das Modell nach der Validierung ein nach unserem Projekt-Metamodell korrektes Modell ist. Alle aus dem Modell generierten Artefakte sind damit zumindest nach den vorgegebenen Regeln korrekt.
Nach dem Belastungstest
Nach erfolgreicher Beendigung des Belastungstests steht uns ein Modellierungsansatz zur Verfügung, der von einem Teil der Mitarbeiter bereits akzeptiert und

verstanden wurde und damit produktive Arbeit erlaubt.

Nach dem Belastungstest

Nach erfolgreicher Beendigung des Belastungstests steht uns ein Modellierungsansatz zur Verfügung, der von einem Teil der Mitarbeiter bereits akzeptiert und verstanden wurde und damit produktive Arbeit erlaubt.

Schulung vorbereiten: Um den definierten Modellierungsansatz in die ganze Gruppe, Abteilung, Firma oder den ganzen Konzern auszurollen, sollte nun ein Schulungskonzept entwickelt werden. Meist reicht es, wenn das Referenzmodell als Beispiel herangezogen wird und die Anpassungen und Automatismen bereits ausgerollt wurden. Viele Beispiele haben mir allerdings gezeigt, dass ein Referenzmodell alleine nicht ausreichend ist. Das Projekt-Metamodell muss vorhanden sein, um alle Regeln zu kennen. Das Referenzmodell dient nur als exemplarisches Beispiel.

Schulung durchführen: In der Schulung wird zielgerichtet der bereits erstellte und verifizierte Modellierungsansatz vermittelt, ohne sich mit all den Möglichkeiten und Eventualitäten herumzuschlagen, die eine Modellierungssprache, ein Werkzeug und Methoden mit sich bringen.

Der Einstieg in die produktive Arbeit wird dadurch erheblich erleichtert, da die Mitarbeiter nur das lernen, was sie anschließend auch wirklich brauchen. Die Anpassungen stellen darüber hinaus sicher, dass sie nichts falsch machen können.

Regeln und Anpassungen

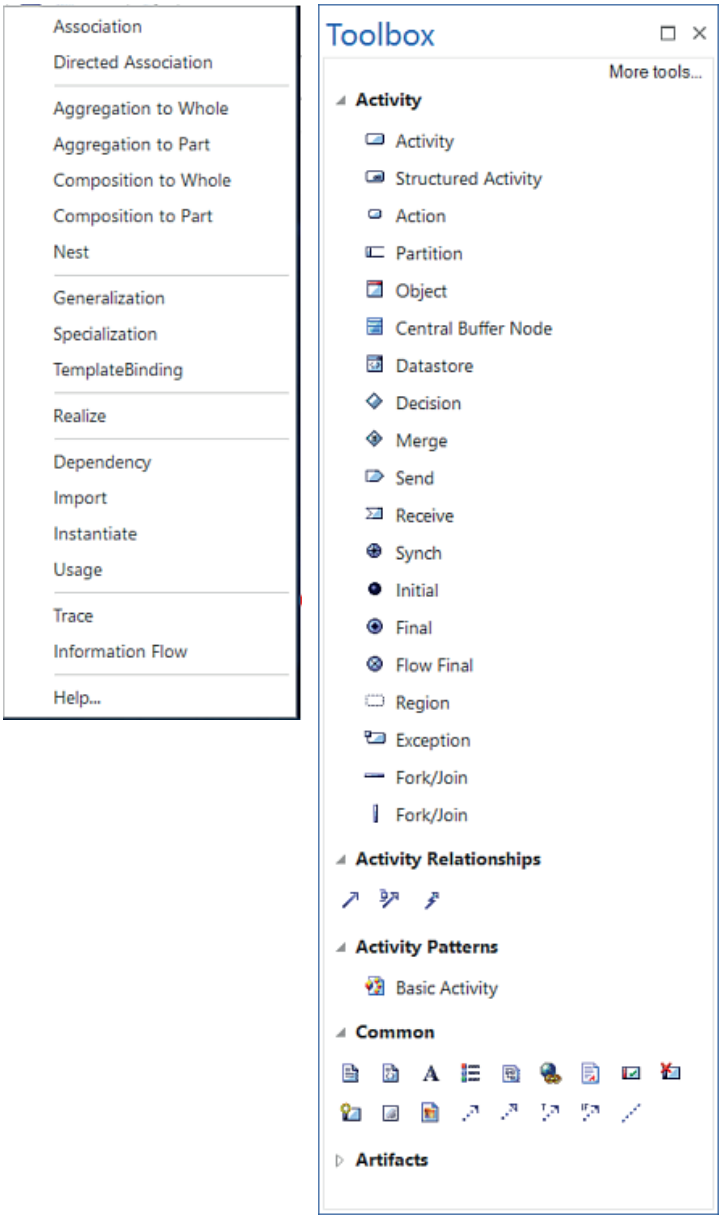
Durch den Belastungstest haben wir die Ecken und Kanten unseres aktuellen Modellierungsansatzes herausgefunden. Wir wissen nun, was gut funktioniert und wo wir noch nachbessern können.

Einer der ersten Punkte ist die Bereinigung der Toolboxen und des Quicklinkers (Schnellzeichenpfeil), um alle unnötigen Modell-Elemente und Beziehungen auszublenden.

Danach folgt eine automatische Modellvalidierung, die das Projekt Metamodell verifiziert. Damit erhalten die Mitarbeiter die Sicherheit, dass das Modell nach getaner Arbeit auch wieder korrekt ist.

Werden bei der Validierung des Modells immer wieder dieselben Fehler gefunden, lässt sich die Validierung auch direkt beim Erstellen der betreffenden Modellstruktur durchführen (on the fly). Dabei wird zusätzlich zur Fehlermeldung ein Vorschlag zur Richtigstellung gegeben. Wenn die Regel es erlaubt, kann die Richtigstellung des Modells auch automatisch geschehen. Der Fantasie sind dabei wenig Grenzen gesetzt.

Wenn zum Beispiel die Richtung einer Beziehung zwischen zwei Modell-Elementen immer wieder falsch ist, lässt sie sich automatisch richtigstellen. Wurde etwa eine Klasse aus einer falschen Abstraktionsebene als Typ eines Attributes auf der Implementierungsebene verwendet, kann dies beim Zuweisen überprüft und verhindert werden.



Durch das Validieren on-the-fly wird die lästige Nacharbeitung fehlerhafter Modelle vermieden. Auch ist der Lerneffekt meist größer, wenn der Fehler sofort bei der Erstellung des Modells aufgezeigt wird.
Im folgenden Abschnitt sehen wir uns die wichtigsten Anpassungsmöglichkeiten an und diskutieren Einsatzszenarien.

Mit MDG-Technologien Konfigurationen verteilen

Enterprise Architect bietet die Möglichkeit, Konfigurationen und Anpassungen der Modellierungssprache als MDG-Technology anzulegen. Der Begriff MDG steht für **M**odel-**D**riven-**G**eneration, hat aber nur bedingt etwas mit den Abkürzungen MDA, MDD, MDE, etc. zu tun. Im Kontext des Enterprise Architect bedeutet MDG immer Erweiterung oder Anpassung.
In einer MDG-Technologie können wir also viele individuelle Konfigurationen zusammenfassen und einfach verteilen. Die MDG-Technologie kann dabei lokal auf Ihrem Rechner vorhanden sein oder beim Start des Enterprise Architect von einem Server geladen werden. Ab Enterprise Architect 13 besteht die Möglichkeit, die in einem Projekt notwendigen MDG-Technologien automatisch zu aktivieren und zu deaktivieren. Damit lässt sich sicherstellen, dass nur die für das Projekt notwendigen MDG-Technologien zum Einsatz kommen.
Im folgenden Abschnitt wird erläutert, was in eine MDG-Technology aufgenommen werden kann, um ein neues Modell (vergleichbar zum Referenzmodell) einfacher zu erstellen.

Model Templates

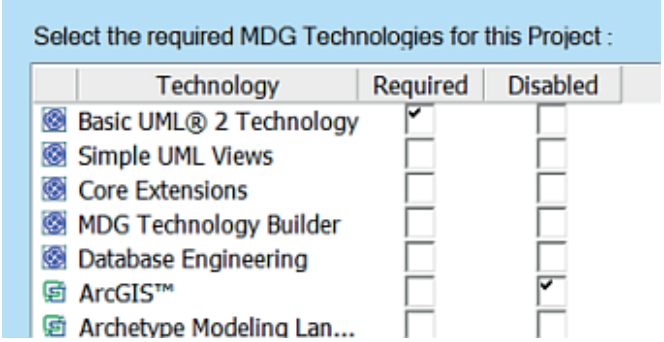
Das Referenzmodell gibt eine bestimmte Modell-Struktur vor, die aber nicht unbedingt für alle weiteren Projekte anwendbar sein muss. Wie wir schon gesehen haben, entsteht aus der Analyse und Verallgemeinerung des Rerenzmodells das Projekt-Metamodell.
Teilstrukturen, die in allen weiteren Projekten wiederverwendet werden sollen, lassen sich als Modell-Templates abspeichern und in eine MDG-Technologie aufnehmen. Ein Modell-Template wird anschließend mit dem Model-Wizard an beliebiger Stelle im Project Browser eingefügt.

Das Template kann neben der Paket-Struktur auch Diagramme mit enthaltenen Modell-Elementen und damit bereits ein kleines Beispiel mit einer Beschreibung umfassen. Dies hilft beim Verständnis der Struktur und deren Befüllung.

Aus dem Referenzmodell wird so die grundlegende Modell-Struktur definiert, also die Root-Pakete sowie die erste und vielleicht zweite Ebene der Pakete im Project-Browser. Die weitere Struktur des Project-Browser wird ebenfalls mittels Model-Templates aufgebaut.
Haben wir z. B. ein Muster entwickelt, wie Systemkomponenten zu strukturieren sind, wird diese Teil-Struktur als Model-Template abgelegt. Brauchen wir eine neue Komponente, lässt sich dann die vorgesehene Teil-Struktur mit einem Klick an beliebiger Stelle erzeugen.

Modelle bei der Einfügung anpassen: Model-Templates sind auch mittels Add-ins einfügbar. Dabei besteht die Möglichkeit, weitere Informationen vom Enterprise Architect-Benutzer oder anderen Quellen abzufragen und das Model-Template bei der Einfügung noch anzupassen (oft werden so die Namen von Modell-Elementen geändert).

Komplexität durch Deaktivierung von MDG-Technologien reduzieren: Mehr ist nicht immer besser. Je mehr MDG-Technologien aktiviert sind, desto mehr Möglichkeiten hat der Enterprise Architect-Benutzer. Nicht benötigte Technologien sollte man also besser deaktivieren. Enterprise Architect 13 ermöglicht die Festlegung, welche MDG-Technologien bei Öffnung des Projektes aktiviert bzw. deaktiviert werden sollen.



Model Pattern

Model Pattern kommen zum Einsatz, wenn wir in unserem Referenzmodell wiederkehrende Muster erkannt haben. Diese Muster werden als Pattern abgespeichert und in eine MDG-Technologie aufgenommen.
Der EA bietet auch eine Menge vordefinierter Patterns - die bekannten „GoF Design Patterns“ - in einer eigenen MDG-Technologie an.
Ein Model Pattern ist ähnlich zu einem Model Template. Während bei einem Model Template eine komplette Modell-Struktur (mit Paketen, Diagrammen und Diagramm-Inhalten) definierbar ist, werden bei einem Model-Pattern hingegen lediglich Modell-Elemente und Beziehungen eingefügt.
Bei der Einfügung lässt sich je nach Konfiguration festlegen, ob ein im Pattern enthaltenes Modell-Element einfach neu erzeugt, mit einem vorhandenen Modell-Element verschmolzen oder ob eine Instanz davon erzeugt werden soll.

Modell-Suchen

Mit Modell-Suchen lassen sich bestimmte Fragen, die wir mit dem Modell beantworten möchten, einfach per Knopfdruck erledigen. Wir können z. B. nach Modell-Elementen suchen, bei denen ich als verantwortliche Person zugeordnet bin oder an denen ein Change-Element hängt, das mir zugeordnet wurde.
Das Modell-Repositorium des Enterprise Architect ist eine Datenbank. Das Datenbank-Schema ist relativ generisch, um beliebige Arten von Modellen und Modellierungssprachen abspeichern zu können. Der Kern des Schemas ist überschaubar und kann für die Suche nach bestimmten Mustern im Modell zum Einsatz kommen. SQL Kenntnisse sind dabei von Vorteil. Der Kern des Schemas ist auf unserem Blog zu finden. Enterprise Architect erlaubt aber auch eine Suche beispielhaft zusammen zu klicken, wie aus Microsoft Access bekannt. Wir unterscheiden folgende Suchfunktionen:

- **Query Builder Suche**: Die Suche wird einfach zusammengeklickt, indem Eigenschaften von Modell-Elementen ausgewählt und mit den gesuchten Werten befüllt werden. Damit lässt sich allerdings nur nach Objekten mit bestimmten Eigenschaften suchen. Für die Suche nach Elementen und Beziehungen ist eine QL Suche notwendig.

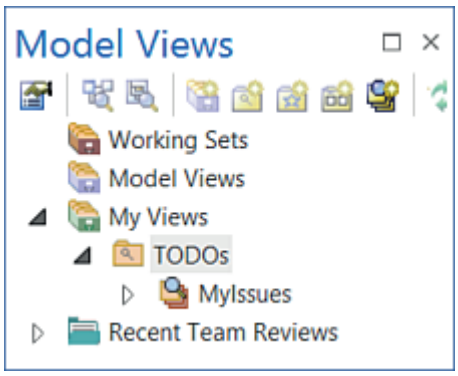
- **SQL Suche**: Die SQL Suche bietet vollständigen Zugriff auf alle Daten in der Datenbank und erlaubt die Suche nach beliebigen Mustern im Modell, allerdings nicht rekursiv Möchten wir also von einer Anforderung ausgehend alle Modell-Elemente auflisten, die mit einer Realisierungs-Beziehung verknüpft sind, kennen wir in der Regel die Anzahl der mit einer Realisierungs-Beziehung verbundenen Modell-Elemente nicht. Mit einer einfachen SQL Abfrage könnten wir zwar auch danach suchen, müssen aber immer von einer festen Anzahl an verbundenen Modell-Elementen ausgehen. Rekursive Suchfunktionen lassen sich mit einem Add-in realisieren.

- **Add-in Suche**: Die Add-in Suche erlaubt mit dem Enterprise Architect Objektmodell den vollen Zugriff und die Anwendung beliebiger Bedingungen. Wir können sogar externe Bibliotheken einbinden und nicht nur im Enterprise Architect, sondern auch in anderen Systemen suchen.

Model Views

Model View erlaubt die Anwendung zuvor definierter Suchfunktionen sowie die Anzeige der Ergebnisse in einem eigenen Fenster (ähnlich zum Project Browser). Die Suche lässt sich auch periodisch starten, um eine automatische Benachrichtigung zu erhalten.
Damit bilden wir eine Art „Soft-Prozess“ ab: Für die Fragen an unser Modell, die wir mit einer Modell-Suche beantworten können, wird das Ergebnis automatisch im Model View angezeigt.
Ich verwende in meinen Projekten z. B. eine Suchfunktion „TODOs“, mit der ich Aufgaben für mich anzeige. Wurde die Aufgabe erledigt und die Eigenschaft, die als Suchkriterium definiert ist, am Modell-Element

geändert, verschwindet das Modell-Element automatisch aus der Ergebnisliste im Model View.



Dokument Templates

Einer der großen Vorteile eines Modell-Repositoriums ist es, die Inhalte des Repositoriums als Dokumente generieren zu können. Enterprise Architect bietet viele Möglichkeiten, um auch sehr komplexe Dokumente aus dem Modell zu generieren.

Die Basis zur Generierung der Dokumente ist das Dokument Templates, in denen beschrieben wird, an welcher Stelle im Dokument welche Information aus dem Modell in welcher Formatierung ausgegeben werden soll.

Definieren wir mehrere unterschiedliche Dokument Templates, lassen sich aus einem Modell mehrere unterschiedliche Dokumente generieren. Die Templates steuern, welche Informationen ausgegeben werden sollen.

Die Dokument Templates setzen in der Regel eine gewisse Projekt Struktur voraus, welche in unserem Projekt-Metamodell beschrieben wurde. Wird das Projekt-Metamodell nicht eingehalten - und die Information mit anderen Modell-Elementen und –Sichten als die vorgeschriebenen abgebildet oder an anderer Stelle in der Modell-Struktur abgelegt - wird die Information im generierten Dokument nicht oder an falscher Stelle enthalten sein. Werden z. B. die Kundenanforderungen nicht im Paket „Kundenanforderungen“ eingeschachtelt, tauchen diese im Anforderungsdokument nicht auf, wenn aus dem Paket „Kundenanforderungen“ ein Dokument generiert wird. Die Anforderungen finden sich unter Umständen in einem anderen Dokument, weil die Anforderungen im falschen Paket eingeordnet waren.

Modell-Validierung verhindert solche Probleme, da sie die mangelhafte Einhaltung des Projekt-Metamodells erkennt.

Dokument Templates könnten auch so intelligent erstellt werden, dass sie alle Eventualitäten im Modell abfangen und die Information wie gewünscht in das Dokument kommt, egal wie die Projekt-Struktur aussieht. Damit erhalten wir zwar aus ungeordneten Modellen geordnete Dokumente, die Arbeit im Modell wird aber nicht erleichtert. Die Fehleranfälligkeit wächst, weshalb die Arbeit nach einem Projekt-Metamodell unbedingt empfohlen wird.

Der Aufwand für die Erstellung solcher „intelligenten“ Dokument Templates ist ebenfalls hoch. Einfacher ist die Abbildung der Überprüfung als explizite Modellierungsregeln, die vor der Generierung der Dokumente das Modell überprüfen.

Neben dem im Enterprise Architect enthaltenen Dokument-Generator existieren auch Lösungen anderer Hersteller für die Erstellung von Dokumenten (siehe dazu „3rd Party Produkte“ auf www.sparxsystems.de).

Code Templates

Je nach Modellierungsansatz lässt sich aus Implementierungsmodellen Code generieren. Enterprise Architect bietet dafür anpassbare Code-Generierungs-Templates. Auch dazu existieren Lösungen von anderen Herstellern zur Codegenerierung aus Modellen (siehe dazu „3rd Party Produkte“ auf www.sparxsystems.de).

Unabhängig vom eingesetzten Code-Generator wird immer eine gewisse Modell-Struktur für die Generierung ausführbaren Codes vorausgesetzt.

Mittels Modell-Validierung lässt sich die Korrektheit und Vollständigkeit von Implementierungsmodellen schon im Enterprise Architect überprüfen. Je nach verwendetem Code-Generator sind eventuell Modell-Validierungen auch im dort bereits enthalten.

Skripte

Referenzmodell und Projekt-Metamodell definieren, welche Information mit welchen Modell-Elementen abgebildet werden dürfen. Dabei kann die Erstellung eines Modells (basierend auf dem Projekt-Metamodell)

etwas aufwendig sein, weil mehrere Klicks notwendig sind bzw. immer wieder dieselben Schritte durchzuführen sind. Skripte erlauben die Automatisierung dieser lästigen Aufgabe.
Der Vorteil von Skripten ist, dass sie im Modell-Repositorium erstellt und dort gespeichert werden. Damit stehen sie jedem ohne zusätzliche Installation einer Erweiterung (Add-in)zur Verfügung, der das Enterprise Architect Projekt öffnet. Enterprise Architect Security verhindert dabei, dass Skripte von Personen erstellt, verändert oder ausgeführt werden, die nicht über die entsprechende Berechtigung verfügen.
Der Nachteil von Skripten ist es, dass sie nicht auf Ereignisse im Enterprise Architect reagieren und keine umfangreiche Benutzeroberfläche erstellt werden kann. Aktuell gibt es eine 3rd Party Erweiterung (Add-in). mit deren Hilfe ein Skript auch auf Ereignisse im Enterprise Architect reagieren kann.
Skripte lassen sich aus verschiedenen Kontexten aufrufen und es ist möglich, über das Kontextmenü im Projekt Browser und Diagramm ein bestimmtes Skript auszuführen.

Workspace Layouts

Enterprise Architect bietet viele Fenster, die jeweils unterschiedliche Informationen aus dem Modell-Repositorium anzeigen. Je nach Arbeitsweise und Rolle benötigt ein Nutzer unterschiedliche Fensterkonfigurationen. Ein Workspace Layout erlaubt dafür die Abspeicherung benannter Fensterkonfigurationen. Diese Workspace Layouts werden mittels einer MDG-Technologie allen Nutzern zur Verfügung gestellt.

Toolboxen und Quicklink anpassen

Die im Enterprise Architect vorhandenen Toolboxen und Quicklink-Kontextmenüs sind ein Vorschlag für die in einem Diagramm zu verwendenden Modell-Elemente und die möglichen Beziehungen zwischen diesen.
In einem Diagramm können allerdings beliebige Modell-Elemente Anwendung finden und mit Beziehungen verbunden werden, die im Quicklink-Menü nicht enthalten sind. Diese Beziehungen werden aus unterschiedlichen Toolboxen ausgewählt.
Zur Erleichterung der Arbeit sowie zur Vermeidung der Suche in der Toolbox lässt sich der Inhalt der vorhandenen Diagramm-Toolboxen und des Quicklink-Menüs erweitern.
Diese Anpassung reduziert die Komplexität des Enterprise Architect enorm, da seine Bedienung bereits an die durch das Projekt-Metamodell definierte und erlaubte Modell-Struktur angepasst ist und somit das Erstellen von „falschen“ Modellen erheblich erschwert. Wird dennoch eine falsche Beziehung aus den erlaubten Beziehungen ausgewählt, wird dieser Fehler bei einer Modell-Validierung gefunden, die die gesamte Modell-Struktur berücksichtigt und nicht nur die beiden Modell-Elemente, die mit der Beziehung verbunden werden.

UML-Profile und ShapeScripts

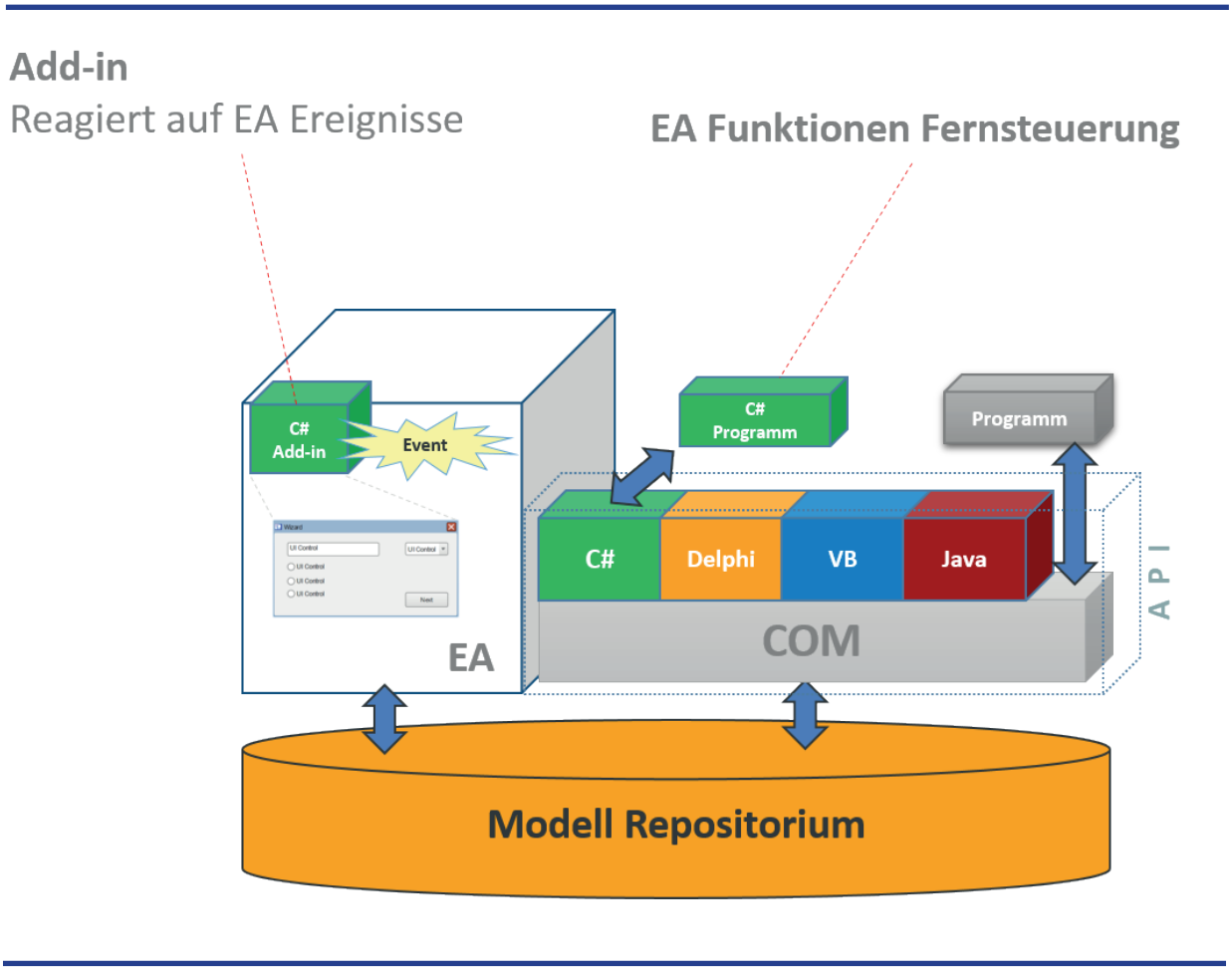
Wenn die Sprachelementen der gewählten Modellierungssprache nicht ausreichen lassen sich speziellere Symbole mit zusätzlichen Eigenschaften als UML-Profil erstellen.
In einem UML-Profil werden Stereotypen mit Tagged Values verbunden, der Stereotyp definiert dabei für das Modell-Element eine neue Bedeutung (Semantik). Wird ein Stereotyp mit einem zugeordnetem Tagged Value auf ein Modell-Element angewendet, erhält dieses automatisch die im UML-Profil definierten Tagged Values. Enterprise Architect erlaubt es, Modell-Elementen mit einem Stereotyp eine alternative grafische Darstellung zu verleihen. Anstelle des von der verwendeten Modellierungssprache vorgesehenen Symbols wird dabei ein alternatives Symbol (Shape) für den Stereotyp definiert.
Dieses Shape kann sein Aussehen auch dynamisch ändern, wenn sich Eigenschaften des Modell-Elements ändern. Damit lassen sich einfach domänenspezifische Sprachen erstellen.
Um die Benützung noch weiter zu vereinfachen, wird für diese stereotypisierten Modell-Elemente ein eigenes Eigenschaftenfenster (Properties-Fenster) erstellt. Damit ist es nicht mehr notwendig, sich im generischen Eigenschaftenfenster des Modell-Elements zurechtzufinden und die relevanten Informationen an unterschiedlichen Stellen zu suchen und dort zu modifizieren. Der Stereotyp und die Tagged Values können

dann z. B. unter anderen Namen und an anderer Stelle wesentlich deutlicher angezeigt werden.
Um eigene Benutzerfenster zu definieren und Funktionen dafür zu erstellen, können wir die Erweiterungsschnittstelle von Enterprise Architect verwenden und Add-ins entwickeln.

Mit Add-ins die Arbeit erleichtern

Ein Add-in oder Plug-in ist eine benutzerspezifisch programmierte Erweiterung, die in den Enterprise Architect geladen wird und auf Ereignisse reagieren kann. Neben einem Add-in lassen sich auch „Automatisierungen“ erstellen. Dabei wird dieselbe Schnittstelle verwendet, um das Modell zu manipulieren und auf Funktionen des Enterprise Architect zuzugreifen. Dieses Programm wird allerdings nicht vom Enterprise Architect selbst geladen und kann dadurch nicht auf Ereignisse reagieren.

Das erstellte Programm lädt vielmehr selbst ein Modell-Repositorium, um die Daten darin zu manipulieren und einige Funktionen auszuführen, wie z. B. die Generierung eines Dokuments oder den Export/Import von XMI Files.



Eigene Fenster

Enterprise Architect verfügt standardmäßig über eine große Anzahl an Fenstern. Hat man allerdings seine eigene Arbeitsweise mittels Referenzmodell und Projekt-Metamodell ausgearbeitet und eventuell zusätzliche Sprachelemente mit Stereotypen und Tagged Values definiert, erfordert die Erstellung der Modelle bzw. das Suchen aller Informationen schon gute Grundkenntnisse.

Erscheint die Umsetzung des ausgearbeiteten Modellierungsansatzes für die Modellierer zu aufwendig, weil die einzutragenden Informationen in zu vielen unterschiedlichen Fenster angezeigt werden, lassen sich eigene Fenster definieren, um die notwendigen Informationen intuitiv eintragen und finden zu können.

Auf Ereignisse reagieren

Ein weiterer Eckpfeiler für Anpassungen sind Ereignisse (Events), die vom Enterprise Architect erzeugt werden und auf die ein Add-in reagieren muss. Ereignisse werden oft erzeugt, bevor oder nachdem etwas geschehen ist. Je nach Aufgabe kann dadurch etwas verhindert werden, was aufgrund des Projekt-Metamodells nicht erlaubt ist bzw. lässt sich das Modell automatisch manipulieren, nachdem etwas geschehen ist.

Mögliche Anwendungsfälle

Um einen Eindruck der vielfältigen Anpassungsmöglichkeiten zu geben, möchte ich nun einige konkrete Beispiele skizzieren.

Validierung basierend auf dem Projekt-Metamodell

Enterprise Architect erlaubt es, das Modell automatisch zu validieren, um Verstöße gegen das zuvor definierte Projekt-Metamodell aufzuzeigen. Voraussetzung für diese Automatismen ist ein Set an Regeln, die im Projekt-Metamodell enthalten und beschrieben sind.

Aufgrund der offenen Schnittstellen des Enterprise Architect ist es auch möglich, andere Systeme in die Validierung einzubeziehen, um so etwa den Inhalt einer gesamten Werkzeugkette zu verifizieren.

Eigene Property Fenster

Das Eigenschaften-Fenster, das mittels Doppelklick auf ein Modell-Element geöffnet wird, kann z. B. durch ein eigenes Fenster ersetzt werden. Neben den Eigenschaften des Modell-Elements lassen sich auch Informationen anzeigen und modifizieren, die aufgrund der Modell-Struktur berechnet wurden und nicht direkt zum Modell-Element gehören.

Es ist daneben möglich, Informationen aus anderen Datenquellen abzufragen und im Fenster anzuzeigen, um so andere Informationsquellen nahtlos zu integrieren. Werden z. B. die Anforderungen, die im Enterprise Architect angezeigt werden, durch einen Import aus einem anderen Werkzeug eingefügt, lässt sich per Doppelklick auf das Anforderungselement das Eigenschaftenfenster des anderen Werkzeugs öffnen. Ist dies nicht möglich, kann zumindest die aktuelle Information zum Anforderungselement aus dem anderen Werkzeug abgefragt und im eigenen Fenster angezeigt werden. Änderungen lassen sich sowohl im Enterprise Architect als auch im Werkzeug, aus dem die Anforderung ursprünglich importiert wurde, aktualisieren. Dies ist natürlich nur eines von vielen möglichen Szenarien.

Eigene Wizards

Die Möglichkeit der Definition eigener Fenster erlaubt auch die Erstellung von Wizards, die dem Modellierer bei der Erstellung und Verwaltung komplizierter Modelle helfen. In der Regel lassen sich alle Informationen mit den standardmäßig gelieferten Funktionen bearbeiten. Wurde allerdings eine spezielle Arbeitsweise etabliert, welche die erforderten Modellinformationen an unterschiedlichen Stellen ablegt, ist ein Wizard hilfreich, um die schnelle Bearbeitung zu erleichtern, ohne dabei den Überblick zu verlieren.

Automatische Synchronisation

Zum Teil werden im gewählten Modellierungsansatz Strukturen definiert, die denselben Namen erhalten sollen. Zum Beispiel wird ein Paket definiert, das logisch eine Komponente beschreibt. In diesem Paket liegt noch ein konkretes Modell-Element vom Typ „Component“ und dazugehörige weitere Informationen wie

z. B. von der Komponente angebotene Schnittstellen und verschickte Signale.
Der Name des Paketes und der Name der darin enthaltenen Komponente sollen nun immer identisch sein. Eine Modell-Validierung überprüft die geforderte Struktur und bietet bei inkonsistenten Namen auch einen Lösungsvorschlag, z. B. die automatische Umbenennung des Paketes oder der Komponente.
Wird nun an einer Stelle das Paket oder die Komponente umbenannt, besteht auch die Möglichkeit, den jeweiligen anderen Teil automatisch anzupassen. Diese automatische Synchronisation stellt sicher, dass das Modell immer aktuell ist und vermeidet so die nachträgliche Validierung und Nachbearbeitung.

Integration mit anderen Werkzeugen

Ein Modellierungswerkzeug wie Enterprise Architect muss nicht zwingend der „Nabel der Welt“ sein. Je nach Setup besteht oft die Notwendigkeit, Informationen in ein anderes Werkzeug zu übertragen (oder umgekehrt). Um den Werkzeugbruch so schmerzlos wie nur möglich zu gestalten, sollte die Werkzeugkette ausgereift sein und die nahtlose Überführung der Informationen gewährleisten.
Mit den offenen Schnittstellen existieren viele Möglichkeiten der Integration des Enterprise Architect mit anderen Werkzeugen. Dabei kann der Standard XMI als Austauschformat verwendet oder die OSLC Schnittstelle des Enterprise Architect angesprochen werden. Mit letzterer ist zwar derzeit nicht der Zugriff auf alle Informationen möglich, aber sie bietet jedenfalls einen standardisierten Weg. Will man Informationen verfügbar haben, bietet sich die Enterprise Architect API an, um Daten zu lesen und zu schreiben. Ein direkter Datenbank-Zugriff für einen schnellen Lesevorgang ist auch möglich. Der direkte Schreibvorgang in die Datenbank ist zwar auch möglich, kann allerdings zu inkonsistenten Daten führen und sollte vermieden werden (Sparx Systems übernimmt dafür keine Gewähr).

Zeitgesteuerte Generierung der Dokumente oder Baselines

Neben einem Add-in, das im Enterprise Architect selbst als Erweiterung läuft, kann der EA auch ferngesteuert werden. Dabei verwenden wir dieselben Schnittstellen wie bei einem Add-in. Der Unterschied liegt darin, wer das Programm startet und mit Ereignissen füttert.
Bei einem Add-in ist es Enterprise Architect selbst, der unser Programm startet und über Menüs oder Ereignisse unsere eigenen Funktionen aufruft. Wir können damit auch die Oberfläche anpassen und erweitern.
Eine einfache Fernsteuerung erlaubt ebenfalls die Manipulation des EA Modells, man erhält aber keine Ereignisse, da die Erweiterung nicht in der Applikation Enterprise Architect läuft.
Vor allem bei zeitaufwendigeren Aufgaben, wie der Generierung textueller Dokumente oder der Erstellung von Baselines für zuvor definierte Pakete, kann die Fernsteuerung Vorteile bringen. Sie bleibt dem Nutzer verborgen und lässt sich auch auf einem anderen Rechner durchführen. Die Ergebnisse sind natürlich wieder sichtbar und lassen sich auch direkt über den Enterprise Architect verwenden. Zum Beispiel kann über Fernsteuerung in einem Diagramm ein Hyperlink zu einem generierten Dokument erstellt werden.

Fazit

Zusammenfassend lässt sich sagen, dass der Wechsel vom textbasierten zum modellbasierten Ansatz mit einem gewissen (überschaubaren) Aufwand verbunden ist. Die wichtigste Forderung bei der Einführung von modellbasierten Ansätzen ist das Vorhandensein konkreter und realistischer Zielvorgaben. Nur so lässt sich rasch der Nutzen von Modellen zeigen und eine Verifikation des Ansatzes erstellen. Dabei sind konkrete Fragestellungen zu definieren, die mit dem Modell beantwortet werden sollen. Keine konkreten Zielvorgaben oder zu hoch gesteckte Ziele sind kontraproduktiv und führen in der Regel dazu, dass der Ansatz vom Team nicht mitgetragen wird und früher oder später „stirbt“. Wurde hingegen darauf geachtet, dass nur die wirklich notwendigen Informationen in der Form im Modell vorhanden sind, wie sie auch von anderen Beteiligten erwartet werden, kann der Nutzen eines Modells und vor allem eines Modell-Repositoriums rasch lukriert werden.

Die Mitarbeiter müssen für das aktuell anvisierte Ziel vorbereitet und geschult werden. Eine Expertenschulung auf hohem Niveau für alle Mitarbeiter führt aus verschiedenen Gründen meist nicht zum gewünschten Erfolg. Zielführender ist es, eine Schulung mit konkreten Vorgaben anhand eines Referenzmodells und eines Projekt-Metamodells zu erstellen. Dadurch wird nicht nur eine bessere Akzeptanz im gesamten Team erreicht. Die Mitarbeiter können durch die bereits gemachten Erfahrungen auch neue Ziele präziser definieren und somit schneller auf geänderte Anforderungen reagieren.

Mit dem einfachen Bild (Diagramm) eines Modells entsteht bereits ein geringfügiger Vorteil zu textbasierten Ansätzen. Die Mächtigkeit modellbasierter Ansätze steckt allerdings primär in der formalen Struktur des Modells. Unterschiedliche Automatismen können auf diese formalen Daten aufsetzen und weitere Artefakte generieren.

Um die Erstellung und Manipulation der Modelle möglichst einfach zu gestalten, sollte die Belastbarkeit des Modellierungsansatzes überprüft werden. Mögliche Fehlerquellen lassen sich durch Assistenten vermeiden, die das Modell automatisch verifizieren und den Nutzer bei der Erstellung unterstützen und führen. Dadurch wird sichergestellt, dass alle Modellierungsregeln befolgt und das Projekt-Metamodell eingehalten wird. Nur so lassen sich alle Automatismen, die ein Modell so wertvoll machen, auch wirklich nützen.