SoaML-basierter Entwurf eines dienstorientierten Überwachungssystems

Michael Gebhart[1], Jürgen Moßgraber[2], Thomas Usländer[2], Sebastian Abeck[1]

[1] Cooperation & Management, Karlsruher Institut für Technologie (KIT)
[2] Fraunhofer Institut für Optronik, Systemtechnik und Bildauswertung (IOSB)

Abstract: Von einem Überwachungssystem zur Begleitung von Personen innerhalb von Gebäuden wird häufig gefordert, sich in die bestehende Anwendungslandschaft eines Unternehmens integrieren zu lassen und vorhandene Anwendungsfunktionalität möglichst wiederzuverwenden. Anwendungslandschaften werden verstärkt in Form dienstorientierter Architekturen strukturiert, weshalb ein integrierbares Überwachungssystem ebenfalls dienstorientiert entworfen werden sollte. Mit der *Service oriented architecture Modeling Language* (SoaML) existiert ein Standard, der eine Modellierung eines dienstorientierten Überwachungssystems ermöglicht. In dieser Arbeit werden die hierfür notwendigen Elemente der SoaML identifiziert und in einen Entwicklungsprozess eingeordnet.

1 Einleitung

Soll die Anwendungslandschaft eines Unternehmens um eine neue Anwendung, wie beispielsweise ein Überwachungssystem zur Begleitung von Personen innerhalb von Gebäuden erweitert werden, ist eine zentrale Anforderung an diese Anwendung, sich angemessen in die bestehende Anwendungslandschaft integrieren zu lassen. Dies bedeutet, dass bereits bestehende Funktionalität in Form von Standardsoftware, wie beispielsweise ein Personenverzeichnis, seitens der neuen Anwendung genutzt und die Überwachungsfunktionalität geeignet anderen Anwendungen zur Verfügung gestellt wird.

Da Unternehmen ihre Informationstechnologie (IT) zunehmend dienstorientiert gestalten, müssen auch neu zu entwickelnde Anwendungen das Paradigma der Dienstorientierung berücksichtigen, um in eine dienstorientierte Architektur integriert werden zu können. Für ein neu zu entwickelndes Überwachungssystem bedeutet das, geeignete Dienste bereitzustellen und eventuell bereits vorhandene Dienste eines Personenverzeichnisses oder einer vorhandenen Gebäudeverwaltung zu nutzen.

Im Rahmen des Projektes NEST (*Network Enabled Surveillance and Tracking*) forscht das Fraunhofer IOSB an dem Ansatz eines automatischen und auftragsbasierten Überwachungssystems [Ba08, MRV10]. Nachfolgend wird gezeigt, wie die *Service oriented architecture Modeling Language* (SoaML) [Om09] zur Dienstidentifikation (Kapitel 2) und Dienstspezifikation (Kapitel 3) eines dienstorientierten Überwachungssystems gezielt eingesetzt werden kann.

2 Dienstidentifikation

Das Überwachungssystem soll folgendes Szenario unterstützen: Eine Person betritt das Gebäude. Die Person äußert am Empfang den Wunsch einen Mitarbeiter aufzusuchen. Der Empfangsmitarbeiter startet einen neuen Auftrag "Person zu Zielort leiten" im Überwachungssystem. Hierbei wählt er das Büro des Mitarbeiters aus. Das System ermittelt nun mit Hilfe des Personenverzeichnisses die Rolle der zu überwachenden Person. Ausgehend von der Rolle werden seitens der Gebäudeverwaltung die Bereiche ermittelt, welche die Person betreten darf. Nun überwacht das System den Weg der Person. Kommt die Person von ihrem Weg ab, wird der Empfangsmitarbeiter darauf hingewiesen, um über eine eventuelle Gefährdung entscheiden zu können. Hierbei bekommt er den Standort der Person angezeigt. Erreicht die Person ihren Zielort, wird der Überwachungsauftrag beendet.

Nach Beschreibung der Anforderungen werden – in Anlehnung an Erl [Er06] und den *Rational Unified Process for Service-Oriented Modeling and Architecture* (SOMA) [Ar04, Wa07] – vom IT-Architekten Dienstkandidaten und ihre Abhängigkeiten identifiziert. Dienstkandidaten repräsentieren Vorschläge für benötigte Dienste. Sie realisieren direkt oder indirekt einen Geschäftsdienst und beschreiben in Form von Operationskandidaten, welche Funktionalität diese Dienste bereitstellen sollen. Dienstkandidaten bilden somit eine Blaupause für die spätere Spezifikation. Die Beschreibung dieser Aspekte in SoaML ist in folgender Tabelle dargestellt:

Tabelle 1: SoaML-Elemente während der Identifikationsphase

Konzeptionelles Element	**SoaML-Element**
Dienstkandidat	Capability-Element
Dienstoperationskandidat	Operation eines Capability-Elements
Abhängigkeit zwischen Dienstkandidaten	Usage-Beziehung zwischen Dienstkandidaten
Bezug zu den geschäftlichen Anforderungen	Realisierung eines Geschäftsdienstes (UML-Anwendungsfalls), stereotypisiert mit "MotivationRealization"

Ausgehend von dieser Zuordnung können die Dienstkandidaten entsprechend mit SoaML erstellt werden. Die Dienstkandidaten, ihre Abhängigkeiten, die zugeordneten Dienstoperationskandidaten und der Bezug von Dienstkandidaten zu den geschäftlichen Anforderungen für das Überwachungssystem sind in Abbildung 2 dargestellt.

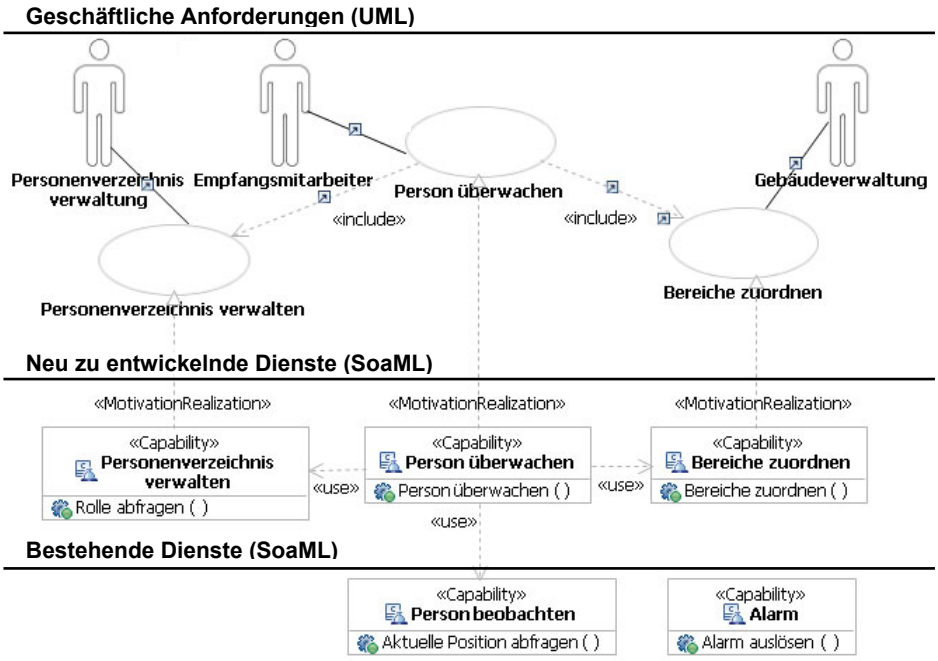

Abbildung 1: Identifizierte Dienstkandidaten und ihr Bezug zu Geschäftsdiensten

3 Dienstspezifikation

Im Anschluss an die Identifikation benötigter Dienste erfolgt ihre Spezifikation. Dies bedeutet, dass (i) die angebotene Dienstschnittstelle, (ii) die benötigten Dienste in Form von Dienstschnittstellen und (iii) die realisierende Dienstkomponente beschrieben werden. Sofern Spezifikationen für zu integrierende Dienste vorliegen, können und sollten diese genutzt werden.

3.1 Spezifikation von angebotenen und benötigten Dienstschnittstellen

Die Spezifikation einer Dienstschnittstelle beschreibt die angebotene technische Schnittstelle, die erforderliche technische Schnittstelle, das Interaktionsprotokoll und den Bezug zu den vorher identifizierten Dienstkandidaten. Die angebotene technische Schnittstelle definiert angebotene Operationen inklusive der Parametertypen während die benötigte technische Schnittstelle erforderliche Operationen auf der Seite des Dienstnehmers festlegt, um beispielsweise Callbacks zu empfangen. Tabelle 2 zeigt die SoaML-Elemente, die zur Beschreibung dieser Aspekte benötigt werden.

Tabelle 2: SoaML-Elemente zur Spezifikation von Dienstschnittstellen

Konzeptionelles Element	SoaML-Element
Dienstschnittstelle	ServiceInterface-Element
Angebotene technische Schnittstelle	Interface, dass durch das ServiceInterface-Element realisiert wird
Erforderliche technische Schnittstelle	Interface, dass durch eine Usage-Beziehung mit dem ServiceInterface-Element verbunden ist
Parametertypen	Ein MessageType, der ggf. andere Datentypen in Form von dataTypes beinhaltet
Interaktionsprotokoll	OwnedBehavior eines ServiceInterface-Elements, d.h. bspw. ein Aktivitätsdiagramm, das als OwnedBehavior und somit einer enthaltenen Verhaltensbeschreibung dem ServiceInterface-Element untergeordnet ist
Bezug zu Dienstkandidat	Eine Expose-Beziehung zum Capability-Element, welches den Dienstkandidaten repräsentiert

Die Spezifikation einer Dienstschnittstelle kann demnach in SoaML, wie in Abbildung 3 dargestellt, modelliert werden. Hierbei ist darauf zu achten, dass mit dem Wechsel von zunächst aus dem Geschäft abgeleiteten Dienstkandidaten zu konkreten Dienstspezifikationen auch ein Wechsel der Sprache stattfinden kann. In diesem Fall wird von zunächst deutschsprachigen Anforderungen und dementsprechend deutschsprachigen Dienstkandidaten auf technisch ausgerichtete, englische Bezeichnungen gewechselt.

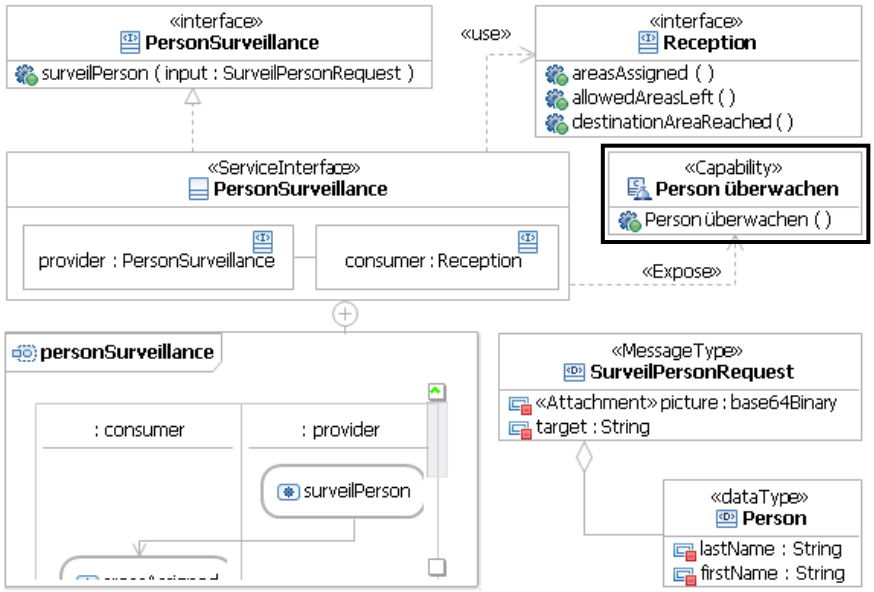

Abbildung 2: Spezifikation der Dienstschnittstelle "PersonSurveillance"

3.2 Spezifikation der Dienstkomponente

Die Dienstkomponente implementiert den Dienst, weshalb Details wie die interne Logik nur zu modellieren sind, wenn der Dienst eigenständig entwickelt und nicht ein bereits bestehender Dienst genutzt wird. Eine Dienstkomponente stellt ihre Fähigkeiten über eine Dienstschnittstelle bereit und benötigt ggf. selbst wieder andere Dienste zur Realisierung ihrer Funktionalität. In SoaML werden folgende Modellierungselemente genutzt:

Tabelle 3: SoaML-Elemente zur Spezifikation von Dienstkomponenten

Konzeptionelles Element	SoaML-Element
Dienstkomponente	Participant-Element
Angebotener Dienst	ServicePoint-Element, typisiert mit einem ServiceInterface-Element, welches den Dienst beschreibt
Benötigter Dienst	RequestPoint-Element, typisiert mit einem ServiceInterface-Element, welches den Dienst beschreibt
Interne Logik	OwnedBehavior eines Participant-Elements, d.h. bspw. ein Aktivitätsdiagramm für jede angebotene Operation, das als OwnedBehavior und somit einer enthaltenen Verhaltensbeschreibung dem Participant-Element untergeordnet ist. Zusätzlich kann ein Participant-Element selbst wieder aus Instanzen anderer Participant-Elemente bestehen und diese mittels ServiceChannel-Beziehungen zu einem komponierten System zusammensetzen.

Diese gezielte Auswahl an SoaML-Elementen ermöglicht es, mit bereits wenigen Sprachmitteln eine Spezifikation der Dienstkomponente mit klar definierter Semantik vorzunehmen. Abbildung 4 zeigt die beispielhafte Spezifikation einer Dienstkomponente in SoaML, die einen Dienst bereitstellt und drei weitere Dienste zur Erbringung ihrer Funktionalität nutzt.

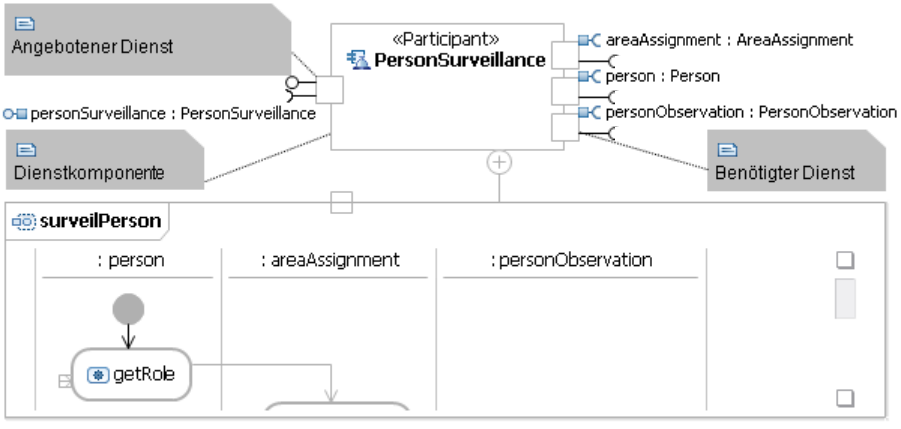

Abbildung 3: Spezifikation der Dienstkomponente "PersonSurveillance"

4 Zusammenfassung und Ausblick

In dieser Arbeit haben wir gezeigt, wie SoaML genutzt werden kann, um ein dienstorientiertes Überwachungssystem zu entwerfen. Bestehende Arbeiten wie Engels et al. [En08] oder Erl [Er06] fokussieren die Beschreibung notwendiger Schritte zur Entwicklung von Diensten, blenden jedoch die Nutzung konkreter Modellierungssprachen aus. SoaML [Om09] hingegen beschränkt sich auf die Beschreibung von Modellierungselementen. Mit der vorliegenden Arbeit wurden die Elemente aus SoaML in einen durchgängigen Entwicklungsprozess eingeordnet. Dadurch wird dem IT-Architekten eine Richtlinie gegeben, die vorgibt wie neue Anwendungen unterstützt durch Modellierungssprachen entwickelt werden können, damit sie sich in eine dienstorientierte Architektur integrieren lassen. Der Einsatz von SoaML als standardisierte Sprache ermöglicht die Nutzung von Modellierungswerkzeugen verschiedener Hersteller und kann aufgrund der zunehmenden Verbreitung als zukunftsweisend bewertet werden. Dabei besitzen erstellte Modelle aufgrund der Standardisierung und der Fokussierung auf dienstorientierte Architekturen eine definierte Semantik. Die bisher erfolgte Nutzung von UML zur Modellierung dienstorientierter Architekturen hingegen erfordert immer zunächst eine Interpretation der Modellierungselemente, die zwischen Architekten und auch zwischen Werkzeugen variieren kann. Zusätzlich bieten einige SoaML-fähige Werkzeuge bereits Transformationen, um Modelle, die auf SoaML basieren, in Quellcode zu überführen. So können u.a. automatisiert Schnittstellenbeschreibungen und ausführbare Prozesse basierend auf der *Web Services Description Language* (WSDL) und der *Business Process Execution Language* (BPEL) generiert werden.

Die Formalisierung von Dienstkandidaten, Dienstschnittstellen und Dienstkomponenten motiviert die Forderung nach einer Bewertung hinsichtlich von Diensteigenschaften wie beispielsweise loser Kopplung oder Autonomie. Eine auf den Ergebnissen des vorliegenden Beitrags aufsetzende Arbeit verfolgt das Ziel, formalisierte Artefakte auf Basis von SoaML hinsichtlich Diensteigenschaften zu bewerten, um Dienste bereits zur Entwurfszeit in Hinblick auf gewünschte Diensteigenschaften gestalten zu können.

Literaturverzeichnis

[Ar04] Arsanjani, A.: Service-Oriented Modeling and Architecture – How to identify, specify, and realize services for your SOA. IBM Developer Works, http://www.ibm.com/developerworks/library/ws-soa-design1, 2004.

[Ba08] Bauer, A. et al.: N.E.S.T. – Network Enabled Surveillance and Tracking. 2008.

[En08] Engels, G. et al.: Quasar Enterprise – Anwendungslandschaften serviceorientiert gestalten. dpunkt.verlag, 2008.

[Er06] Erl, T.: Service-Oriented Architecture – Concepts, Technology, and Design. Pearson Education, 2006.

[MRV10] Moßgraber, J.; Reinert, F.; Vagts, H.: An Architecture for a Task-Oriented Surveillance System. 2010.

[Om09] OMG: Service oriented architecture Modeling Language (SoaML) – Specification for the UML profile and metamodel for services (UPMS)". 2009.

[Wa07] Wahli, U. et al.: Building SOA Solutions Using the Rational SDP. http://www.ibm.com/redbooks, 2007.