OBJEKTspektrum
ONLINE Themen special

Fachartikel

Modellgetriebener Ansatz zur Optimierung von Shopfloor-Datenstrukturen für den Einsatz einer Streaming Data Platform

Shopfloor-Enhancement für Industrie 4.0

von Johannes Mickel

Durch die Definition neuer Geschäftsprozesse im Rahmen der Industrie 4.0 entstehen stetig neue Anwendungsfälle, die Einfluss auf die existierenden Daten haben und somit auch neue funktionale Anforderungen an die Architektur des IT-Shopfloors beziehungsweise an die IoT-Plattform stellen. Der Artikel zeigt, welche Auswirkungen die Industrie 4.0 auf das Design des Shopfloors und die IoT-Platform hat. Im Fokus stehen hierbei die modellgetriebene Optimierung der Datenstrukturen und die daraus resultierenden zusätzlichen Einflussfaktoren auf die neue IoT-Plattform – die Streaming Data Platform.

Definition iIoT

Industrial Internet of Things (vgl. [ItWi16])

- industrielles Konzept des Internet of Things (IoT)
- Fokus liegt auf dem Einsatz von IoTs im Industrieumfeld
- Erfassung von großen Datenmengen (Datenströmen) durch iIoTs, dazu zählen:
  - Sensor-Netzwerke und intelligente Computer-Netzwerke,
  - intelligente und connected Devices, Smart Objects, RFIDs
- verwendete Protokolle:
  - OPC-UA (Open Platform Communications Unified Architecture)
  - MQTT (Message Queuing Telemetry Transport)
- Ziele:
  - Effizienzsteigerung in der Herstellung (lernende Maschinen und Automaten)
  - flexiblere Produktionstechniken
  - Predictive Maintenance
  - Smart Manufactoring
- Einsatzszenarien:
  - Smart Grids, E-Health
  - Frühwarnsystem, Kommunikation zwischen Fahrzeug und Infrastruktur (C2I, C2C)

Kasten 1

Ein Blick auf die aktuelle Situation am Markt lässt sehr schnell erkennen, dass es durch den rasanten Anstieg der Datenmengen zu einer buchstäblichen „Datenexplosion“ gekommen ist: vertikal entlang der Anwendungsschichten sowie horizontal entlang der Wertschöpfungskette.

Ein wesentlicher Treiber ist der immense Anstieg an eingesetzten IoTs und im Industrieumfeld der iIoTs **(siehe Kasten 1)**. Hinzu kommt die Anforderung an eine Echtzeitübertragung. Das bedeutet, es wird eine Architektur/Plattform verwendet werden müssen, welche die Fähigkeit besitzt, sehr schnell sehr viele Daten zu übertragen, diese auszuwerten und weiterzuleiten. Solch eine Streaming Data Architecture oder Streaming Data Platform gilt es, unter Hinzunahme der aktuellen und zukünftigen Datenstruktur aufzubauen.

Im Artikel möchte ich Ihnen zeigen, wie der modellgetriebene Ansatz zu Analyse und Design der Datenstruktur die Entwicklung neuer fachlicher und technischer Anforderungen bis hin zum (Re-)Design der IT-Architektur unterstützt und verbessert. Im Fokus soll hierbei die vertikale, datenorientierte Integration im Unternehmen stehen.

Grundlegende Aspekte der modellgetriebenen Analyse und Design

Modelle bieten nicht nur einen sehr guten Überblick über die aktuelle Architektur und die enthaltenen Funktionalitäten einer Software oder einer Anwendungslandschaft, sondern auch die Möglichkeit, die Zusammenhänge zwischen den verschiedenen Teilbereichen der Softwareentwicklung zielgruppengerecht darzustellen. Zudem können Verbindungen von einem Geschäftsprozess über den Anwendungsfall und die dafür entwickelte Funktionalität im System bis hin zu den verwendeten und erzeugten Daten hergestellt werden.

Die vertikale Integration ermöglicht es, jedes Datum/Attribut einem oder mehreren Geschäftsprozessen zuzuordnen. Dieses Mapping sichert nicht nur Konsistenz innerhalb der IT-Architektur und eine hohe Produktivität der Anwendung, sondern lässt zusammen mit der horizontalen Integration der Systeme und Daten auch die Komplexität der Unternehmensanwendungen beherrschbar machen.

Eine weitere, nicht zu vernachlässigende Stärke der Modellierung beziehungsweise der eingesetzten Modellierungstools ist die Generierung von Programmcode (Java-Klassen, war-Dateien usw.), DSL-Dateien, Datenbankschemata und Konfigurationsdateien. Dadurch wird ein hohes Maß an Standardisierung im Entwicklungsprozess geschaffen und zudem der Grundstein für eine sehr gut strukturierte Codebasis gelegt.

Der Datenlage Herr werden

In den meisten Unternehmen liegt immer noch eine sehr heterogene, proprietäre und zum Teil undokumentierte IT- und Fertigungsinfrastruktur vor. Unternehmen haben gerade durch EAM (Enterprise Architecture Management) in den letzten Jahren schon einen großen Schritt nach vorne gemacht, was die Beherrschbarkeit der eigenen IT-Landschaft angeht. Trotzdem gibt es hier noch Verbesserungsbedarf. Daten werden teilweise noch von Mitarbeitern händisch erfasst und im Nachhinein in Excel-Tabellen übertragen. Im besten Fall gelangen diese Daten im Anschluss noch in ein ERP-System.

Jeder einzelne Bereich hat hierbei sein eigenes Vorgehen und zum Teil obendrein seine eigene Datenhaltung. Die jetzt schon eingesetzten Sensoren, meist integriert in den Maschinen, senden kontinuierlich einfache Daten (Messdaten). Dieser Datenstrom ist auf den von der Maschine eingesetzten heterogenen Bereich isoliert und ohne Bezug zu beispielsweise den Produktdaten, obwohl er gerade im Hinblick auf Predictive Maintenance (die „vorausschauende Wartung“) ein wichtiger Bestandteil ist.

Ziel ist es unter anderem, nicht nur die eingesetzten Maschinen zu überwachen und zu kontrollieren, sondern auch zu jeder Zeit auf die Daten des Produktes in all seinen Zuständen zugreifen zu können und gegebenenfalls Anpassungen im laufenden Produktionsbetrieb vorzunehmen. Um dieses Ziel zu erreichen, ist eine Integration der Maschinen und der benötigten Devices des iIoTs in die restliche IT-Landschaft des Unternehmens notwendig.

Das Ziel ist definiert, nur wie wird es erreicht? Die Analyse und das Managen der Daten gerade im iIoT-Umfeld ist eine der großen Herausforderungen in diesem Anwendungskontext.

Ein bedeutender erster Schritt ist, sich einen Überblick über die vorliegenden Daten im Unternehmen zu verschaffen. Neben dem Systemkontext und der Prozesslandkarte des eigenen Unternehmens beziehungsweise der eigenen IT-Landschaft ist die Erstellung einer „Datenlandkarte“ ein bewährtes Mittel zum Erfolg. Folgende grundlegenden Fragen sollten dabei beantwortet werden:

Welche Daten sind im Unternehmen vorhanden?
Welche Eigenschaften besitzen diese Daten noch?
Welchen Strukturierungsgrad besitzen die Daten?
Stammdatum oder nicht?
Woher stammen diese Daten (Kunde, Produkt, Zulieferer, Maschinen, Software, Unternehmensbereich)?
Wie gelangen diese Daten ins Unternehmen (Webservice, Sensoren, Mail, Eingabe durch Mitarbeiter)?
Wo sind diese Daten gespeichert/abgelegt (Anwendung, Datenbank)?
Verarbeitung der Daten im Unternehmen?
Gibt es jetzt schon sichtbare Verbindungen zwischen diesen Daten?

Um hier den Überblick nicht zu verlieren, sollte frühzeitig damit begonnen werden, ein konzeptionelles Fachdatenmodell aufzubauen. Hilfreich, um an die Daten in einem Unternehmen zu kommen, ist auch die Möglichkeit des toolunterstützten Imports von Quellcode oder SQL-Schemata der eingesetzten Datenbanken in das Modell.

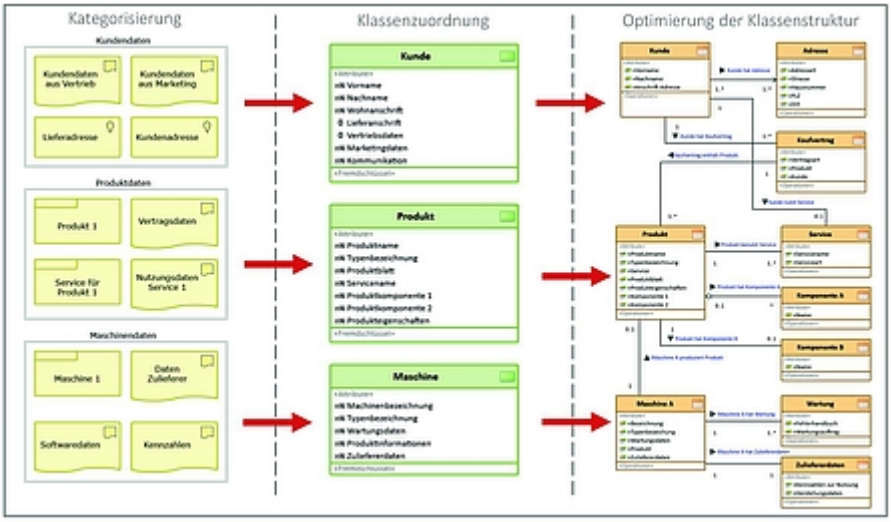

Abb. 1: Von Kategorisierung der Daten bis Optimierung der Klassenstruktur

Nachdem die Daten vorliegen, werden sie auf ihre Art, Zugehörigkeit und Eigenschaften analysiert und kategorisiert. Eine Kategorisierung in zum Beispiel Maschinen-, Betriebs- und Softwaredaten, Kommunikations-, Interaktions- oder Verhaltensdaten sowie Kunden- und Produktdaten kann zum besseren Verständnis und zur Einordnung der Daten sehr hilfreich sein. Im Anschluss an die Kategorisierung erfolgt die Zuordnung der Daten als Attribute zur jeweiligen Fachklasse oder Komponente.

Schaut man sich nur allein die Kundendaten an – zum Beispiel Name, Anschrift, Kommunikation usw. –, wird schnell ersichtlich, dass es meistens nicht ausreicht, nur eine Fachklasse ‚Kunde‘ zu erstellen und dieser dann sämtliche Daten mit Kundenbezug zuzuordnen beziehungsweise als Attribut hinzuzufügen. Für den ersten Überblick würde dies ausreichen, für eine spätere Weiterverarbeitung und Auswertung oder mit dem Ziel des Kundenzuwachses und des einhergehenden Datenanstieges nicht.

Hier empfiehlt sich eine weitere Optimierung der Datenstruktur beziehungsweise eine weitere Aufteilung der Fachklassen, die in Verbindung mit der ‚Kunde‘-Fachklasse stehen **(siehe Abbildung 1)**. Jedes Datum sollte im Anschluss zu einer Fachklasse in Verbindung stehen und jede Fachklasse darf nur ein einziges Mal vorhanden sein. Wichtig ist hierbei, dass der Kontext der Fachklasse und deren Attribute, sprich durch wen werden die Daten verwendet und erzeugt, nicht verloren geht. Dies kann durch Hinzufügen von Metainformationen und Assoziationen zu entsprechenden Anwendungsfällen und Anforderungen aus den jeweiligen Anwendungen oder Komponenten sichergestellt werden. Des Weiteren ist es hilfreich, für die Anwendungsfälle zusätzlich Aktivitätsdiagramme zu modelliert, um auch die Aktionen innerhalb eines Anwendungsfalls zu verstehen und fachlich besser bewerten zu können.

Auch darf die Klassifizierung der Daten nicht vergessen werden. Gerade die Einstufung der Daten in öffentliche und nicht öffentliche Daten hat später Auswirkung auf deren Verwendung und nicht zuletzt auch auf die Zugriffssteuerung der Securitykomponente innerhalb der Software/Plattform. Abhängig von der Größe des Unternehmens und der Produkte

beziehungsweise deren Komplexität können auch multidimensionale Modellierungsmethoden zum Tragen kommen. Gerade durch die Modellierung der Stammdaten im Bereich der Kunden, der Produkte und eingesetzten Maschinen wird ein hohes Maß an Strukturierung und in Hinblick auf die weitere Nutzung der Daten auch ein hohes Maß an Standardisierung gewonnen. Es handelt sich hierbei um eine datenzentrierte und bereichsübergreifende Fachmodellierung, die den Kern der IT-Landschaft in einem Unternehmen bildet. Dies ist gerade bei Unternehmen mit heterogener IT-Landschaft wichtig, um ein einheitliches, unternehmensweites Verständnis der Daten zu gewinnen.

Kontinuierliche, einfache Daten

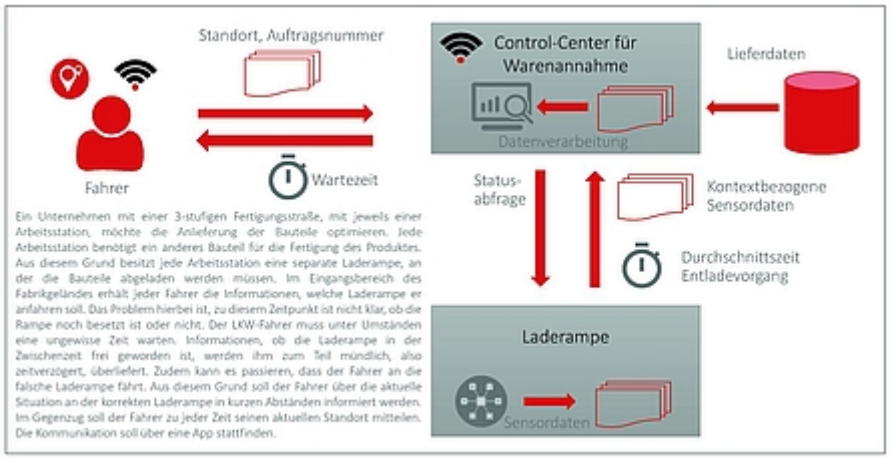

Abb. 2: Anwendungsbeispiel Digitalisierung/Optimierung des Control-Centers für die Warnannahme (vgl. [DpDh17])

Die Stammdaten sind im Modell, fehlen noch die Daten, die eventbasiert und zum großen Teil kontinuierlich in großen Mengen erstellt werden. Die Rede ist im Industrie 4.0-Umfeld unter anderem von den Daten, die durch die iIoTs erzeugt werden. Hierzu gehören zum Beispiel Umgebungsdaten (Temperatur, Geschwindigkeit, Füllstand usw.), Standortdaten (GPS), Unique Identifier Daten (RFID), deskriptive Daten und Netzwerkdaten. Hier lohnt es sich, zunächst einen eigenen in sich abgeschlossenen Bereich im Modell nur für die Sensorik aufzubauen, der im Anschluss durch Aggregation in das globale Fachmodell integriert wird. Nehmen wir dazu das Beispiel aus **Abbildung 2**. Folgende Daten sind in diesem Szenario wichtig:

Position des LKWs,
Status der Laderampen und des Abladefortschrittes,
Daten für die Lieferung (Art der Bauteile, Anzahl, Auftragsnummer usw.).

Die Position des LKWs (GPS-Daten) sowie Auftragsnummer für die Lieferung werden über das Smartphone an das Control-Center für die Anlieferung übermittelt. Der Status der Laderampe wird zum Beispiel über einen Lichtschrankensensor abgerufen und vom Control-Center an den Fahrer weitergegeben. Ist zudem noch gefordert, dass auch eine Wartezeit übermittelt werden soll, müssen noch weitere Daten ermittelt werden. Wie viele Bauteile hat der Vorgänger-LKW insgesamt geladen, wie viele sind schon abgeladen und wie lange dauert das Abladen eines Bauteiles im Durchschnitt? Hieraus kann zu jeder Zeit die Wartezeit berechnet werden.

Aus dem vereinfachten Beispiel lassen sich nun folgende Hauptanforderungen an das System ableiten:

schneller Zugriff auf die Sensordaten,
schneller Zugriff auf Lieferdaten,
schnelle Berechnung von Wartezeiten,
interne und externe ereignisgesteuerte Echtzeitkommunikation,
Übertragung und Verarbeitung von Datenströmen.

Bedeutend hierbei ist, dass für diesen Geschäftsprozess nicht nur die Sensordaten benötigt werden, sondern zudem auch die Stammdaten (Daten zur Lieferung und zur Arbeitsstation) und zusätzlich ein Koeffizient für die Berechnung der Wartezeit für die jeweilige Laderampe. Bei den Sensordaten handelt es sich um eine einfach strukturierte Datenstruktur, aber in Summe um eine kontinuierlich eintreffende große Anzahl.

Abb. 3: Daten werden durch Kontextbezug zu Informationen

Die Modellierung der Sensoren beziehungsweise der Sensordaten konzentriert sich im ersten Schritt auf zwei wesentliche Punkte – die „rohen“, unverarbeiteten Sensordaten und die eigentliche, transportierte Information, welche über den Kontextbezug entsteht **(siehe Abbildung 3)**. In unserem vereinfachten Beispiel benötigen wir zwei Sensoren für die Laderampe. Eine Lichtschranke, die den LKW in der Laderampe erfassen kann, und einen Sensor (Scanner), der die abgeladenen Bauteile erfasst. Für die zwei unterschiedlichen Sensoren wird jeweils ein eigenes Modell erstellt. Als Nächstes müssen die Informationen ausgewertet werden. Auf Modellebene bedeutet dies, dass die einzelnen Sensor-Modelle in unser globales Modell aggregiert werden. Da sie beide zur Laderampe gehören, werden sie der Komponente ‚Laderampe‘ zugeordnet. Die Laderampe kann nun durch das Verwenden der Informationen der zwei Sensoren Auskunft über den eigenen Status geben.

Die Komponente ‚Laderampe‘ soll aber nicht nur diese Informationen weitergeben, sondern auch die durchschnittliche Zeit für das Abladen eines Bauteiles bereitstellen. Dieser Koeffizient wird hier vereinfacht berechnet aus der Zeit, die zwischen dem Scannen der einzelnen abgeladenen Bauteile vergeht. Weitere Faktoren, wie zum Beispiel die gesamte Verweildauer jedes einzelnen LKWs an der Laderampe, könnten die Durchschnittsdauer zusätzlich beeinflussen.

Im Control-Center werden zur Berechnung der Wartezeit für den nächsten LKW jetzt die berechnete Durchschnittszeit, die Anzahl der Bauteile der Gesamtlieferung (Ermittlung aus dem Lieferschein) und die Anzahl der schon abgeladenen Teile herangezogen. Das Control-Center benötigt also nicht nur die Sensordaten der Laderampe, sondern auch die Information zur aktuellen Lieferung.

Und hier kommt nun einer der wesentlichen Vorteile der Modellierung zum Tragen – das Verdeutlichen von Verbindungen und Abhängigkeiten zwischen den Daten, Klassen beziehungsweise den Komponenten. Dabei ist es zudem hilfreich, folgende Beziehungen darzustellen und auswerten zu können:

zwischen Daten und Anwendungsfällen,
zwischen Daten und den Geschäftsprozessen,
zwischen Daten und den fachlichen Repräsentanten der iIoT-Komponenten,
zwischen den Daten und den Softwarekomponenten bis hin zu den einzelnen Funktionen (fachlich und technisch),
zwischen den Daten und den Metainformationen.

Die Information, wie das Gesamtvolumen einer Lieferung, kann nur ermittelt werden, wenn das Control-Center entweder direkten Zugriff auf die entsprechenden Daten in der Datenbank hat oder durch die Nutzung eines Service mit der Sendungsnummer, die vom Fahrer übergeben wurde. Dies bedeutet für das Control-Center eine Anbindung (horizontale Integration) an ein weiteres System/Anwendung, das solch einen Service anbietet.

Alles strömt zusammen – Streaming Data/Event Streams

Durch den Einsatz von Sensoren, die permanent eventbasiert Daten versenden, ändert sich das Bild der im Unternehmen anfallenden Daten erheblich. Es sind nun nicht mehr nur Stammdaten, die versendet, verarbeitet und gespeichert werden, sondern Datenströme aus unterschiedlichen Quellen. Jedes Datum wird letztendlich durch ein Event erstellt. Also liegt uns ein kontinuierlicher Strom von Events mit den zugehörigen Daten vor. Hier den Überblick zu gewährleisten, ist gerade in Hinblick auf die stetig wachsende Anzahl an iIoTs im Unternehmen eine Herausforderung. Von Vorteil ist es, zum einen zu wissen, welche Events innerhalb einer Domäne eintreffen können, und zum anderen zu wissen, welche Daten in welchem Format zu jenem Ereignis vorliegen.

Um herauszufinden, beziehungsweise später auch zu definieren, welche Ereignisse zu einer Domäne gehören, gibt es zum Beispiel die Methode des „Event Storming“ (vgl. [Wikia]). Mit dieser Methode lassen sich schnell alle Ereignisse in einer Domäne auffinden. Sie bildet damit eine sehr gute Grundlage für die Geschäftsprozessmodellierung. Neben den Ereignissen an sich sind

aber auch die vorkommenden Daten zu jenem Zeitpunkt wichtig. Nehmen wir aus unserem Beispiel den Scanprozess der abgeladenen Bauteile an der Laderampe. Folgende Daten werden zu diesem Zeitpunkt erzeugt: ProduktID und Zeitpunkt der Erfassung. Diese sollen im Anschluss zusammen mit der ID des Scanners weitergeleitet werden.

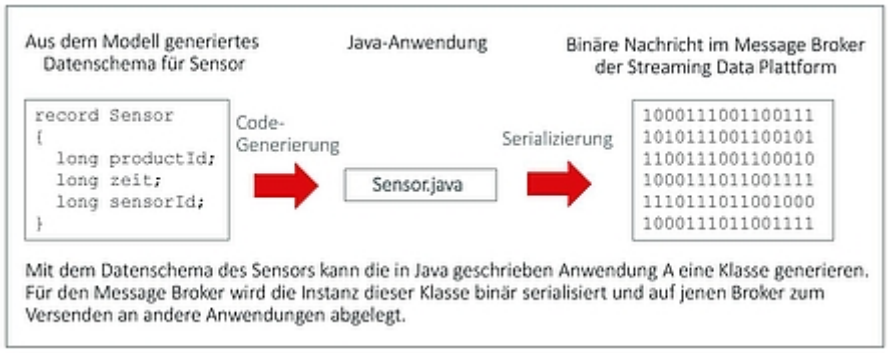

Abb. 4: Codegenerierung und Serialisierung von Daten zur Übertragung als Strom in einer Streaming Data Platform

Ein wichtiger Aspekt hierbei ist das Format der Daten. Da höchstwahrscheinlich diese Daten nicht nur in einem System verwendet werden, sondern unternehmensweit zur Verfügung stehen müssen, ist es von großer Bedeutung, dass hierfür ein Standardformat festgelegt ist. Somit kann sich auch jedes zugreifende System darauf verlassen. Zum Einsatz kommen hier Modellierungsformate, mit denen plattformunabhängige Schemata (IDL – Interface Definition Language, JSON usw.) erstellt werden können. Diese Schemata können aus dem Modell direkt herausgeneriert werde. Damit ist sichergestellt, dass jedes System mit der gleichen Datenstruktur arbeitet. In den jeweiligen Systemen werden dann für die jeweils unterstützte Programmiersprache die benötigten Klassen zur Laufzeit erstellt und weiterverwendet **(siehe Abbildung 4)**.

Bekannte Beispiele von Frameworks zur Erstellung solcher Formate sind Apache Thrift oder Google Protocol Buffers. Eine der Stärken dieser Modellierungsformate ist die Möglichkeit der Verwendung von verschiedenen Versionen der Schemata. Das bedeutet, muss durch eine Änderung in der Datenstruktur das Schema angepasst werden, so muss nicht jeder Client oder Server sofort mitgeändert werden. Doch bevor diese Frameworks zum Einsatz kommen, muss klar definiert sein, welche Daten zu welchem Zeitpunkt von welchem System benötigt werden.

Transparenz schaffen

Wir können nun durch das „Event Storming“ eine Aussage über die Events, die Daten und die Akteure in einer Domäne beziehungsweise in einem Geschäftsprozess treffen. Wir wissen, welche Daten durch die jeweiligen Sensoren bereitgestellt werden. Was jetzt noch wichtig ist, sind die Verbindungen zwischen den Geschäftsprozessen und den Funktionen sowie zu den bereitgestellten Daten der Sensoren.

Hierbei kann man die Verknüpfungsmöglichkeiten zwischen BPMN (Business Process Model and Notation) und der UML (Unified Modeling Language) nutzen, da mit der UML Komponenten, Klassen und Funktionen der Anwendung entworfen werden können. Jeder Geschäftsprozess besteht mindestens aus Tasks und Datenobjekten (hier: unsere Eventdaten). Aus jedem Task lässt sich mindestens ein Anwendungsfall (Use Case, UML) für das System extrahieren und verknüpfen. Das BPMN-Datenobjekt aus dem Geschäftsprozess wird mit der entsprechenden (Daten-)Fachklasse und obendrein mit einem Zustand verbunden. Die Fachklasse beinhaltet nicht nur die Daten (= Attribute), sondern auch die benötigten Funktionen, um die Daten bereitzustellen.

Ein nächster Schritt ist das „Verheiraten“ der Daten mit den Anwendungsfällen. Dies wird über die Funktionen (Fachoperationen) der einzelnen Fachklassen realisiert. Dabei wird eine Assoziation zwischen dem Anwendungsfall und der benötigten Funktion im Fachklassenmodell gezogen.

Dadurch wird jetzt Transparenz zwischen den Geschäftsprozessen, den Anwendungsfällen und den dafür verwendeten Daten geschaffen. Das Ergebnis kann zum Beispiel aufzeigen, dass für den Ablauf eines einzigen Geschäftsprozesses mehrere Anwendungsfälle und somit auch verschiedenste Daten aus unterschiedlichen Fachklassen verwendet beziehungsweise erzeugt werden.

In unserem Beispiel besteht der Geschäftsprozess ‚Warenannahme’ aus mehreren Tasks (Anmelden des Fahrers, Lieferung

abladen usw.). Der Task ‚Lieferung abladen' steht wiederum in Verbindung mit mehreren Anwendungsfällen (Prüfen der Laderampe, Scannen der Bauteile, Berechnung der Abladezeit) und den Datenklassen aus beiden Sensoren (Lichtschranke und Scanner). Bei der Analyse und Optimierung von Geschäftsprozessen können so auch die dazugehörigen Daten besser und effizienter herangezogen und ausgewertet werden.

Auch können die Auswirkungen auf die Shopfloor-IT beziehungsweise auf die Architektur schnell und in einem frühen Stadium sichtbar gemacht werden. Mögliche Wechselwirkungen zu anderen Prozessen und somit auch auf andere Daten können frühzeitig erkannt und Gegenmaßnahmen eingeleitet werden.

Hierbei hilft zum Beispiel die Durchführung einer Impact Analyse, welche die Verbindungen (Assoziationen, Generalisierungen usw.) zwischen den einzelnen Elementen nutzt, um solche Wechselwirkungen zu erkennen. Wird zum Beispiel in einem Prozess ein bestimmtes Datenobjekt nicht mehr benötigt, kann, bevor es gelöscht wird, geprüft werden, ob dies noch in einem anderen Prozess oder Anwendungsfall verwendet wird. Zudem können betroffene Klassen und dadurch auch die betroffene Komponente lokalisiert werden – ein großer Vorteil der Modellierung des gesamten Systems.

Hilfreich ist es auch, wenn bei der Modellierung mit Aktivitäts-, Sequenz- und Verteilungsdiagrammen gearbeitet wird. Dadurch können nicht nur die Reihenfolge, die zeitlichen und logischen Ablaufbedingungen sowie die Schleifen und Nebenläufigkeiten des Informationsaustauschs abgebildet werden, sondern auch Aktionen, Kontroll- und Objektflüsse innerhalb eines Systems und zwischen anderen Systemen. Gerade bei den Schnittstellen zu anderen Systemen ist es extrem wichtig, zu wissen, welche Daten und Artefakte übertragen werden – und in welcher Ausprägung. Jedes Partnersystem muss sich diesbezüglich darauf verlassen können.

Globales Fachmodell

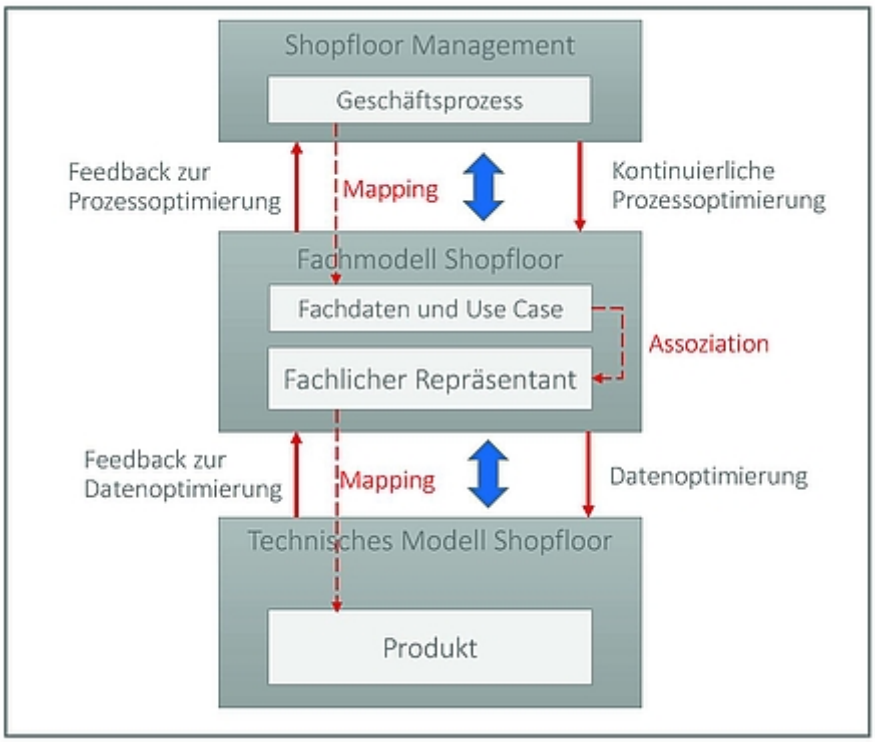

Abb. 5: Mapping zwischen Fachmodell und technischem Modell

Am Ende erhalten wir ein Fachmodell des eigenen IT-Shopfloors, das ein Abbild des eigentlichen, physischen Shopfloors darstellt und die Verbindung zwischen dem technischen Modell des Shopfloors und dem Shopfloor-Management (vgl. [Wikib])mit all seinen Geschäftsprozessen herstellt **(siehe Abbildung 5)**. Hierbei werden sogenannte fachliche Repräsentanten (Komponenten und Fachklassen) aufgebaut, die jeder für sich ein Produkt oder eine Komponente aus dem technischen Modell repräsentieren.

Hinzu kommen die Daten, Kennzahlen und auch die Anwendungsfälle, die in Verbindung mit dem jeweiligen fachlichen Repräsentanten stehen. Über die Anwendungsfälle und Datenklassen wird die Verbindung zu den Geschäftsprozessen hergestellt.

Die richtige Entscheidung treffen

Gerade bei der Automatisierung und Digitalisierung von bestehenden Geschäftsprozessen oder der Einführung komplett neuer Geschäftsmodelle ist es umso wichtiger zu wissen, welche neuen Anforderungen sich für die bestehende IT-Architektur/Plattform ergeben – technisch wie fachlich. Der Fokus liegt hierbei nicht nur darauf, Produkte auf den Markt zu bringen, die mit neuer Technologie (Sensorik usw.) ausgestattet werden sollen. Ein weiteres Ziel ist zudem, den Kunden bei der Nutzung der Produkte durch die Bereitstellung neuer Services (Apps usw.) zu unterstützen und ihn somit auch enger an das Unternehmen zu binden.

Des Weiteren können durch die digitale Transformation von internen Geschäftsprozessen eines Unternehmens auch Partner, wie die Zulieferer von Teilkomponenten, stärker ins Unternehmen eingebunden werden und gleichermaßen davon profitieren, wie zum Beispiel durch eine Verkürzung der Wartezeit ihrer Fahrer. Anhand des oben aufgeführten, vereinfachten Beispiels lassen sich die Anforderungen an die IT-Architektur/Plattform sehr gut extrahieren:

Umgang mit großen Datenmengen,
Umgang mit Datenströmen,
automatische, eventbasierte, schnelle Datenverarbeitung,
interne und externe Echtzeitkommunikation,
Skalierbarkeit.

Bei der anstehenden Entscheidung, welche Architektur für die vorherrschenden und zukünftigen Daten die geeignetste ist, sollten des Weitern folgende Faktoren mitberücksichtigt werden:

aktuelle und zukünftige Geschäftsmodelle (Prozesse, Produkte, Services),
Digitalisierungsgrad der Produkte,
aktuelle und zukünftige Daten (Struktur, Menge, Herkunft und Verwendung),
aktuelle Enterprise- und IT-Architektur-Vorgaben.

Klassische Architekturen/Plattformen können die oben aufgeführten Anforderungen, gerade im Bereich Echtzeitverarbeitung und -kommunikation mit Datenströmen, nicht mehr erfüllen. Eine hierfür immer mehr an Bedeutung gewinnende Zielarchitektur ist die der Streaming Data Architektur (vgl. [Wam16]). Bisher ist hierfür die Lambda-Architektur (vgl. [Lamb]) in verschiedensten Ausprägungen gesetzt gewesen.

Ein bekannter Vertreter im Open-Source-Bereich ist Apache Hadoop im Zusammenspiel mit Apache Spark und Druid. Schwachpunkt dabei sind die Latenzen. Und hier setzt das Konzept der Kappa-Architektur **(siehe Abbildung 6)** an. Sie soll die Lambda-Architektur flach und simpel halten. Der Fokus liegt hier auf dem Streaming-Prozess. Die Batch-Verarbeitung, wie in der Lambda-Architektur, fällt komplett weg. Zum Verteilen aller Daten wird ein Streaming-Messaging-System, wie zum Beispiel Apache Kafka (vgl. [ApKa]), verwendet. Im Anschluss werden die Daten durch ein Stream-Processing-Framework (z. B. Flink, Akka Stream, Spark Streaming) verarbeitet. Je nach Anwendungsfall werden die Daten über das Messaging-System an die gewünschte Anwendung oder zur Speicherung an eine Datenbank weitergeleitet.

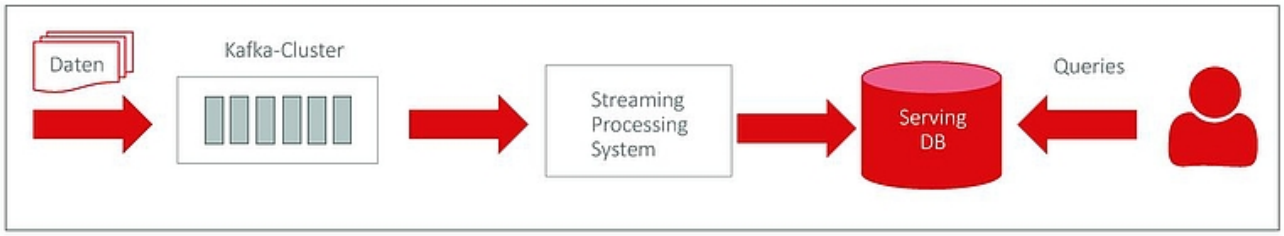

Abb. 6: Kappa-Architektur am Beispiel von Apache Kafka

Um die Anforderungen und Abläufe für solch eine Architektur korrekt zu erfassen, sind gerade Sequenz- und Verteilungsdiagramme sehr hilfreich und aussagekräftig. Hierdurch können Aussagen zur Erzeugung, Transport und Weiterverarbeitung bis hin zur Speicherung der Daten getroffen und somit die Anforderungen an die Architektur konkretisiert werden.

Ausblick: Horizontale Integration der Daten

Gerade in einer heterogenen IT-Landschaft besitzt jedes System zum größten Teil seine eigene Datenhaltung – historisch gewachsen oder gewollt, mit dem Ziel eines unabhängigen Software-Entwicklungsprozesses und -Deployments. Ein Abgleich gerade in Bezug auf die redundant gehaltenen Daten ist sehr aufwendig, zeitintensiv und bedarf einer strengen Kontrolle und festen Vorgaben – besonders an etwaigen Schnittstellen zu anderen Systemen. Ein Beispiel sind hier die Kundendaten, die im Vertrieb und gleichzeitig im Marketing erzeugt und weiterverarbeitet werden, aber aus unterschiedlichen Quellen stammen und meist auch mit unterschiedlichen Datentypen definiert wurden.

Durch die Optimierung der Datenstruktur im Unternehmen werden zugleich auch Vorgaben über die Struktur der Daten zentral in einem globalen Fachdatenmodell festgelegt. Diese Vorgaben können dann in den vorherrschenden Systemen integriert werden.

Die Streaming-Data-Architektur muss aber nicht zwangsläufig die aktuell vorherrschende IT-Architektur ablösen, sondern kann die bestehende Architektur gerade im Umgang mit Streaming Data sehr gut ergänzen. Besonders für Unternehmen, die seit Jahren auf eine zentrale Unternehmens-IT setzen, ist dies eine sehr gangbare Lösung. Gerade hierbei ist es wichtig, dass die Datenstruktur bekannt und zentral geregelt ist, da die Datenhaltung hierfür redundant erfolgt und somit beide Systeme unabhängig voneinander entwickelt und gewartet werden können. Ein Austausch der Daten ist dadurch wesentlich konfliktärmer und weniger komplex.

Durch diese Streaming Data Platform beziehungsweise Architecture ist es nun möglich, nicht nur den Lebenszyklus eines Produktes innerhalb eines Unternehmens zu überwachen. Durch die immer stärkere Vernetzung und die Konnektivität zum Kunden über zusätzliche Services (z. B. Apps fürs Smarthome) kann ein Unternehmen den kompletten Lebenszyklus des Produktes außerhalb des Shopfloors verfolgen. Eine Auswertung der vom Kunden gesendeten Daten (Nutzungsdaten, Fehlermeldungen usw.) hat zudem das Potenzial, sogar Einfluss auf die aktuelle Produktion zu nehmen.

Nicht nur für den Kunden entstehen hierdurch neue Business Values, sondern auch für das Unternehmen selbst und sogar für dessen Partner (Drittanbieter, Zulieferer). Dafür müssen die benötigten Systeme miteinander verbunden werden.

Fazit

Modellgetriebene Analyse und Design ist gerade heute im Hinblick auf die neuen Geschäftsmodelle und Anforderungen (vgl. [Kau15]), welche die Industrie 4.0 mit sich bringt, immens wichtig. Die Herausforderung hierbei besteht darin, all die Daten, die in einem Unternehmen anfallen, zu managen.

Und genau dabei können Modelle einen wichtigen Beitrag leisten. Durch das Aufzeigen der Zusammenhänge zwischen den Geschäftsprozessen und den Fachklassen bis hin zu den einzelnen Datenobjekten können die Auswirkungen und Zusammenhänge neuer Anforderungen auf eine Anwendung schnell und effizient analysiert werden. Modellierte Prozesse, Zustände und Entscheidungsstrukturen können, bevor diese zum Einsatz kommen, direkt im Modell geprüft werden – ein wesentlicher Faktor bei der Qualitätssicherung. Und zu guter Letzt unterstützen Modelle auch die Kommunikation zwischen Technik und Fachbereich und sorgen unternehmensweit für ein einheitliches Verständnis. Die Komplexität innerhalb eines Unternehmens wird beherrschbarer.

Literatur & Links

[ApKa] Apache KafkaTM, siehe: https://kafka.apache.org/

[DpDh17] Deutsche Post AG, 6.9.2017, siehe: http://www.dpdhl.com/de/presse/pressemitteilungen/2017/dhl_und_huawei_beschleunigen_inbound-to-manufacturing-logistik_mit_iot-loesung.html

[ItWi16]http://www.itwissen.info/IIoT-industrial-Internet-of-things-Industrie-IoT.html

[Kau15] T. Kaufmann, Geschäftsmodelle in Industrie 4.0 und dem Internet der Dinge, Springer Vieweg, 2015

[Lamb] N. Bijnens, M. Hausenblas, Lambda Architecture, 2017, siehe: http://lambda-architecture.net/

[Wam16] D. Wampler, Fast Data Architectures For Streaming Applications, O'Reilly, 2016

[Wikia]https://en.wikipedia.or/wiki/Event_Storming

[Wikib]https://de.wikipedia.org/wiki/Shopfloor_Management

Johannes Mickel

beschäftigt sich seit 2011 mit Softwarearchitekturen für die Industrie und im öffentlichen Dienst. Seit 2015 ist er für die MID GmbH als Senior Consultant für den Bereich Softwarearchitektur und Requirements Engineering tätig.

E-Mail: j.mickel(at)mid.de

Bildnachweise:

MID GmbH