Generische Visualisierung strukturierter Daten
Generic Visualisations of Structured Information

Steffen Kruse und Philipp Gringel
OFFIS – Institut für Informatik, Oldenburg, Deutschland, {kruse,gringel}@offis.de

Kurzfassung

Das Enterprise Architecture Management (EAM) beschäftigt sich mit der Verwaltung und Planung der Anwendungslandschaften von Unternehmen. Hier können Visualisierungen unterstützen, wenn sie exakt und aktuell sind. Leider ist die Erstellung guter Visualisierungen eine aufwendige und fehleranfällige Aufgabe und ihre Pflege eine repetitive manuelle Tätigkeit. Dieser Beitrag beschreibt, wie maßgeschneiderte Visualisierungen automatisch generiert werden können, um exakte und aktuelle Sichten auf unternehmensspezifische Modelle zu bieten.

Abstract

A central concern of Enterprise Architecture Management (EAM) is the management and planning of enterprise application landscapes. This task can be supported by visualisations, if they display accurate and up to date information. Unfortunately, the creation of good visualisations is a complex and error-prone task and their maintenance a repetitive manual activity. This paper describes how customised visualisations can be automatically generated from enterprise-specific models to provide accurate and up to date views.

1 Einleitung

Komplexe Anwendungslandschaften von Unternehmen sind ähnlich wie Städte, über lange Zeit gewachsene, komplexe und heterogene Systeme, die ständigen Veränderungen unterliegen. Um die Komplexität einer Stadt begreifbar zu machen und ihre Entwicklung zu steuern, haben sich in der Stadtplanung die Darstellungen komplexer Zusammenhänge durch Visualisierungen bewährt. Dazu gehören zum Beispiel diverse Arten von Karten, wie Katasterkarten oder thematische Karten zur Bevölkerungsdichte. Ähnlich wie bei der Planung komplexer physischer Systeme, lassen sich solche Visualisierungstechniken auch auf Unternehmensmodelle anwenden ([18]).

Dieser Beitrag stellt in Abschnitt 2 die Domäne EAM als Grundlagen und Kontext für die Visualisierungen vor. Im folgenden Abschnitt werden ein Generierungsprozess und das zugehörige Werkzeug *JavaViz*[1] erläutert, welche Visualisierungen automatisch erzeugen können. Die realisierten Standardvisualisierungen werden kurz beschrieben, bevor sich Abschnitt 4 ausführlich der neuen Konsolidierungskarte widmet. Der Beitrag schließt mit einer Zusammenfassung in Abschnitt 5.

2 Grundlagen

Das Enterprise Architecture Management (EAM) ist die kontinuierliche und andauernde Managementaufgabe, die essentiellen Elemente eines Unternehmens und ihre wechselseitigen Beziehungen untereinander und zur Umwelt, zu beschreiben, zu analysieren und zu planen, um deren Komplexität verständlich zu machen und die Evolution der Organisation zu steuern (vgl. [8, 15, 16]). Die „essentiellen Elemente und deren Beziehungen“ werden hier als Unternehmensarchitektur (engl. Enterprise Architecture / EA) bezeichnet.

Die wesentlichen Aufgaben des EAM sind durch diese Definition gegeben: Die Tätigkeiten (Analyse, Planung, Gestaltung) können unter den Begriffen *EA-Dokumentation* und *EA-Weiterentwicklung* zusammengefasst werden. Die Dokumentation des Ist-Zustandes der EA bildet die notwendige Voraussetzung für eine gesteuerte Weiterentwicklung auf einen Ziel-Zustand hin (vgl. [2, 5, 4]). Die gesammelte Dokumentation der Elemente einer EA lässt sich gemeinhin als Informationsmodell begreifen. Dabei kann eine EA-Dokumentation je nach Vorgehen, Betrachtungsgegenstand und Reife des EAM sehr unterschiedlich ausfallen: diese reicht von fragmentierten Tabellen und textuellen Berichten bis hin zu spezialisierten EA-Werkzeugen (vgl.[12]). Als hilfreiches mentales Modell wird dabei eine Strukturierung der Elemente der EA in üblicherweise mindestens drei Ebenen angesehen (vgl. [1]). Winter und Fischer schlagen zum Beispiel die fünf *Ebenen Business Architecture*, *Process Architecture*, *Integration Architecture*, *Software Architecture* und *Technology Architecture* vor [17]. Jede Ebene enthält verschiedene Modellelemente unterschiedlicher Granularität, die untereinander und ebenenübergreifend über Relationen verbunden sind.

Die Visualisierung von strukturierten Daten ist ein wichtiges Thema des EAM und spielt dort im Stakeholder Management als Kommunikationsmittel für Entscheidungsträger eine zentrale Rolle. Dabei ist die zeitnahe, aktuelle, ef-

[1] Für Projektinformationen siehe: `http://javaviz.offis.de`

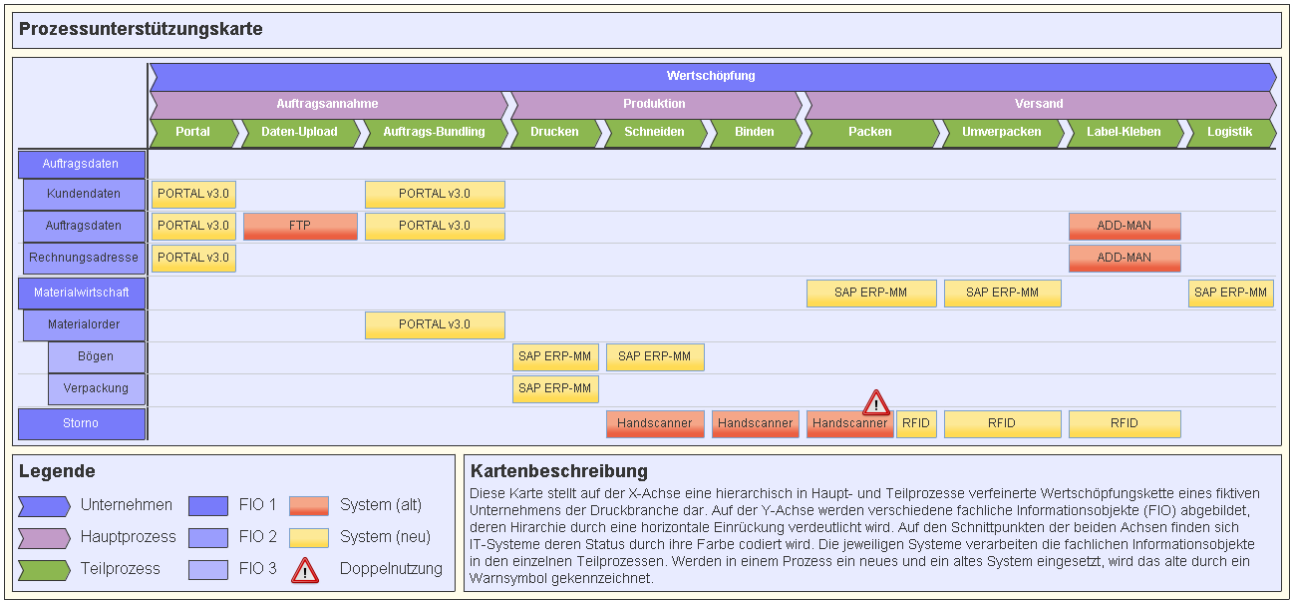

Bild 1 Beispielvisualisierung: Die Prozessunterstützungskarte

fektive und korrekte Visualisierung großer Datenmengen aufwendig. Ein den Aufwand beeinflussender Faktor sind die große Anzahl heterogener Modelle, die im EAM genutzt werden. Zwar bieten generische EA-Frameworks Referenzmodelle, jedoch müssen diese für einzelne Unternehmen oder Domänen konkretisiert und ausgeprägt werden, um zweckmäßig zu sein [15, 19, 14]. Der zweite Faktor ist die stetige Veränderung, der Unternehmen unterworfen sind. Sich wandelnde Märkte, veränderte rechtliche Anforderungen oder Änderungen der Organisationsstruktur führen zur kontinuierlichen Anpassung von Unternehmen. Die Evolution der Geschäftsanforderungen muss von den unterstützenden IT-Systemen aufgenommen ([7]) und von dem jeweiligen Unternehmensmodell reflektiert werden. Daher kann erwartet werden, dass ausgeprägte Unternehmensmodelle für jedes Unternehmen einzigartig sind und sich im Laufe der Zeit verändern ([11]). Drittens kann auch davon ausgegangen werden, dass sich z.B. die in [3] identifizierten EAM-Anliegen und genannten Lösungsansätze verändern werden. Dabei beeinflussen sowohl der Reifegrad einer Unternehmensarchitektur, als auch die Eigenheiten der Branche, in der das Unternehmen agiert, den individuellen Informationsbedarf der involvierten Stakeholder. Die drei genannten Faktoren bedingen sich ständig ändernde Unternehmensmodelle, was dazu führt, dass Visualisierungen aufwendig manuell erstellt und gepflegt werden müssen.

In [13] werden Kriterien für gute Visualisierungen von Graphen genannt, die sich auf das Layout der Knoten und Kanten sowie auf den gesamten Platzbedarf beschränken. Ähnliche allgemeingültige Kriterien für gutes Layout lassen sich für viele strukturierte Visualisierungen finden. Darüber hinaus ist es schwierig die Frage nach einer guten Visualisierung zu beantworten, da viel von der Wahrnehmung der einzelnen Menschen abhängt. „Weiche“ Kriterien sind beispielsweise Farben und Farbverläufe, Linienstärke, Muster, Beleuchtungseffekte und Schatten, 2D, 3D und animierte Graphiken. Diese Kriterien zählen zu den Visualisierungselementen, die stark von individuellen Vorlieben Einzelner abhängen und von graphischen Trends beeinflusst werden, die sich mit der Zeit ändern.

3 Prozess und Werkzeug

Um den Herausforderungen der Komplexität und der ständigen Veränderung der Datenbasis zu begegnen, wurde im JavaViz-Projekt ein Ansatz entwickelt, der die automatisierte Generierung von Visualisierungen aus vorhandenen Datensätzen ermöglicht (siehe Bild 2). Da in den meisten Fällen Unternehmen bereits über umfassende Datenmengen in elektronischer Form verfügen und diese auf die individuellen Bedürfnisse des Unternehmens abgestimmt oder aus diesen erwachsen sind, können zum Zweck der Visualisierung kaum Annahmen über Struktur oder Format der vorhandenen Daten gemacht werden. Auch wäre der Aufwand zu hoch, Daten manuell zur Visualisierung in einem bestimmten Format zu erheben oder zu konvertieren.

Daher greift JavaViz auf Techniken der Modelltransformation [10] zurück und bietet minimale View-Modelle für unterschiedliche Visualisierungstypen an. Auf der vorhandenen Datenbasis (dem Domänenmodell) wird eine „Sicht“ konfiguriert (engl. View), die der Fragestellung entspricht, die durch die Visualisierung adressiert werden soll. Ein View ist definiert als eine Sicht auf ein Modell zu einem bestimmten Zweck (hier der Zweck der Visualisierung). So lassen sich Visualisierungsmodelle von Unternehmensmodellen entkoppeln.

Die View-Modelle können als Muster verstanden werden, das auf die relevanten Teile des Unternehmensmodells gelegt wird (siehe Bild 3 für eine schematische Darstellung einiger View-Modelle). Auf Basis des Musters wird dann die gewünschte Visualisierung vollautomatisch generiert und bei Änderung an der Datenbasis jederzeit aktualisiert.

In der Generierung werden die Anordnung der graphischen Elemente einer Visualisierung und deren Gestaltung

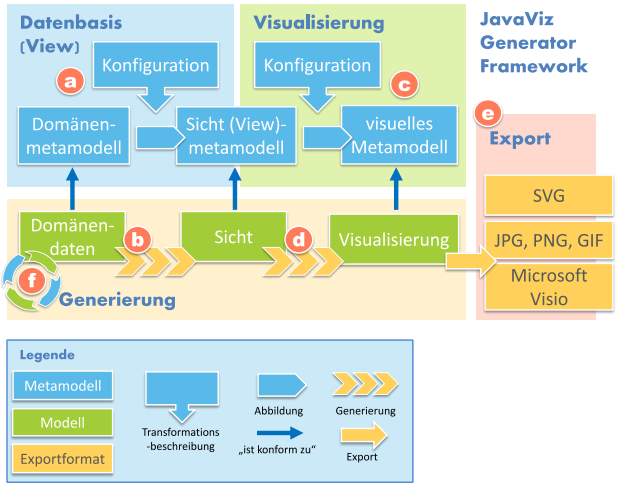

Bild 2 JavaViz Prozess zur Generierung von Visualisierungen

von kombinierten Layoutern übernommen, die jeweils auf bestimmte Elemente und Anordnungen spezialisiert sind. Die visuelle Konfiguration lässt sich zwischen den Visualisierungen teilen, wodurch der Konfigurationsaufwand reduziert und ein gleichartiges Erscheinungsbild in unterschiedlichen Kontexten ermöglicht wird. Es ist wichtig, dass der Generierungsprozess ohne manuelle Anpassungen eine akzeptable Visualisierung liefert, damit diese sofort verwendet werden kann und so z.B. in automatisierte Dokumentation, Web-Seiten oder Cockpits eingebettet werden kann.

Bild 2 zeigt eine abstrakte Übersicht des Generierungsprozesses. Er beginnt mit der Konfiguration der gewünschten Sicht ⓐ . Dieser Schritt beinhaltet die Wahl eines geeigneten Visualisierungstypen und die Entscheidung, welche Domänenelemente wo in der Visualisierung dargestellt werden sollen. Die technische Schnittstelle zur Generierung unterstützt bereits eine große Anzahl von Ausgangsformaten (u.A. Microsoft Excel, XML Formate oder Datenbankadapter) in denen Ausgangsdaten angebunden werden können. Technisch stellt die konfigurierte Abbildung eine Modelltransformation dar, die ausgeführt werden kann ⓑ.

Ist der darzustellende Ausschnitt gewählt, können die graphischen Eigenschaften konfiguriert werden ⓒ. Hier können z.B. die zu verwendenden Farben, Schriftarten und Abstände festgelegt werden. Die Konfiguration der visuellen Eigenschaften der Visualisierung kann mit anderen Visualisierungen geteilt werden.

Ist die Konfiguration abgeschlossen, kann die Visualisierung generiert werden ⓑ. Dabei werden Domänendaten ausgelesen und anhand der Abbildungsbeschreibung zwischen Domänenmetamodell und View-Modell transformiert. Im nächsten automatischen Schritt wird die generierte Sicht in graphische Elemente übersetzt und anhand der visuellen Konfiguration gestaltet ⓓ. Die endgültige Position und Größe der Elemente in der Visualisierung werden von unterschiedlichen Layoutern berechnet, um eine angemessene Darstellung der Ausgangsdaten zu erzeugen. Im Anschluss kann die generierte Visualisierung je nach Einsatzzweck in eins oder mehrere Ausgabeformate geschrieben werden ⓔ. Hier bieten sich z.B. SVG für die Darstellung auf Webseiten oder Microsoft Visio für die manuelle Nachbearbeitung an.

Der Prozess der Generierung geschieht vollautomatisch und kann bei Veränderungen in der Datenbasis jederzeit wiederholt werden, um eine aktuelle Visualisierung zu liefern ⓕ.

JavaViz stellt unterschiedliche Visualisierungstypen zu Verfügung und erlaubt bei Bedarf die Erstellung neuer Typen unter Verwendung aller bestehenden Elemente und Layouter. Die obere Reihe von Bild 3 zeigt schematische Darstellungen der vier zweidimensionalen Visualisierungstypen. Die Elemente in der aufgespannten Fläche werden mit den Elementen der Achsen in Beziehung gesetzt. Platzhalter für Titel, Legende und einen beschreibenden Text sind als schraffierte Flächen dargestellt. Die schematischen Darstellungen in der unteren Reihe von Bild 3 stellen die Visualisierungstypen vor, die die Elemente unabhängig von Referenzachsen miteinander in Beziehung setzen. Bild 1 ist eine Beispielvisualisierung, die Zusammenhänge zwischen Prozessschritten (X-Achse, hierarchisch), den sie unterstützenden IT-Systemen (Flächenelemente) und den die Systeme nutzenden Organisationseinheiten (Y-Achse, hierarchisch), darstellt.

4 Beispiel einer Visualisierung: Die Konsolidierungskarte

Die Konsolidierungskarte wurde entwickelt, um Fragestellungen im Kontext von Unternehmensübernahmen zu beantworten. Der in [6] entwickelte Kartentyp wird im Folgenden anhand des dort beschriebenen, motivierenden EAM-Szenarios vorgestellt.

Typische Treiber des EAM sind Unternehmenszusammenschlüsse, wie *Mergers* and *Aquisitions*. Hierbei stellt sich für die IT die Frage, wie die beiden Anwendungslandschaften (AL) der zusammenzuführenden Unternehmen zu konsolidieren sind. Neben einem kompletten Neubau der AL für das fusionierte Unternehmen, kann auch eine AL ersatzlos abgeschafft werden, und nur eine AL weiterbetrieben werden. Ein weiteres Verfahren ist das sog. *Cherry Picking*, bei dem sich eine neue AL aus den Systemen der beiden zu fusionierenden Unternehmen ergibt [9]. Besonders die letzte Strategie profitiert von der Konsolidierungskarte. Bild 4 enthält die graphische Darstellung einer Sicht auf ein Cherry-Picking-Szenario.

Die Transformation einer Anwendungslandschaft bei einem Zusammenschluss ist in der Regel ein langwieriger Prozess, der als internes Projekt mit diversen Unterprojekten gestaltet wird. Typischerweise müssen einzelne Softwaresysteme ausgewählt, migriert oder ersetzt, abgeschaltet oder umgezogen werden. Die Systeme können dabei wieder aus diversen Subsystemen bestehen, die jeweils einzeln betrachtet und behandelt werden können. Dabei kann der Umbau der Systeme als Sub-Projekte geplant, gestaltet und durchgeführt werden. In der Regel geschieht der Um-

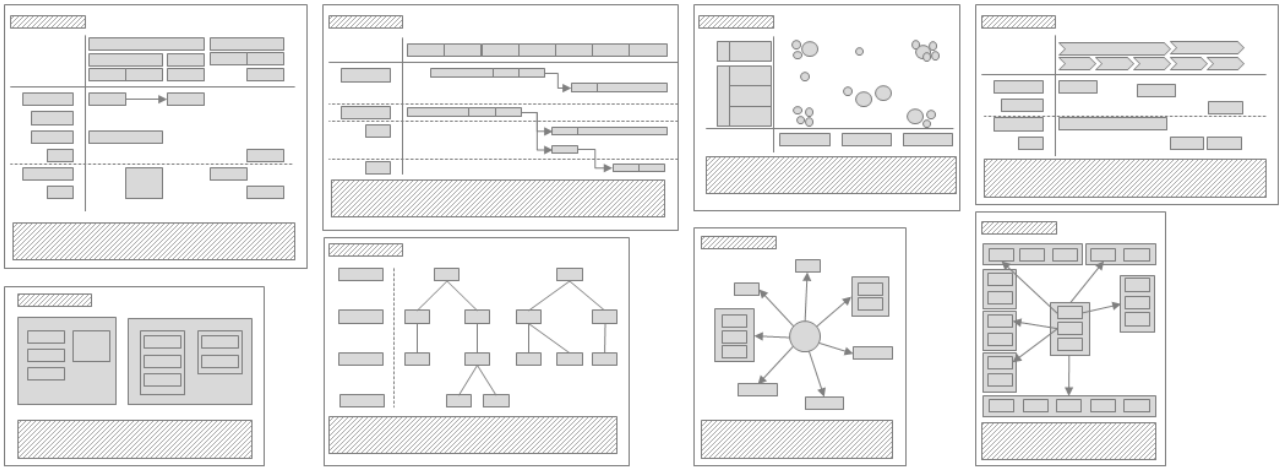

Bild 3 Oben: Visualisierungstypen, die zwei Achsen nutzen und Elemente in der aufgespannten Fläche mit den Achsenelementen in Beziehung setzen. Unten: Visualisierungen ohne Achsen als Bezugssystem.

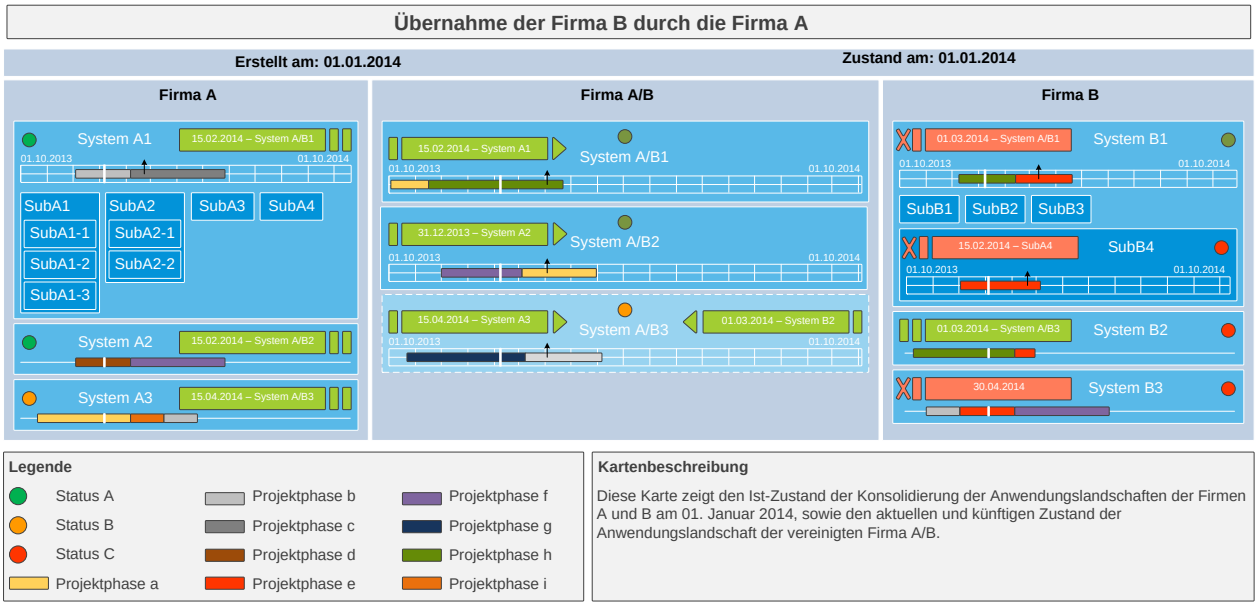

Bild 4 Beispielvisualisierung: Die Konsolidierungskarte

bau nicht zu einem festen Zeitpunkt für alle Systeme sondern wird nach und nach durchgeführt, um den regulären Betrieb so wenig wie möglich zu beeinflussen. Die drei beteiligten Anwendungslandschaften (die zwei der fusionierenden Unternehmen und die neue, gemeinsame Anwendungslandschaft) haben dabei zu jedem Zeitpunkt einen definierten Zustand, der z.B. die bereits durchgeführten Maßnahmen, alle geplanten Maßnahmen und die Zustände der einzelnen Projekte und Systeme umfasst. Die Konsolidierungskarte soll dabei helfen, den Zustand einer Konsolidierung zu einem gewählten Zeitpunkt übersichtlich darzustellen (siehe Bild 4).

Im oberen Bereich bietet die Karte Platz für einen Titel, der im Beispiel in Bild 4 beschreibt, dass die *Firma A* eine *Firma B* übernommen hat. Darunter findet sich links das Erstellungsdatum der Visualisierung und rechts daneben findet sich das Datum, auf welches sich die dargestellten Informationen beziehen. So können vergangene, aktuelle (wie in der Abbildung) und künftige (Plan-) Zustände der drei ALs der beiden Ausgangsunternehmen und des fusionierten Unternehmens visualisiert werden. Letzteres wird immer in der Mitte dargestellt und heißt in diesem Beispiel *Firma A/B*. Die Systeme der AL von *Firma A* werden durch Rechtecke dargestellt, die jeweils oben einen Bezeichner führen (*System A1* bis *System A3*). Dabei zerlegen sich diese Systeme weiter in Subsysteme, die ihrerseits teilweise eigene Subsysteme haben.

Links neben dem Bezeichner eines Systems ist ein farbiger Kreis dargestellt, der den Status der Konsolidierung des betreffenden Systems codiert. Alle drei Systeme A1-A3 enthalten Zeitleisten, wobei die von A1 mehr Informationen enthält als die von A2 und A3. Allen drei ist gemeinsam, dass sie einen Zeitraum von einem Jahr darstellen, die Dauer von Projektphasen durch farbige Rechtecke und das Kartenerstellungsdatum (01.01.2014) durch eine vertikale weiße Linie wiedergeben. In der erweiterten Zeitleiste von

System A1 sind Start- und Enddatum, sowie ein senkrechter schwarzer Pfeil dargestellt. Dieser Pfeil markiert einen Meilenstein im Projektverlauf.

Das vom Pfeil markierte Datum, (der Meilenstein am 15.02.2013 in System A1) stimmt mit dem jeweils oben rechts platzierten Pfeiltypen überein. Es gibt drei Pfeiltypen, die bei Firma A jeweils rechts oben in den Systemen, bei Firma B links oben in den Systemen und bei den Systemen der Firma A/B in den beiden oberen Ecken platziert werden können. Diese Pfeiltypen bestehen aus einem Rechteck mit Text, einem kleinen Rechteck als Zwischensymbol und einem Endstück, dass ein Dreieck, ein Andreaskreuz oder ein weiteres kleines Rechteck sein kann.

Der erste Pfeiltyp, siehe Systeme A1, A2, A3 und B2 repräsentiert eine im Vergleich zum Zustandsdatum (01.01.2014) noch anstehende Migration. Der Pfeil zeigt dabei in die Richtung der AL in die das Anwendungssystem migriert wird, bei A1-A3 nach rechts, bei B2 nach links. Da Anwendungssysteme im Zuge der Migration umbenannt werden können, trägt der Pfeil im Quellsystem den Namen des Zielsystems und umgekehrt. Im Pfeil von System A1 ist notiert, dass das System am 15.02.2014 in das System A/B1 umbenannt wird.

Der zweite Pfeiltyp (siehe Systeme B1, SubB4 und B3) bedeutet, dass das System durch das im Pfeil angegebene System zum entsprechenden Datum abgelöst wird. System B1 wird am 01.03.2014 durch System A/B1 abgelöst, was durch eine Migration von System A1 am 15.02.2014 aus der AL der Firma A in die AL der Firma A/B hervorgegangen ist. Dies drückt aus, dass die fusionierte Firma A/B ein System der Firma A unter neuem Namen fortführt und ein funktional redundantes System der Firma B abschaltet. Die Veränderungen werden in den Systemen der mittleren AL der Landschaft A/B nachvollzogen. Bei System A/B2 kann aus dem grünen Migrationspfeil abgelesen werden, dass es am 31.12.2013 aus dem System A2 hervorgegangen ist.

Das System A/B3 ist mit einem gestrichelten Rahmen dargestellt, da dieses System sich zum aktuellen Datum noch nicht in der fusionierten AL befindet. Werden diese Systeme mit in die Visualisierung einbezogen, kann der Projektplan des Konsolidierungsprojekts mit Fokus auf die IT-Systeme nachvollzogen werden. An System A/B3 kann abgelesen werden, dass es die Funktionen der beiden Systeme A3 und B2 am 15.04.2014 bzw. am 01.03.2014 zusammenführt.

5 Fazit

JavaViz ermöglicht die vollautomatische Generierung von Visualisierungen aus bestehenden Datenbasen, mit dem Ziel, ohne manuelle Nachbearbeitung ein akzeptables Ergebnis zu produzieren. Dabei ist die Frage, was eine akzeptables Ergebnis ausmacht, nicht immer leicht zu bestimmen. Zum einen kann eine sehr große Menge an Ausgangsdaten oder deren Zusammenhänge eine sinnvolle Darstellung verhindern. Ein einfaches Beispiel für diesen Fall liefern Graphen: ab einer bestimmten Menge von Knoten und / oder Kanten sind statische Darstellungen trotz ausgeklügelten Layout-Strategien nicht mehr praktikabel. Ein weiterer Einflussfaktor bei der Nutzerakzeptanz von Visualisierungen ist das ästhetische Empfinden, das auch vom individuellen Geschmack abhängt. Außerdem spielt der intendierte Einsatzzweck eine wichtige Rolle bei der Bewertung einer Visualisierung. Eine Darstellung kann sehr gut geeignet sein für die Beantwortung einer Fragestellung und gleichzeitig völlig ungeeignet sein für eine andere (auch wenn die gewählte Sicht auf die Daten für beide Fragestellungen geeignet wäre.) Diese Faktoren lassen sich nur sehr schwer oder gar nicht formalisieren und deshalb nur zu einem gewissen Grad in eine automatische Generierung einfließen lassen.

Für den Fall, dass eine generierte Visualisierung für einen bestimmten Einsatzzweck nicht ausreicht, wurden Mechanismen geschaffen, um manuelle Anpassungen möglichst einfach vornehmen zu können. So bietet jeder Visualisierungstyp vielfältige Konfigurationsmöglichkeiten für den Generierungsprozess. Diese reichen von Farbschemata bis hin zu den von Layoutern berücksichtigten Pixelabständen zwischen Elementen und Elementgruppen. Die Standardkonfiguration jeder Visualisierung dient als Ausgangspunkt und ist grundsätzlich mit sinnvollen Einstellungen belegt. Anpassungen der Generierung gelten dabei für alle Instanzen einer Visualisierung.

Sollen einzelnen Instanzen einer Visualisierung angepasst werden (z.B. für den „letzten Schliff“ für die Verwendung in Microsoft PowerPoint-Präsentationen oder Microsoft Word-Dokumenten), eignet sich der Export nach Microsoft Visio, um die Darstellung manuell an die individuellen Vorstellungen anzupassen. Zwar ist so jede erdenkliche Art der Anpassung möglich, jedoch lassen sich diese Änderungen nicht automatisieren und gehen bei einer erneuten Generierung der Visualisierung bei veränderten Daten verloren.

Die hier als Beispiel vorgestellte Konsolidierungskarte kann neben der Konsolidierung von Anwendungslandschaften auch genutzt werden, um andere Szenarien darzustellen, beispielsweise die Veränderung von Organisationseinheiten der fusionierenden Unternehmen. Links und rechts wäre jeweils die Ausgangssituation und in der Mitte die neue, zusammengeführte Struktur dargestellt. Denkbar wäre auch die Übertragung von Produkten in neue, gemeinsame Bündelprodukte oder eine Zusammenlegung von Geschäftsfeldern und Geschäftsprozessen.

Derzeit wird das in Java geschriebene JavaViz mit einem eigenen Cross-Compiler nach JavaScript übersetzt, um eine native Verwendung der Visualisierungen in HTML5 zu ermöglichen. Damit werden mobile Endgeräte als Plattform für Management-Cockpits als Endgeräte erschlossen.

6 Literatur

[1] J. S. Addicks, “Bewertung betrieblicher Anwendungen im Kontext ihrer Unternehmensarchitektur,” Dissertation, Carl von Ossietzky Universität Oldenburg, Oldenburg, 2010.

[2] S. Aier, C. Riege, and R. Winter, “Unternehmensarchitektur - Literaturüberblick und Stand der Praxis,”

Wirtschaftsinformatik, vol. 50, no. 4, pp. 292–304, 2008.

[3] S. Buckl, A. M. Ernst, J. Lankes, and F. Matthes, "Enterprise architecture management pattern catalog (version 1.0, february 2008)," Chair for Informatics 19 (sebis), Technische Universität München, München, Technischer Bericht TB 0801, 2008.

[4] G. Engels, A. Hess, B. Humm, O. Juwig, M. Lohmann, J.-P. Richter, M. Voß, and J. Willkomm, *Quasar Enterprise: Anwendungslandschaften serviceorientiert gestalten*, 1st ed. Heidelberg: dpunkt.verlag GmbH, 2008.

[5] I. Hanschke, *Strategisches Management der IT-Landschaft: Ein praktischer Leitfaden für das Enterprise Architecture Management*. München: Carl Hanser Verlag, 2009.

[6] T. Hellkamp, "Zeitorientierte Visualisierungen von Unternehmensarchitekturen: Konzept einer Konsolidierungskarte," Bachelorarbeit, Fakultät II - Department für Informatik, Abt. Business Engineering, Carl von Ossietzky Universität Oldenburg, Oldenburg, Jun. 2014.

[7] J. C. Henderson and N. Venkatraman, "Strategic Alignment: Leveraging Information Technology for Transforming Organizations," *IBM Systems Journal*, vol. 32, no. 1, pp. 472–484, Jan. 1993.

[8] ISO/IEC/IEEE, "International Standard ISO/IEC/IEEE 42010:2011(E): Systems and Software Engineering - Architecture Description," pp. 1–46, 2011.

[9] W. Keller, *IT-Unternehmensarchitektur: Von der Geschäftsstrategie zur optimalen IT-Unterstützung*, 2nd ed. Heidelberg: dpunkt.verlag GmbH, 2012.

[10] S. Kruse, J. S. Addicks, M. Postina, and U. Steffens, "Decoupling Models and Visualisations for Practical EA Tooling," in *Proceedings of the 2009 International Conference on Service-Oriented Computing, ICSOC/ServiceWave '09 Workshops*, A. Dan, F. Gittler, and F. Toumani, Eds. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 62–71.

[11] S. Kurpjuweit and R. Winter, "Viewpoint-based Meta Model Engineering," in *Proceedings of the 2nd International Workshop on Enterprise Modelling and Information Systems Architectures - Concepts and Applications (EMISA)*, ser. P-119, M. Reichert, S. Strecker, and K. Turowski, Eds. Gesellschaft für Informatik (GI), 2007.

[12] F. Matthes, S. Buckl, J. Leitel, and C. M. Schweda, *Enterprise Architecture Management Tool Survey 2008*. Chair for Informatics 19 (sebis), Technische Universität München, 2008. [Online]. Available: http://wwwmatthes.in.tum.de/wikis/sebis/eamts2008

[13] O. Niggemann and B. Stein, "A Meta Heuristic for Graph Drawing: Learning the Optimal Graph-drawing Method for Clustered Graphs," in *Proceedings of the Working Conference on Advanced Visual Interfaces (AVI)*. New York, NY, USA: Association for Computing Machinery (ACM), 2000, pp. 286–289.

[14] J. Schekkerman, *How to Survive in the Jungle of Enterprise Architecture Framework - Creating or Choosing an Enterprise Architecture Framework*, 2nd ed. Trafford Publishing, 2004.

[15] The Open Group, *Open Group Standard TOGAF Version 9.1*. Van Haren Publishing, Dec. 2011.

[16] P. van der Merwe, *A Definition for Enterprise Architecture*. The Open Group, Architecture Forum, 2009.

[17] R. Winter and R. Fischer, "Essential Layers, Artifacts, and Dependencies of Enterprise Architecture," *Journal of Enterprise Architecture*, vol. 3, no. 2, pp. 7–18, May 2007.

[18] A. Wittenburg, "Softwarekartographie: Modelle und Methoden zur systematischen Visualisierung von Anwendungslandschaften," Dissertation, Technische Universität München, Institut für Informatik, München, 2007.

[19] J. A. Zachman, "A Framework for Information Systems Architecture," *IBM Systems Journal*, vol. 26, no. 3, pp. 276–292, Sep. 1987.