Agiles Service-Engineering für Industrie 4.0 Erster Schritt: Anforderungsanalyse mit Anwendungsfällen

Conference Paper · June 2016

CITATIONS
0

READS
299

1 author:

Thomas Usländer
Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IOSB
98 PUBLICATIONS **507** CITATIONS

Agiles Service-Engineering für Industrie 4.0

Erster Schritt: Anforderungsanalyse mit Anwendungsfällen

Dr.-Ing. **Thomas Usländer**, Fraunhofer IOSB, Karlsruhe

Kurzfassung

Das Ziel des Beitrags ist es, die Bedeutung eine systematischen agilen Software-Engineerings speziell für Industrie 4.0 zu verdeutlichen. Beim Engineering von Industrie 4.0 Software-Anwendungen müssen die funktionalen und nicht-funktionalen Anforderungen von IT-Anwendern aus unterschiedlichen Disziplinen mit den Fähigkeiten einer sich entwickelnden Industrie 4.0 Dienstplattform gemäß RAMI4.0 in Einklang gebracht werden. Diese Plattformfähigkeiten werden von Software-Entwicklern und Architekten beschrieben und passen begrifflich und formal zumeist nicht mit der Sprache der Anwender zusammen. Der Beitrag beschreibt, wie diese "Lücke" mithilfe der agilen Software-Entwicklungsmethodik SERVUS geschlossen werden kann. Grundlagen der Methodik sind „user stories“ und davon abgeleitete semi-strukturierte Anwendungsfälle (use cases). SERVUS erlaubt deren Erfassung, Ablage und Vernetzung in einer Web-basierten Kollaborationsumgebung. Dort können sie innerhalb einer Fach-Community strukturiert, miteinander verlinkt, recherchiert, Schritt für Schritt verfeinert und dann später auf generische Funktionsbausteine abgebildet werden. Bei der Industrie 4.0 werden solche generischen Funktionsbausteine zukünftig als Basis- und höherwertige Plattform- und Anwendungsdienste in Industrie 4.0 Referenzarchitekturen definiert.

Abstract

This paper highlights the importance of systematic agile service engineering for Industrial Internet and Industrie 4.0 software applications. Here, the functional and non-functional requirements of IT users (mostly of engineering disciplines) need to be mapped to the capabilities of emerging service platforms. The capabilities of service platforms are usually described, structured and formalized by software architects. However, very often, their description does not fit to the language of the user. This complicates the transition from requirements analysis to system design. The paper describes how this 'gap' may be closed with help of a service-oriented analysis and design (SOAD) methodology entitled SERVUS and a corresponding Web-based collaborative tool that supports the documentation according to

the SERVUS design methodology. SERVUS denotes a Design Methodology for Information Systems based upon Service-oriented Architectures and the Modelling of Use Cases and Capabilities as Resources. It describes individual design activities interconnected by a common modelling environment combining several viewpoints of architectural reference model, e.g. the RAMI4.0 Business, Information and Functional Layers. SERVUS was successfully used in numerous collaborative and inter-disciplinary software projects.

1. Motivation

Beim Engineering von Industrie 4.0 Software-Anwendungen müssen die funktionalen und nicht-funktionalen Anforderungen von IT-Anwendern (zumeist aus den Disziplinen des Maschinenbaus und/oder der Elektro- und Automatisierungstechnik bzw. des Fabrik- und Anlagenbaus) mit den Fähigkeiten einer sich entwickelnden Industrie 4.0 Service Plattform gemäß einer Referenzarchitektur in Einklang gebracht werden. Dabei ist zu beachten, dass es mehrere Industrie 4.0 Servicereferenzarchitekturen geben wird [1], je nach Position der Problemstellung im Betrachtungsraum des Referenzarchitekturmodells Industrie 4.0 (RAMI4.0) [8], das als Orientierung und Meta-Modell für Industrie 4.0 Systemkonzepte dient.
Die Fähigkeiten einer Industrie 4.0 Service Plattform werden derzeit von Software-Entwicklern und Architekten beschrieben, strukturiert und formalisiert. Es ist abzusehen, dass diese Fähigkeitsbeschreibungen begrifflich und formal zumeist nicht mit der Sprache der Anwender zusammen passen, was den Übergang von der Anforderungsanalyse zum Systementwurf erschwert. Der vorliegende Beitrag beschreibt, wie diese "Lücke" mithilfe der an die Industrie 4.0 angepassten agilen Software-Entwicklungsmethodik SERVUS geschlossen werden kann. Grundlagen der Methodik sind „user stories“ und davon abgeleitete semi-strukturierte Anwendungsfälle (use cases). SERVUS steht für eine „Service-oriented Analysis and Design of Requirements based upon Use Cases and Information Resources“ [2] und wurde schon in zahlreichen kollaborativen und interdisziplinären Software-Entwicklungsprojekten erfolgreich eingesetzt [4], hauptsächlich in Projekten, die den Prinzipien einer serviceorientierten Architektur (SOA) folgen.

2. Stand der Technik

2.1 Service-orientierte Analyse und Design

Es wurden in den letzten Jahren zahlreiche Methoden für die Service-orientierte Analyse und Design (SOAD) in der Literatur vorgeschlagen. Umfangreiche Übersichten und Bewertungen finden sich in [10] und [11]. Man kann sie in zwei wesentliche Klassen einteilen [12]:

1. MDA-to-SOA: Erweiterung und/oder Anpassung der modellgetriebenen Architektur (model-driven architecture MDA) auf service-orientierte Umgebungen. Beispiel dafür ist die IBM Methodology for Service-oriented Modelling and Architecture (SOMA) [13].
2. BPM-to-SOA: Geschäftsprozessorientierte Methoden (business process modelling BPM), die mit der Modellierung von Geschäftsprozessen in dedizierten BPM-Sprachen (z.B. OASIS Business Process Modeling Notation BPMN oder Business Process Execution Language BPEL) beginnen und dann auf die Service-Ebene abbilden.

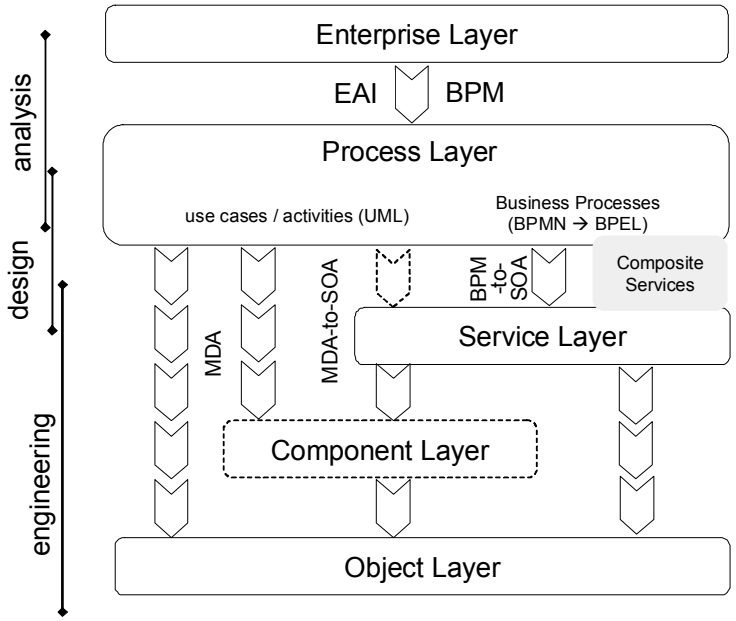

Bild 1: Grundlegende Kategorisierung von SOAD Methoden

Die hauptsächlichen Defizite der bestehenden Methoden betreffen die folgenden Punkte:

1. Der Anwender muss seine Anforderungen in formalen Sprachen der Systemarchitekten formulieren, die seiner Welt fremd sind. Dadurch leidet die Akzeptanz und die Nachvollziehbarkeit (traceability [15]) und es entstehen leicht Inkonsistenzen zwischen den Anforderungen und dem Systemmodell.
2. Es besteht zumeist die grundsätzliche Annahme, dass ein System frei von bestehenden Fähigkeiten einer Service-Plattform entwickelt werden kann.
3. Es werden keine Randbedingungen von bestehenden oder entstehenden Referenzarchitekturmodellen beachtet, was gerade bei Anwendungen des Industriellen Internet of Things (IIoT wie z.B. Industrie 4.0) sehr wichtig ist.

Der vorliegende Beitrag fokussiert auf die Behandlung des ersten Defizits, d.h. es wird die Frage behandelt, wie Anwenderanforderungen so formuliert werden können, dass sie verständlich und nachvollziehbar analysiert und dokumentiert werden können.

2.3 Standpunkte und Sichten auf eine Systemarchitektur

Als Grundstruktur für die übersichtliche Darstellung der unterschiedlichen Aspekte der Entwicklung einer Systemarchitektur werden zumeist Referenzmodelle verwendet. Als Standardbeispiel soll das ISO-Referenzmodell für die verteilte, offene Informationsverarbeitung (ISO RM-ODP [5]) betrachtet werden. Entsprechend dem ISO RM-ODP wird die verteilte System- und Software-Architektur von fünf unterschiedlichen Standpunkten (engl. viewpoint) aus in Augenschein genommen. Die Standpunkte bilden allerdings nur einen strukturellen Rahmen, der für eine konkrete Architekturbeschreibung passend interpretiert werden muss. Für eine auf ISO RM-ODP basierende Architekturbeschreibung wird folgende Interpretation[1] angewandt:

- Anwenderstandpunkt (enterprise viewpoint): Sicht der behördlichen und ggf. externen Anwender, Randbedingungen in Form von Anwendungsfällen als Ergebnis der Anforderungsanalyse.
- Informationsstandpunkt (information viewpoint): Hauptsächliche Informationsobjekte und deren Verknüpfungen auf konzeptioneller Ebene in Form eines fachlichen Informationsmodells.
- Dienstestandpunkt (computational viewpoint[2]): Grundfunktionen und deren Zusammenwirken auf konzeptioneller Ebene in Form von Funktionen und Diensten.
- Technologischer Standpunkt (technology viewpoint): Technologische Grundlagen und verwendete Standards.
- Engineering-Standpunkt (engineering viewpoint): Technische Realisierung (engineering) im Sinne einer Implementierungsarchitektur, d. h. die Abbildung der konzeptionellen Ebene auf die verwendete Technologie.

Der grundsätzliche Vorschlag von ISO RM-ODP, Architekturbeschreibungen nach verschiedenen Standpunkten zu strukturieren, hat sich allgemein durchgesetzt, auch wenn die einzelnen Standpunkte unterschiedliche geschnitten und benannt sind. Für Industrie 4.0 oder allgemein Anwendungen des industriellen Internet sind die Strukturierungsvorschläge des

[1] Der ISO Standard RM-ODP verwendet die folgenden englischen Begriffe: Enterprise Viewpoint, Information Viewpoint, Computational Viewpoint, Technology Viewpoint und Engineering Viewpoint.

[2] Oftmals, insbesondere in SOA Umgebungen, auch als Service Viewpoint interpretiert, vgl. [2]

Referenzarchitekturmodells Industrie 4.0 (RAMI4.0) [8] und des Referenzarchitektur (IIRA) des Industrial Internet Consortium (IIC) [9] relevant (vgl. Bild 2).

Das RAMI4.0 verwendet folgende Unterteilung[3] mit den beschriebenen Beschreibungsaspekten:

- Geschäftsschicht (Business Layer): u.a. Abbildung der Geschäftsmodelle und der sich daraus ergebenden Gesamtprozesse
- Funktionsschicht (Functional Layer): u.a. Laufzeit- und Modellierungsumgebung (Plattform) für Funktionen (Dienste) zur Unterstützung von Geschäftsprozessen
- Informationsschicht (Information Layer): u.a. Laufzeitumgebung für die Ereignis(vor)-verarbeitung, Gewinnung und konsistente Integration verschiedener Daten, Bereitstellung strukturierter Daten über Dienstschnittstellen
- Kommunikationsschicht (Communication Layer): u.a. Vereinheitlichung der Kommunikation unter Verwendung eines einheitlichen Datenformats
- Integrationsschicht (Integration Layer): u.a. technische Integration und Bereitstellung der rechnergestützten Informationsverarbeitung der Gegenstandsschicht (assets).
- Gegenstandsschicht (Asset Layer): repräsentiert Aspekte der Realität, z. B. physikalische Elemente wie Linearachsen, Blechteile, Dokumente, aber auch Ideen. Sie beinhaltet auch den Menschen, der über die Integrationsschicht an die virtuelle Welt angebunden wird.

Die IIRA unterscheidet folgende Standpunkte:

- Business Viewpoint: u.a. Identifikation der Interessensvertreter (stakeholder) mit ihren Geschäftsvisionen, Werten und Zielen
- Usage Viewpoint: u.a. Aspekte der Systembenutzung in Form von Aktivitätssequenzen
- Functional Viewpoint: u.a. funktionale Komponenten und ihre Beziehungen, Interaktionen und Schnittstellen untereinander und gegenüber externen Systemkomponenten zur Unterstützung der Systemaktivitäten (vgl. usage viewpoint).
- Implementation Viewpoint: betrachtet u.a. die technologischen Aspekte zur Implementierung der funktionalen Komponenten, ihrer Kommunikation und ihres Lebenszyklusmanagements.

Durch verschiedene IIoT/Industrie4.0-Projekte und die angekündigte Zusammenarbeit zwischen Industrie 4.0 und dem IIC ist zu erwarten, dass eine Abbildung der IIRA Standpunkte auf die RAMI4.0 Architekturaspekte genauer beschrieben werden wird.

[3] In der RAMI4.0 Spezifikation als „Layer“ und „Schicht“ bezeichnet.

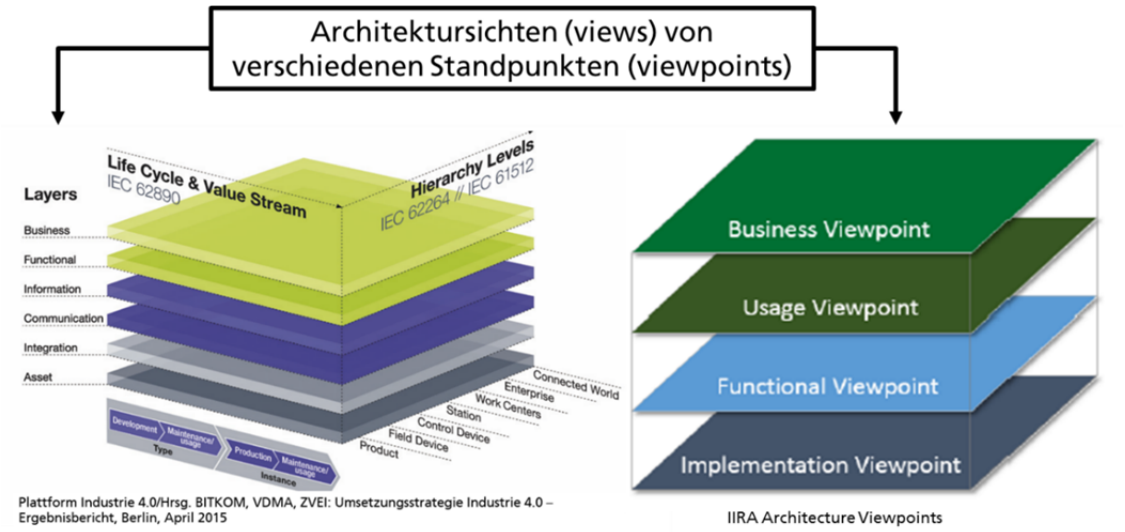

Bild 2: RAMI4.0 und IIRA Architektursichten

Der vorliegende Beitrag beschäftigt sich mit der Frage, wie die fachlichen Anforderungen aus Sicht eines Fachexperten mit den Fähigkeiten einer Plattform in Form von Funktionen (Diensten) und (Informations-)Ressourcen aus Sicht eines IT-Experten in Einklang gebracht werden können. Es geht also um die Frage, wie die Anforderungsanalyse und der abstrakte, d.h. technologieunabhängige Systementwurf agil miteinander verschränkt werden können (vgl. Bild 3). Bezogen auf die beschriebenen Referenzmodelle beinhaltet dies die folgenden Architekturstandpunkte:

- ISO RM-ODP: Enterprise, Information und Computational (Service) Viewpoint
- RAMI4.0: Business, Functional und Information Layer
- IIC IIRA: Business, Usage und Functional Viewpoint

3. Die SERVUS Methodik

Grundlage der SERVUS Methodik sind semi-strukturierte Anwendungsfälle (use cases). Diese können in einer Web-basierten SERVUS Kollaborationsumgebung strukturiert abgelegt, recherchiert, Schritt für Schritt verfeinert und auf generische Funktionsbausteine abgebildet werden (vgl. Bild 3). Bei der Industrie 4.0 werden solche generischen Funktionsbausteine als höherwertige Plattformdienste in einer zukünftigen Industrie 4.0 Referenzarchitektur zur Verfügung stehen. Beispiele hierfür sind Verzeichnisdienste, Suchdienste, Verwaltungsdienste für I40-Komponenten [8] und an der jeweiligen Wertschöpfungskette orientierte Planungs- und Betriebsunterstützungsdienste. Dazu kommen allgemeine generische Basisdienste zum Lesen und Schreiben von Informationen gemäß unterschiedlichen Architekturstilen und Dienstemustern (z.B. request/reply oder Representational State Transfer REST [6]).

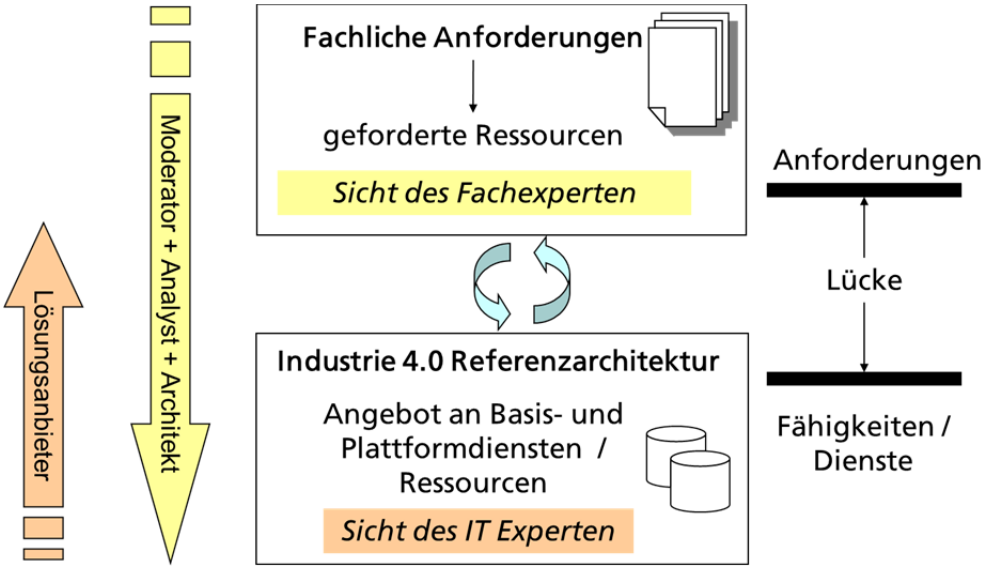

Bild 3: Agile, serviceorientierte Anforderungsanalyse

Um eine Industrie 4.0-Kompatibilität zu erreichen müssen diese Dienste beim Software-Engineering von vornherein bei der Architekturfragen systematisch berücksichtigt werden. Eine besondere Bedeutung liegt hierbei auf der Beachtung nicht-funktionaler Anforderungen wie z.B. IT-Sicherheit und Echtzeit.

3.1 Ablauf bei der Erfassung von Anforderungen

Unabhängig von den oben beschriebenen Standpunkten betrachtet die SERVUS-Methode folgende Tätigkeiten für die agile, serviceorientierte Anforderungsanalyse (vgl. Bild5):

- Planung des Projekts inkl. Erzeugung eines Projektrahmens zur Dokumentation der Anwendungsfälle
- Formalisierte textuelle Beschreibung der Anwendungsfälle (semi-formal)
- Nutzung eines grafischen Modellierungswerkzeugs (z.B. UML Enterprise Architect), um die verschiedenen Anwendungsfälle zu modellieren, dokumentieren, strukturieren und analysieren (formale Beschreibung)
- Besprechung der Modelle mit den Nutzern
- Identifikation der geforderten (Informations-)Ressourcen und deren Basisoperationen (lesen/schreiben/erzeugen/löschen)
- Entwicklung einer Ontologie zur Begriffsklärung und des Informationsmodells
- Erfassung der nicht-funktionalen Anforderungen
- Erarbeitung des Dienstemodells

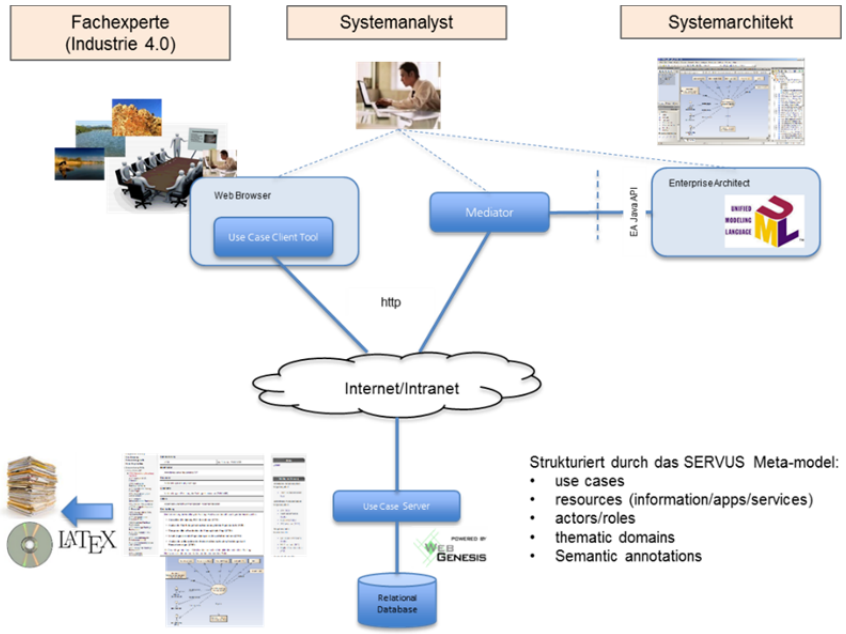

Bild 4: SERVUS Werkzeug zur Erfassung und Verwaltung von Anwendungsfällen

3.3 Erfassung von Anwendungsfällen

Zunächst werden in enger Abstimmung mit den Anwendern Beschreibungen von Anwendungsfällen erstellt, ggf. auch über den Zwischenschritt von rein textuell formulierten kurzen „Anwendererzählungen“ (user stories). Der Systemanalyst überführt diese in eine semiformale, tabellenorientierte Kurzbeschreibung sowie ggf. eine formale grafische Beschreibung unter Einbezug der geforderten Informationsobjekte, um nach Rückkopplung mit den Anwendern zu einem einheitlichen gemeinsamen Verständnis zu kommen. Der Einsatz eines Software-Werkzeugs führt zur Vereinheitlichung über die verschiedenen Anwendungsfälle hinweg zur Eliminierung von Inkonsistenzen sowie zu einer einheitlichen Modellierungstiefe.
Die Beschreibung der Anwendungsfälle als Basis für eine formale Modellierung wurde durch die Vorgabe einer einheitlichen Schablone vorstrukturiert. Dennoch sind Fälle hinsichtlich der Akteure, der Verarbeitungsschritte (Geschäftsprozessketten) sowie der Informationsobjekte, die verwendet werden oder neu entstehen, häufig nicht ausreichend bestimmt. Mit der formalen Modellierung wird eine präzisere informationstechnische Beschreibung erreicht, die jedoch allgemein verständlich bleibt, so dass deren Richtigkeit und Vollständigkeit mit den betroffenen Benutzern abgestimmt werden kann.

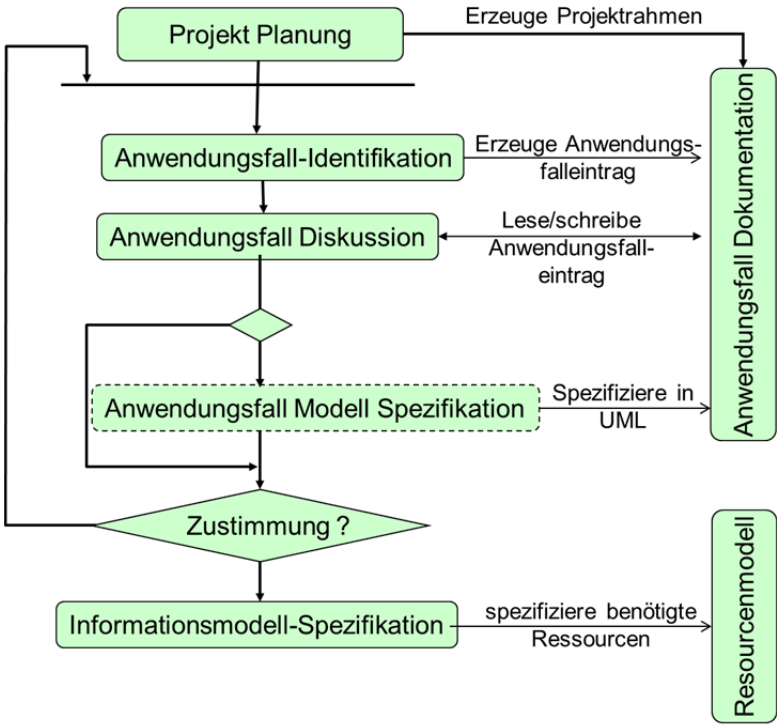

Bild 5: SERVUS-Vorgehensweise zur Anforderungsanalyse

Für die formale Modellierung wird die sogenannte Unified Modeling Language (UML) eingesetzt. In dieser Methode stehen die Informationsobjekte (Fachobjekte, Ergebnisobjekte), deren Attribute sowie die Beziehungen zwischen ihnen im Mittelpunkt. Fachobjekte sind eindeutig identifizierbare Informationsressourcen mit einer eingeschränkten Zahl an unterstützten Operationen. Sie können entweder gelesen, verändert, erzeugt oder gelöscht werden, d.h. sie folgen den Prinzipien des REST-Architekturstils [6]. Bezogen auf das RAMI4.0 ergibt sich hier eine interessante Abbildung auf I4.0 Komponenten, die damit als Kernartefakte einer Industrie 4.0 Service-Plattform sehr früh in die Anforderungsanalyse eingebunden werden können. Ein zweiter Schwerpunkt liegt darin, Anlässe, Bearbeiter und Anwendungsfälle (Geschäftsvorgänge oder Dienste) möglichst einheitlich zu strukturieren. Daher wird für jeden Anwendungsfall untersucht:

- Wer arbeitet mit dem Anwendungsfall? → Festlegung der Benutzerrollen
- Welches Ergebnis soll erreicht werden? → Festlegung der Ergebnisobjekte
- Wie soll dieses erreicht werden? → Festlegung der wesentlichen Arbeitsschritte
- Was wird dazu benötigt? → Festlegung der notwendigen Informationsobjekte

Die Anwendungsfälle können grafisch beschrieben werden (Bild 6) in einem Diagramm mit drei übereinander liegenden Ebenen:

- Der Anlass (trigger) (obere Ebene) ist das auslösende Ereignis für die Bearbeitung eines Anwendungsfalls; z.B. ein eingehender Auftrag, ein Störfall oder ein Termin.
- Der Anwendungsfall (mittlere Ebene) steht im Zentrum der Darstellung. Er beschreibt einen Geschäftsprozess oder einen Software-Dienst, der es ermöglicht, Teile des Vorgangs automatisiert oder gar vollautomatisch ablaufen zu lassen. Der Anwendungsfall wird von einem oder mehreren Bearbeitern bearbeitet.
- Der Bearbeiter (mittlere Ebene) bearbeitet einen Anwendungsfall, d.h., er führt den Geschäftsprozess durch.
- Fachobjekte (obere Ebene) werden zur Bearbeitung eines Anwendungsfalls benötigt und beschreiben als Informationsobjekte die Objekte der virtuellen oder realen Welt (Asset-Typen oder Instanzen), etwa Produktionsanlagen oder Pläne. Insbesondere können neue oder geänderte Fachobjekte Ergebnisse des Anwendungsfalls sein.
- Ergebnisobjekte (untere Ebene) werden bei der Bearbeitung des Anwendungsfalls erzeugt. Ergebnisse können neue Informationsobjekte sein.

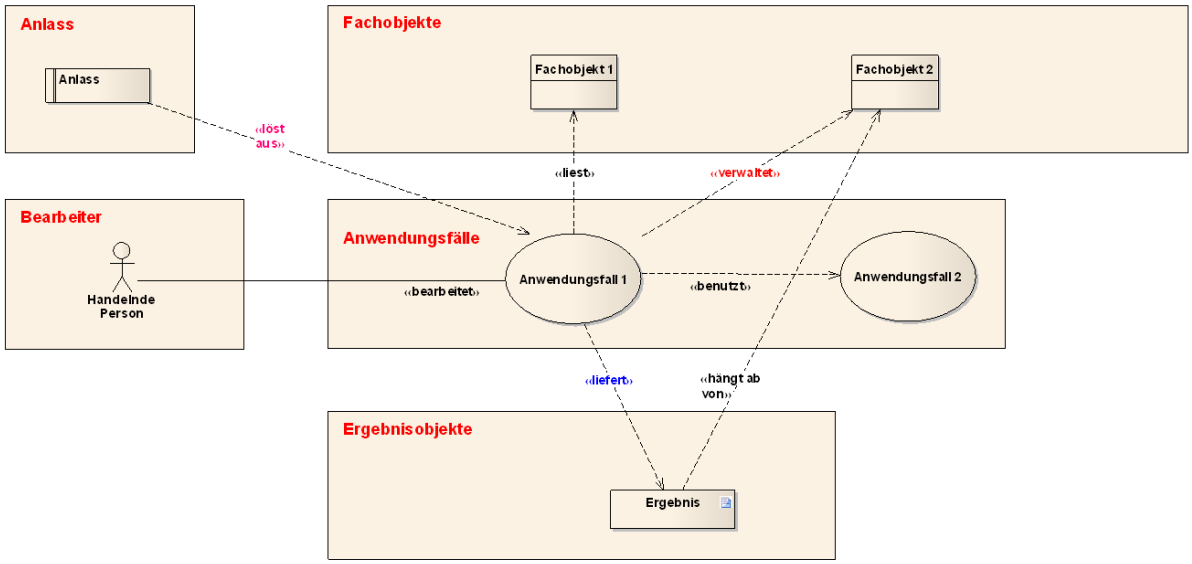

Bild 6: SERVUS Beschreibungsmethode für Anwendungsfälle

Zwischen den Objekten können u.a. die folgenden Beziehungen bestehen:

- Als allgemeine Bezeichnung wird „hängt zusammen mit“ verwendet mit Ausprägungen wie „löst aus“, „hängt ab von“, „hat Auswirkung auf“ u.a.
- Beziehungen zwischen Bearbeiter und Anwendungsfall (Beziehung „bearbeitet“)
- Beziehungen zwischen Anwendungsfällen: Die Beziehung „benutzt“ beschreibt die Verwendung einer gekapselten Teilfunktionalität. Dabei kann es sich um einen ande-

ren Anwendungsfall handeln oder um einen neu zu entwickelnden generischen Anwendungsfall wie „Intelligente Suche“, der als Dienst realisiert wird.

- Beziehung zwischen Anwendungsfall und Fachobjekt:
  Sie soll allgemein mit „verarbeitet“ bezeichnet werden. In der Regel wird aber mit Präzisierungen gearbeitet, dies sind u. a. „liest“, „verwaltet“ oder auch „verwendet“.
- Beziehung zwischen Anwendungsfall und Ergebnisobjekt (Beziehung „liefert“).

4. Zusammenfassung und Ausblick

Die SERVUS-Methodik hat sich mit ihrer kollaborativen und semi-formalen Struktur in zahlreichen Projekten bewährt. Sie wird vor allem immer dann erfolgreich eingesetzt, wenn nicht „auf der grünen Wiese“ neue Software-Anwendungen entworfen werden, sondern auf eine generische Software-Plattform mit bereits spezifizierten und/oder implementierten Funktionen oder Diensten aufgesetzt werden soll als Teil der Systemanforderungen. Dies ist auch bei Industrie 4.0 und anderen Anwendungen des industriellen Internets der Fall sein, sobald sich die jeweilige Fach-Community auf eine servicebasierte Referenzarchitektur geeinigt hat. Um eine Industrie 4.0 -Kompatibilität zu erreichen, müssen diese Dienste von vornherein systematisch beim Architekturentwurf und beim Software-Engineering berücksichtigt werden. Deshalb werden sie als Anforderungen an Systemfähigkeiten ebenfalls in SERVUS erfasst. Use cases können dann im Verlauf eines Systementwurfs darauf abgebildet werden mit bilateraler Rückverfolgbarkeit (traceability). Die SERVUS-Methodik wurde u.a. auch im BITKOM Arbeitskreis Industrie 4.0 Interoperabilität verwendet[4] und steht der Industrie 4.0 Community als (mobile) Web-Anwendung zur strukturierten Ablage von Industrie 4.0 Use cases zur Verfügung. Sie ist strukturell kompatibel zur DKE Use Case Datenbank, die auch in der AG1 der Plattform Industrie 4.0 verwendet wird, und folgt den Empfehlungen der Deutschen Normungs-Roadmap I4.0 V2.

Aktuelle Arbeiten beziehen sich auf die systematische, strukturierte und durchgängige Behandlung von nicht-funktionalen Anforderungen wie z.B. IT-Sicherheit und Verlässlichkeit und deren Abbildung of Quality of Service (QoS)-Eigenschaften von Serviceplattformen sowie die iterative Abbildung auf Fähigkeiten von Service Plattformen gemäß den Referenzarchitekturmodellen IIRA und RAMI4.0 [14]. Eine entscheidende Rolle spielt hierbei die Interpretation der Fachobjekte (Informationsressourcen) als geforderte und angebotene Industrie 4.0-Komponenten einer Service-Plattform.

[4] vgl. Fraunhofer IOSB use case Server unter http://i40-usecases.iosb.fraunhofer.de .

5. Referenzen

[1] Usländer, T. (Ed.): Industrie 4.0 - Auf dem Weg zu einem Referenzmodell. VDI Statusreport. VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik, Düsseldorf, April 2014, http://www.vdi.de/industrie40

[2] Usländer, T.: Service-oriented Design of Environmental Information Systems. PhD thesis of the Karlsruhe Institute of Technology (KIT), KIT Scientific Publishing. ISBN 978-3-86644-499-7, 2010.

[3] Usländer, T. und Epple, U.: Reference Model of Industrie 4.0 Service Architectures – Basic Concepts and Approach. at – Automatisierungstechnik Special Issue: Industrie 4.0 (Ed.: Beyerer/Jasperneite/Sauer), Band 63, Heft 10, Okt 2015.

[4] Usländer, T., Batz, T.: How to Analyse User Requirements for Service-Oriented Environmental Information Systems. ISESS 2011 Proceedings, IFIP AICT, vol. 359, S. 165-172, Springer Heidelberg, 2011.

[5] ISO/IEC10746 Reference Model of Open Distributed Processing (RM-ODP),

[6] Fielding, R.T.: Architectural Styles and the Design of Network-Based Software Architectures. Doctoral dissertation, University of California, Irvine, 2000.

[7] Jacobson, I. and Ng, P.-W. Aspect-Oriented Software Development with Use Cases. The Addison-Wesley Object Technology Series, ISBN 0-321-26888-1, 2005.

[8] DIN SPEC 91345: Referenzarchitekturmodell Industrie 4.0 (RAMI4.0), 2016

[9] Industrial Internet Consortium (IIC): The Industrial Internet Reference Architecture Technical Report. 2016. http://www.iiconsortium.org/IIRA.htm .

[10] Ricken, J., Petit, M.: Characterization of Methods for Process-Oriented Engineering of SOA. BPM 2008, ISBN 978-3-642-00327-1, pp. 621-632, 2008.

[11] Kohlborn, T., Korthaus, A., Chan, T. and Rosemann, M.: Service Analysis - A Critical Assessment of the State of the Art. ECIS 2009, Verona, Italy, 2009.

[12] Bieberstein, N., Bose, S., Fiammante, M., Jones, K. and Shah, R.: Service-Oriented Architecture (SOA) Compass – Business Value, Planning and Enterprise Roadmap. IBM Press developerWorks® Series. ISBN 0-13-187002-5, 2006.

[13] Arsanjani, A. et al.: SOMA: A method for developing service-oriented solutions. IBM Systems Journal, Vol. 47, No. 3, pp. 377-396, 2008.

[14] Usländer, T.: Agile Service-oriented Analysis and Design for Industrial Internet Applications. Proceedings of the CIRP-CMS 2016, Stuttgart, 2016.

[15] Favaro, J. et al.: Next Generation Requirements Engineering. DOI: 10.1002/j.2334-5837.2012.tb01349.x, 2014.