systemum
Software Management Consulting

UML ist nicht alles

Modellierungserfahrungen mit FMC (Fundamental Modeling Concepts)

Dr. Jörg-Volker Müller, Systemum GmbH & Co. KG

Die UML ist seit vielen Jahren ein De-Facto-Standard in der Modellierung von Software. In der Praxis werden jedoch neben der UML häufig andere grafische Notationen und Darstellungen verwendet, die je nach Einsatzzweck Vorteile bieten und daher im Werkzeugkasten von Requirements Engineers und Architekten nicht fehlen dürfen. Von diesen Notationen hat sich FMC (Fundamental Modeling Concepts) und speziell das darin definierte Blockdiagramm als wertvolle Notation für die Darstellung und Dokumentation von Architekturen in zahlreichen Projekten bewährt.

Der Begriff „Softwarearchitektur“ ist in der Literatur mit vielen unterschiedlichen Definitionen belegt worden. Eine einheitliche und anerkannte Definition fehlt bisher. Dies ist auch der Grund, warum der Begriff Architektur in so unterschiedlichen Facetten verwendet wird.

Ebenen und Sichten

Tatsächlich gibt es nicht die *eine* Architektur in einem System, sondern die Architektur stellt zunächst eine Hierarchie von Strukturen auf mehreren Ebenen dar. Die untere Ebene einer Architektur sind die programmiersprachlichen Elemente, also Klassen, Funktionen, Anweisungen usw.

Die Trennung von Architekturebenen und die Zuordnung der Elemente zu den entsprechenden Ebenen ist eine zentrale Aufgabe des Softwarearchitekten und sorgt dafür, dass mit den Stakeholdern jeweils auf der richtigen Abstraktionsebene kommuniziert wird.

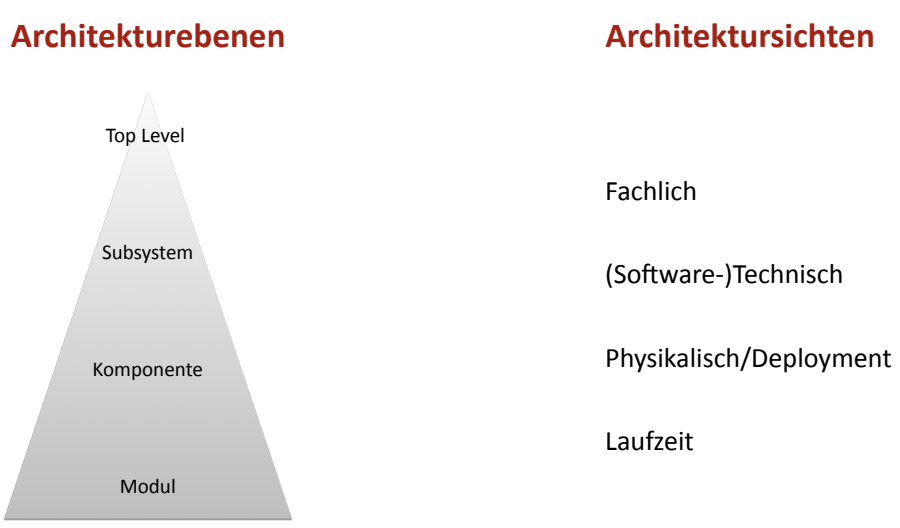

Abbildung 1: Architekturebenen und -sichten

Gleichzeitig werden die Elemente und Strukturen der Software (also die Architektur) stets aus verschiedenen Blickwinkeln (Sichten) betrachtet. Je nach Zielgruppe werden so die jeweils relevanten Sachverhalte dargestellt und fokussiert. Vor allem die Trennung von

- Fachlicher Sicht – Aus welchen Komponenten der Anwendungsdomäne besteht das System?
- Technischer Sicht – Welche technischen Mechanismen liegen der Architektur zugrunde (Middleware, Load Balancer, Eventsteuerung, ...)?
- Physikalischer Sicht – Auf welchen physikalischen Knoten im (Hardware-)Netzwerk laufen welche Komponenten?
- Laufzeitsicht – Wie erfolgt der Daten- und Kontrollfluss durch das System? Wie ist der Ablauf im Detail?

hat sich in der Praxis als sinnvoll erwiesen.

Architektur ist Kommunikation

Softwarearchitektur wird typischerweise mittels grafischer Modelle dokumentiert. Dabei stellt sich die Frage: Wozu dokumentiert und modelliert man überhaupt Architektur?

Die Architektur als zentrales Element des Softwaresystems bildet die Basis für die folgenden Entwicklungsarbeiten. Die Architektur gibt vor, welche fachlichen und technischen Anforderungen in welchen Architekturelementen realisiert werden, wie diese Architekturelemente interagieren und auf welche Weise die Qualitätsanforderungen Berücksichtigung finden. Die Architektur bildet also die Grundlage beim Übergang von der Analyse zur Realisierung.

Im Rahmen der Weiterentwicklung macht die Architekturdokumentation die Strukturen des Systems nachvollziehbar und bewertbar. Es wird möglich, die implementierte Software mit der dokumentierten Architektur zu vergleichen und somit Architekturdokumente und Code konsistent zu halten. Die Architekturdokumentation dient also auch der Aufrechterhaltung der Qualität und der Weiterentwicklung des Systems.

Beim Entwurf von Softwarearchitekturen gilt es stets, zwischen widersprüchlichen oder konkurrierenden Forderungen abzuwägen und zu entscheiden. Aufgrund dieser komplexen Entscheidungsprozesse ist der Architekturentwurf immer eine Ingenieurleistung, die nicht automatisierbar ist.

Die Zusammenarbeit zwischen Architekten, Entwicklern, Testern und anderen Stakeholdern erfordert ein gemeinsames Verständnis des Systems. Hier kann die dokumentierte Architektur den zentralen Beitrag leisten, sie bildet somit die Basis für die Kommunikation im Team.

Architekturdokumentation und UML

Wer die einschlägige Literatur und die werblichen Darstellungen der Werkzeuganbieter verfolgt, bekommt den Eindruck, dass für die Architekturdokumentation allein die UML geeignet ist, weil es zahlreiche UML-Modellierungswerkzeuge gibt, mit denen man aus der dokumentierten Architektur per Knopfdruck fertige Systeme generieren kann.

Tatsächlich muss bei dieser modellbasierten Entwicklung der Detaillierungsgrad zwangsläufig derart hoch sein, dass die Codegenerierung aus den Modellen möglich wird. Oft wird daher, wie auch in der klassischen textuellen Programmierung, schnell mit der Detailarbeit begonnen, die Darstellung der oberen Architekturebenen wird vernachlässigt.

Es bleibt eine zentrale Forderung, dass die oberen Architekturebenen durch übersichtliche und leicht verständliche Grafiken dargestellt werden. Dies leistet die UML durch Ihren Fokus auf die Details nur bedingt.

Zudem wirkt die UML für viele Stakeholder „zu technisch“. Daher entstehen in der Praxis immer wieder optisch ansprechend gestaltete, aber inhaltlich unpräzise PowerPoint-Grafiken, die für eine Kommunikation mit den technischen Stakeholdern wiederum unzureichend sind.

Für die Darstellung der oberen Architekturebenen bedarf es daher einer präziseren, aber leicht verständlichen grafischen Darstellung für die Kommunikation mit allen Beteiligten.

Die Lösung für diese Darstellung ist das Blockdiagramm der FMC.

Fundamental Modeling Concepts

Die Entwicklung von FMC entstand aus einer Zusammenarbeit des Teams um Prof. Wendt an der Universität Kaiserlautern mit SAP. SAP suchte Anfang der 1990er Jahre eine Methode, um die Architektur von SAP R/3 zu dokumentieren. SAP-Mitgründer Hasso Plattner stiftete 1998 das nach ihm benannte Institut an der Universität Potsdam, an dem die Arbeiten weitergeführt wurden. 2001 schließlich wurde die Methode *Fundamental Modeling Concepts*, kurz FMC, vorgestellt.

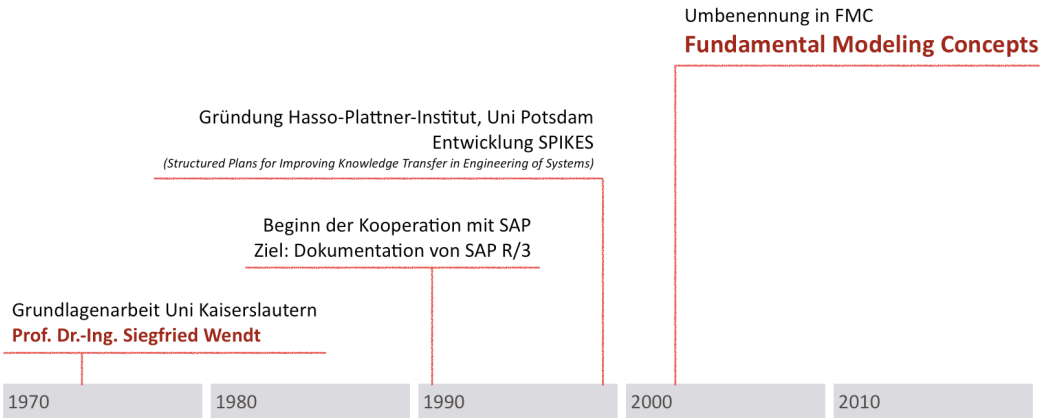

Abbildung 2: Die Entstehung der FMC

Die FMC bietet drei zentrale Diagrammtypen:

- Das Blockdiagramm stellt den strukturellen Aufbau des Systems als Komposition von miteinander verbundenen Systemkomponenten in Form von Agenten zur Funktionsausführung und passiven Datenspeichern dar.
- Petri-Netze werden für die dynamischen Abläufe im System verwendet.
- Entity-Relationship-Diagramme dienen der Darstellung von Datenstrukturen und deren Beziehungen zueinander.

Von diesen Diagrammtypen ist vor allem das Blockdiagramm beachtenswert, weil es auf einfache Weise ermöglicht, den statischen Aufbau eines Systems zu visualisieren. Die UML bietet kein vergleichbares Diagramm.

Die Möglichkeiten der Petri-Netz-Darstellung (die auf die Arbeiten von Carl A. Petri in den 1960er Jahren zurückgeht) finden sich in vergleichbarer Form auch im UML Aktivitätsdiagramm wieder. Das E/R-Diagramm der FMC bietet zwar eine innovative Darstellung von Vererbungsstrukturen, ist aber ansonsten nicht mächtiger als das UML Klassendiagramm.

Tatsächlich hat auch SAP die FMC als Methode nicht komplett übernommen. SAP hat mit TAM (*Technical Architecture Modeling*) ein eigenes Modellierungs-framework aufgebaut. TAM definiert eine Reihe von UML-Diagrammen und verbindet das FMC Blockdiagramm mit dem Komponentendiagramm der UML.

FMC Blockdiagramm

Das FMC Blockdiagramm kommt mit wenigen einfachen Grundelementen aus, die je nach Einsatzzweck, Architekturebene, Sicht usw. für verschiedene Aspekte des darzustellenden Systems genutzt werden können.

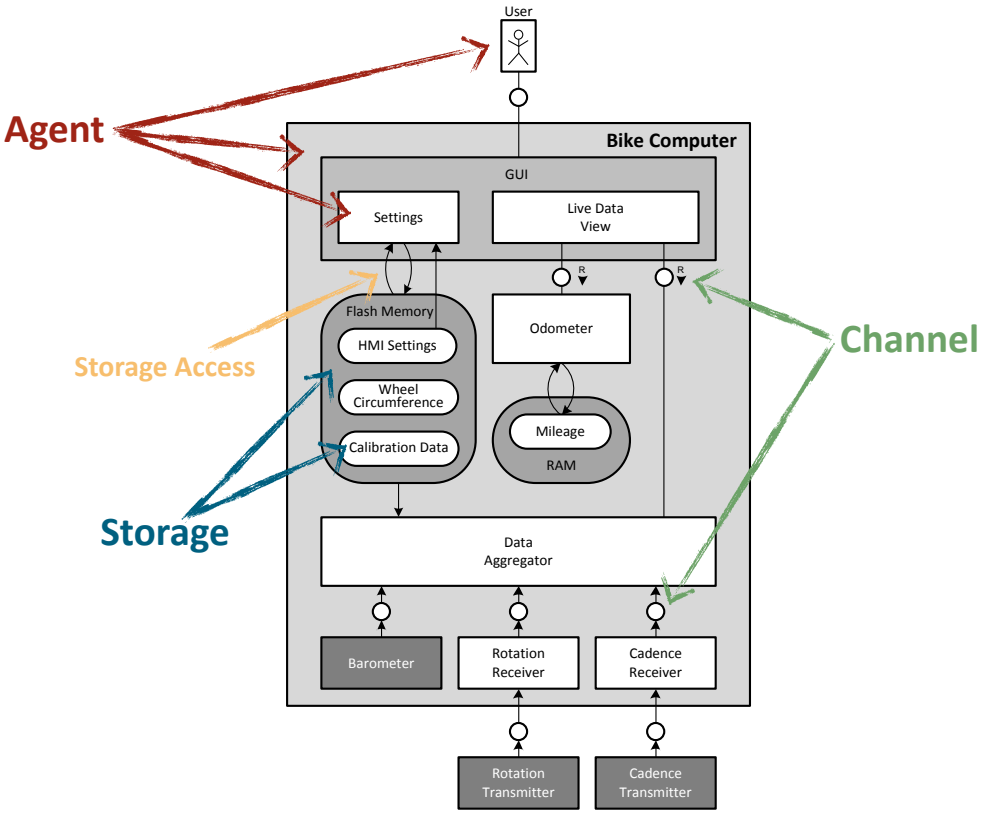

Abbildung 3: Beispiel für ein FMC Blockdiagramm

Abbildung 3 zeigt die fachliche Architektur eines Fahrradcomputers, ein einfaches eingebettetes System. In diesem Beispiel werden bereits die drei grundsätzlichen Elemente des Blockdiagramms sichtbar:

Agenten (*Agents*) sind aktive Systemkomponenten, die einem bestimmten Zweck dienen. Agenten können sowohl Softwaresysteme, -komponenten und -module, als auch Personen und Personengruppen, aber auch beispielsweise Unternehmen, Abteilungen etc. darstellen.

Speicher (*Storages*) sind passive Systemkomponenten. Agenten legen Informationen in diesen Speichern ab. Diese behalten die Informationen bis zum Überschreiben. Speicher können einfache Dateien, Filesysteme, Register, Flash RAM, größere Elemente wie Datenbanken, Cloud-Speicher aber auch Dienste wie E-Mail-Postfächer sein.

Der Zugriff von Agenten auf Speicher kann lesend oder schreibend erfolgen. Dies wird über Pfeile dargestellt. Damit kann in einem komplexen System sofort aufgezeigt werden, welche Daten woher kommen und von welchen Agenten sie gelesen oder auch manipuliert werden.

Die Kommunikation zwischen Agenten erfolgt über sogenannte Kanäle (*Channels*). Diese können unterschiedliche Kommunikationsmechanismen darstellen, von einfachen Prozeduraufrufen, der Prozess-zu-Prozess-Kommunikation via globale Variablen, Netzwerkmechanismen auf unterschiedlichen Ebenen (Seriell, TCP/IP, HTTP, Web Services, ...) bis zum Versenden per E-Mail.

Kanäle sind entweder ungerichtet (bidirektional/multiplex oder die Kommunikationsrichtung ist zu diesem Zeitpunkt im Projekt noch nicht definiert) oder gerichtet (Sender-Empfänger-Kommunikation, simplex). Eine besondere Form der Kommunikation stellt der Request-Response-Mechanismus dar, eine synchrone Kommunikation mit Rückantwort.

Gerade die Request-Response-Kommunikation ist ein wichtiges Darstellungselement der Architektur, zeigt sie doch sofort, welche Agenten die Kontrolle haben und welche als Dienste fungieren.

In der Praxis

Der Autor hat das FMC Blockdiagramm in zahlreichen Projekten in verschiedenen Domänen und für sehr unterschiedliche Arten von Architekturen und Sichten verwendet:

- Im Rahmen eines Forschungsprojekts in der Automobilindustrie wurde mit der Methode ein großes Übersichtsbild für eine komplexe Telematiklösung mit verschiedenen Frontends und einem mehrteiligen Backend erstellt. Dieses „Big Picture“ diente während der Projektlaufzeit als zentrales Element für die Kommunikation von Architekten und Entwicklern im Team (Abbildung 4 zeigt dieses Bild – die Details sind allerdings unlesbar gemacht).
- Im Zuge der Bewertung einer bestehenden Architektur eines Steuergeräts im Automobilsektor wurde mit dem Entwicklerteam zunächst ein FMC Blockdiagramm erstellt, das die Architekturebenen – vor allem in der Basissoftware – separiert und damit gezeigt hat, welche Architekturelemente der bisherigen Architektur nicht sauber in diese Ebenen passten. Auf dieser Grundlage wurde ein Refactoring-Projekt geplant und umgesetzt.
- Da in einem Projekt der Build-Prozess eine hohe Komplexität aufwies und nicht dokumentiert war, wurde mit dem Kundenteam ein FMC Blockdiagramm erstellt, das den kompletten Prozess mit seinen Artefakten und Prozessschritten zeigt.

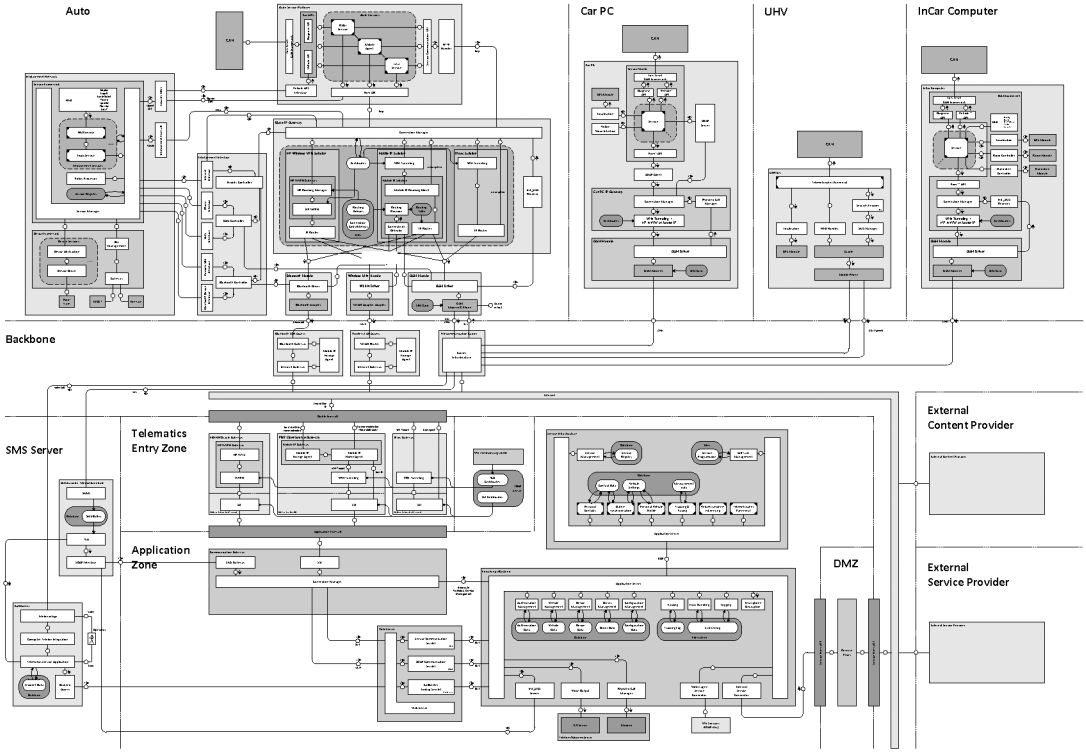

Abbildung 4: Big Picture eines Telematik-Systems in der Automobilindustrie

- In einem Projekt im Anlagenbau wurde das FMC Blockdiagramm genutzt, um die Architektur der einzelnen Anlagenbestandteile und der zahlreichen PC-basierten Tools für Fertigung, Lizenzierung, Konfiguration, Software-Updates usw. darzustellen. Im Gegensatz zum obigen Telematik-Beispiel ist hier nicht ein großes Bild, sondern eine Reihe von Einzelbildern entstanden, die als Grundlage für alle technischen Diskussionen im Team dienen.

Das FMC Blockdiagramm ist durch die offene Definition der Diagrammelemente für alle Aspekte der Software, ja sogar für darüber hinaus gehende Konzepte wie Build-Prozesse oder sogar Geschäftsprozesse nutzbar.

Der Autor setzt das FMC Blockdiagramm wie in den Beispielen zu sehen im Regelfall für die Darstellung der oberen Ebenen der fachlichen Architektur ein. Hier hat sich gezeigt, dass die UML oft zu nah an der tatsächlichen Implementierung ist, zu viele Details erfordert, die Kombination aus aktiven Komponenten und Daten nicht unterstützt und somit für eine Diskussion über die fachlichen Strukturen des Systems nur bedingt geeignet ist.

Detaillierungsgrad und Sichten

Durch die vielfältigen Möglichkeiten ist es zunächst nicht offensichtlich, welche Elemente man in welcher Detailtiefe am besten einsetzen sollte.

Es ist daher empfehlenswert, sich im Projekt darauf zu einigen, auf welchem Detaillierungsgrad und für welche Zielgruppe man die Architekturelemente darstellt. Oft verwendet man dann zusätzliche Diagramme für die detaillierte Sicht auf die dahinter liegenden Mechanismen.

Abbildung 5 zeigt ein einfaches Beispiel einer Heizungssteuerung durch eine App. Gezeigt wird hier die fachliche Sicht, bei der die App per WLAN mit der Heizungssteuerung verbunden ist. Für die Diskussion auf fachlicher Sicht mit den nicht-technischen Stakeholdern und bei der Arbeit auf Anforderungsebene ist dies zunächst die richtige und Übersichtlichkeit schaffende Darstellung.

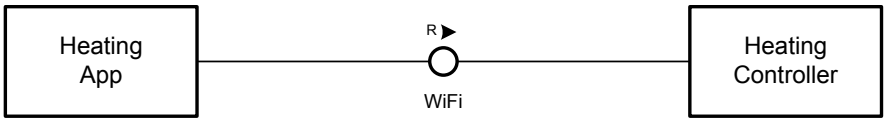

Abbildung 5: Darstellung einer Heizungssteuerung per App

Wenn innerhalb der Technik jedoch aufgezeigt werden soll, welche Komponenten an dieser Netzwerkkommunikation beteiligt sind und wie das Zusammenspiel der einzelnen Komponenten funktioniert, kann die Kommunikation auch im Detail erläutert werden (Abbildung 6).

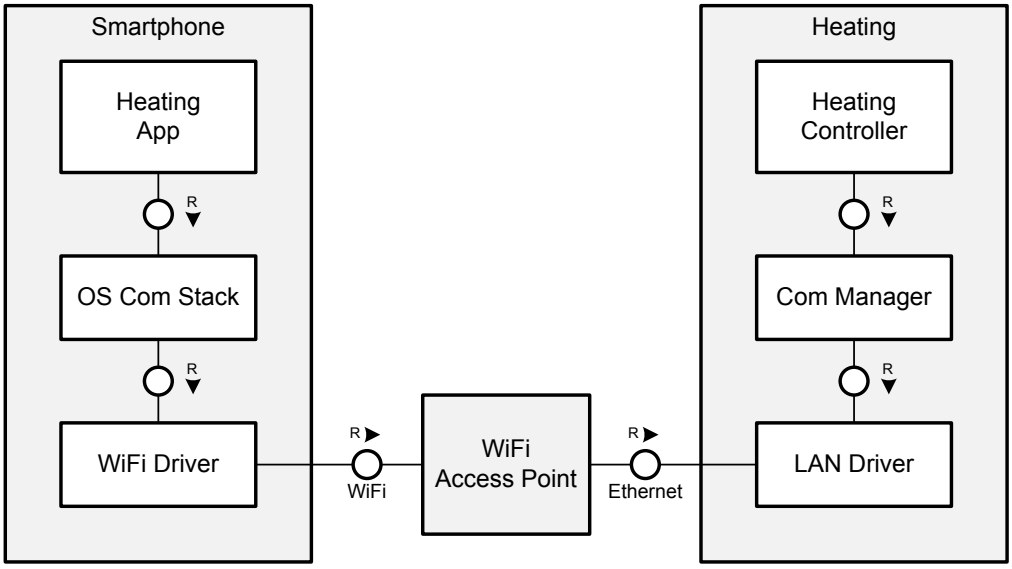

Abbildung 6: Die Kommunikationsdetails der Heizungssteuerung

Ähnlich gelagert ist die Darstellung von zentralen Middleware-Komponenten. Oft werden alle Komponenten nur mit dieser Middleware als zentralem Kommunikationsmechanismus verbunden. Dies führt zwar zu einer kreuzungsfreien Darstellung, jedoch zeigt es nur die technische Sicht – es ist nicht klar, welche fachlichen Komponenten miteinander interagieren.

In diesem Fall ist es sinnvoll, die technische Sicht (Komponenten kommunizieren über eine Middleware) darzustellen und die fachliche Architektur (welche Komponenten interagieren miteinander) ohne die Middleware in einem separaten Diagramm aufzuzeigen.

Tipps & Tricks

Der Umgang mit dem FMC Blockdiagramm, gerade in Verbindung mit Microsoft Visio (und den FMC Stencils) als Werkzeug, ist recht einfach. In der Praxis haben sich dabei folgende Vorgehensweisen bewährt:

- Die Darstellung sollte zunächst nur mittels Graustufen erfolgen. Dies lässt sich auf jedem Schwarzweiß-Drucker drucken und wird von jedem verstanden (Farbenblindheit!). Farben können dann punktuell für Hervorhebungen (Wer macht was? Alt vs. neu) genutzt werden.
- Bei den Graustufen empfiehlt es sich, definierte Tonwerte für die einzelnen Sachverhalte durchgängig zu verwenden. So nutzt der Autor dunkle Grautöne für Hardwareelemente, mittlere für Speicher, helle für Systeme/Subsysteme und weiße Kästen für die einzelnen Elemente (Komponenten, Entitäten im Speicher).
- Die Darstellung wird auf Management-Ebene oft als „technisch“ abgetan und ignoriert. Um ein FMC Blockdiagramm dennoch auf dieser Ebene nutzen zu können, können Farben und Farbverläufe gezielt eingesetzt werden.
- Da die Übersichtlichkeit der Darstellung wichtig für die Verständlichkeit ist, sollte mit *Align to Grid* gearbeitet werden und Linienfluchten usw. eingehalten werden.
- Linien sollten wenn immer möglich kreuzungsfrei sein.
- Die Kreuzungsfreiheit ist eine wesentliche Metrik für eine gute Architektur. Im Zweifelsfall sollte man bestimmte Sachverhalte (z.B. ein zentrales Logging, das von praktisch allen Komponenten genutzt wird) eher in eigene Sichten auslagern.
- Bei der Darstellung von Varianten können in bestimmten Fällen Visio-Layer verwendet werden.

Beim Umgang mit dem FMC Blockdiagramm hilft es in jedem Fall, eine einheitliche Vorgehensweise im Team (bzgl. Detaillierungsgrad, Graustufen etc.) festzulegen, damit die Details im Diagramm sofort von jedem verstanden werden.

Auf der FMC-Homepage und im Buch von Knöpfel et. al. (siehe Literaturhinweise) sind weitere Guidelines für die Modellierung zu finden.

Zusammenfassung

Das FMC Blockdiagramm hat sich in der Praxis der Architekturarbeit in zahlreichen Projekten immer wieder bewährt. Bei der Zerlegung der Problemstellung in fachliche Agenten, der Einbeziehung der gespeicherten Daten, der Beschreibung von Kontrollflüssen und Zuständigkeiten sowie der Hervorhebung von System- und Komponentengrenzen liefert die Methode sehr gute Ergebnisse und bildet somit eine wesentliche Grundlage für die Kommunikation im Team.

Während die UML ihre Berechtigung in der präzisen Modellierung von Software zur Spezifikation oder gar zur (modellbasierten) Programmierung hat, eignet sich das FMC Blockdiagramm vor allem als Basis für die Teamkommunikation und die Darstellung von ausgewählten Aspekten der Software für die Stakeholder.

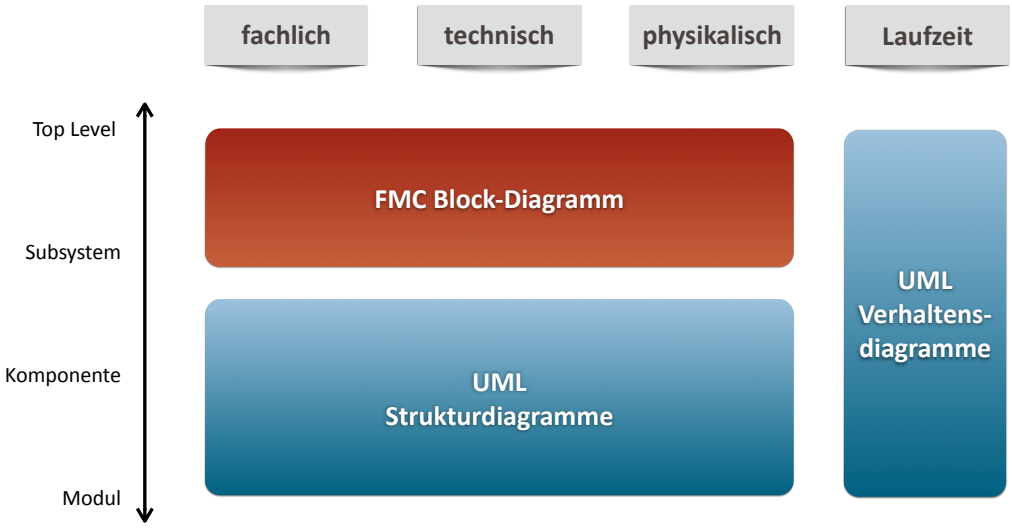

Abbildung 7: Diagramme für Architekturebenen und Sichten

Das FMC Blockdiagramm empfiehlt sich vor allem für die oberen Ebenen der Softwarearchitektur und kann in der Praxis die UML Diagramme für andere Sichten und detaillierte Architekturebenen ergänzen (Abbildung 7). Darüber ist das Block-diagramm ein Ersatz für unpräzise PowerPoint-Diagramme.

Das FMC Blockdiagramm ist somit ein wesentliches Element im Werkzeugkasten von Softwarearchitekten.

Literatur- und Quellenverzeichnis

Andreas Knöpfel; Bernhard Gröne; Peter Tabeling:
Fundamental Modeling Concepts – Effective Communication of IT Systems
Chichester, 2006

Stefan Zörner:
Softwarearchitekturen dokumentieren und kommunizieren – Entwürfe, Entscheidungen und Lösungen nachvollziehbar und wirkungsvoll festhalten
München, 2012

Block Diagrams – Reference Sheet
http://www.fmc-modeling.org/download/notation_reference/Reference_Sheet-Block_Diagram.pdf

Fundamental Modeling Concepts – Visualization Guidelines
http://www.fmc-modeling.org/visualization_guidelines

Standardized Technical Architecture Modeling – Contepual and Design Level
http://www.fmc-modeling.org/download/fmc-and-tam/SAP-TAM_Standard.pdf

TAM Stencils for Visio 2002
http://www.fmc-modeling.org/download/tam_stencils/TAM_Stencils.zip

Über den Autor

Dr. Jörg-Volker Müller ist Gründer und Geschäftsführer der Systemum GmbH & Co. KG. Er ist seit mehr als 20 Jahren in leitender Funktion in der Softwareentwicklung tätig und hat erfolgreich IT-Systeme, Frameworks und Softwareprodukte mit unterschiedlichen Teams realisiert. Seine fachlichen Schwerpunkte sind vor allem Architekturen und Prozesse in der Entwicklung von Embedded Software und IT-Lösungen. Seit 2012 hat er einen Lehrauftrag für Softwaretechnik an der Hochschule Harz.