Modellgetriebene Entwicklung von Sensorschnittstellen

Werkzeugunterstützung durch SensIDL

Nathalie Hipp, Hahn-Schickard
Dr. Christoph Rathfelder, Hahn-Schickard
Emre Taspolatoglu, FZI Forschungszentrum Informatik
Jörg Henß, FZI Forschungszentrum Informatik

SensIDL (http://www.sensidl.de) ist ein Open-Source-Werkzeug für die vereinfachte Implementierung von Kommunikationsschnittstellen intelligenter und ressourcen-eingeschränkter Sensorsysteme, welche als zentraler Baustein des „Internets der Dinge“ angesehen werden können. SensIDL (Sensor Interface Definition Language) unterstützt den Sensorentwickler mit einer dedizierten Schnittstellenbeschreibungssprache sowie zugehörigen Editoren und ermöglicht so die effiziente Definition und Dokumentation der Sensorschnittstellen. Ausgehend von dieser Beschreibung erzeugen Code-Generatoren automatisiert den auf Sensor- und Empfängerseite notwenigen Code. Diese semantisch angereicherte API ermöglicht die Daten zu instanziieren und danach zu übertragen. Hierdurch werden die Entwickler entlastet und die Effizienz der Entwicklung und Implementierung sowie deren Qualität kann gesteigert werden.

Die Integration von intelligenten Sensorsystemen in die IT-Landschaft und das „Internet der Dinge“ sowie die Kommunikation mit mobilen Endgeräten stellt für Entwickler immer eine aufwändige und fehlerträchtige Herausforderung dar. Es fehlen hier sowohl einheitliche Standards als auch effiziente Entwicklungswerkzeuge, die den Entwickler bei der Programmierung unterstützen und entlasten. Durch die sehr heterogene Mischung von Sensorsystemen in Bezug auf deren Rechenkapazität, Kommunikationsanbindung, Prozessorarchitekturen, Betriebssysteme und Programmiersprachen sind verschiedenste Kommunikationstechniken und Schnittstellen im Bereich der eingebetteten Systeme entstanden. Auf Grund des Mangels eines einheitlichen Standards sowie unterstützender Entwicklungswerkzeuge, sind Entwickler in vielen Fällen gezwungen die Kommunikation manuell als individuelle Lösungen selbst zu implementieren.

Anwendungsszenario

Für die reibungslose Kommunikation zwischen Sensorinformationen vom Sensorsystem zu einem mobilen Endgerät sind diverse Verarbeitungsschritte notwendig. Mit dem SensIDL-Werkzeug sollen sowohl die Sensorentwickler als auch die Entwickler der empfangenden Systeme in wiederkehrenden Aufgaben unterstützt werden indem der notwendige Implementierungscode automatisch erzeugt wird (siehe Abb. 1).

Der Sensorentwickler wird durch die entwickelte SensIDL-Sprache und die dazugehörigen Editoren dabei unterstützt, die Sensordaten bzw. -schnittstellen eindeutig auf einer inhaltlichen/semantischen Ebene zu beschreiben. Die notwendige Implementierung der Schnittstelle kann danach durch das Werkzeug aus dieser Beschreibung automatisiert generiert werden.

Der Software-Entwickler für das empfangende IT-System bekommt mit der vom Sensorentwickler erstellten Beschreibung eine formale und eindeutige Definition der

Schnittstelle. Der notwendige Code auf Empfängerseite kann ebenfalls mit Hilfe von SensIDL automatisch erzeugt werden. Hierdurch wird die Implementierung des Zugriffs auf Sensordaten und somit die Integration signifikant vereinfacht.

Modellgetriebener Ansatz

Der Ansatz der modellgetriebenen Software-Entwicklung, wie er auch in SensIDL angewendet wird, kombiniert die beiden technischen Konzepte der domänenspezifischen Sprachen (domain-specific languages DSLs) und der Code-Generierung mittels Modelltransformationen. Domänen-spezifische Sprachen bieten ein formales Konzept, um Strukturen, Abhängigkeiten, Verhaltensmodelle oder Anforderungen von Software-Anwendungen innerhalb definierter Domänengrenzen zu beschreiben. Dies schränkt die Anwendbarkeit im Vergleich zu allgemeinen Programmiersprachen wie Java oder C ein. Im Gegensatz dazu ermöglichen DSLs den Entwicklern sich auf die wichtigen Aspekte der Modellierung zu konzentrieren. Aus diesem Grund werden DSLs häufig dann eingesetzt, wenn die Komplexität reduziert bzw. gekapselt oder Technologien und Implementierungsdetails ausgeblendet werden sollen. DSLs besitzen immer ein formales Modell im Hintergrund. Dieses wird als Meta-Modell oder abstrakte Syntax bezeichnet, da es die Struktur und den Aufbau von Informationen vorgibt.

Code-Generatoren und Transformationsmechanismen sind das zentrale Element für die praktische Umsetzung der modellgetriebenen Software-Entwicklung. Sie erzeugen aus dem per DSL erstellten Modell automatisiert neue Elemente wie z.B. Quellcode. Diese Ergänzung mit Code-Generatoren führt zu einer signifikanten Steigerung der Entwicklungs- und Implementierungseffizienz.

Der SensIDL-Werkzeug

Dem modellgetriebenen Ansatz folgend wird in SensIDL eine eigene Beschreibungssprache mit speziellen Elementen für die Beschreibung von Sensorschnittstellen verwendet. Eine effiziente Implementierung der Kommunikationsschnittstellen smarter Sensoren unabhängig von der gewählten Kommunikationstechnologie wird daraus erzeugt.

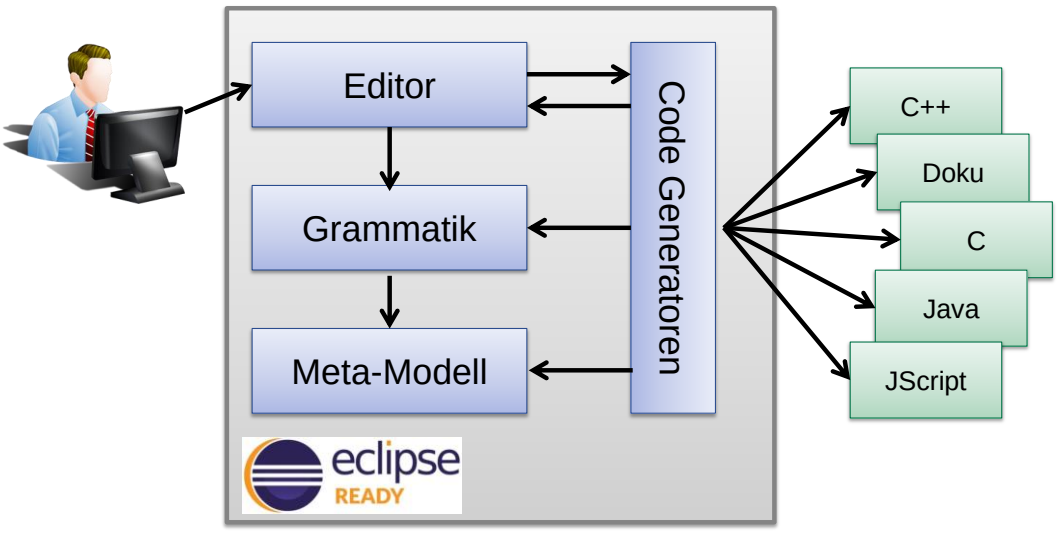

Abb. 1: Bausteine des SensIDL-Werkzeuges

Abb. 1 veranschaulicht den Aufbau des SensIDL-Werkzeuges. Die SensIDL Beschreibungssprache und der SensIDL Editor ermöglichen dem Nutzer, die von smarten Sensoren angebotene Schnittstellen und Daten einfach und intuitiv zu definieren. Die SensIDL Grammatik ist das Bindeglied zwischen dem Editor und dem zugrundliegenden Meta-Modell. Ausgehend vom SensIDL Meta-Modell und der durch den Benutzer erstellten Beschreibung können verschiedene Code-Generatoren ausgeführt werden. Diese Code-Generatoren können sowohl den notwenigen Implementierungscode für unterschiedliche Programmiersprachen als auch eine zugehörige Dokumentation erzeugen. Hierdurch wird unter anderem sichergestellt, dass die Dokumentation immer der Implementierung entspricht.

Das SensIDL Plug-In

Das SensIDL Werkzeug basiert auf der Eclipse Plattform und lässt sich deshalb auch als Eclipse-Plug-In und darauf aufbauender Embedded-Entwicklungsumgebungen, wie z.B. CodeComposer, installieren und dort sowohl für die Programmierung mit C und C++ als auch Java ohne Wechsel der Entwicklungsumgebung einsetzen. Es kommen hierbei verschiedene Technologien der Eclipse Plattform zum Einsatz. Für die Definition der SensIDL Sprache und der Editoren wird das Eclipse Modelling Framework (EMF) und das Xtext Projekt eingesetzt. Die Generatoren für Code und Dokumentation basieren auf der Generatorinfrastruktur von Xtend.

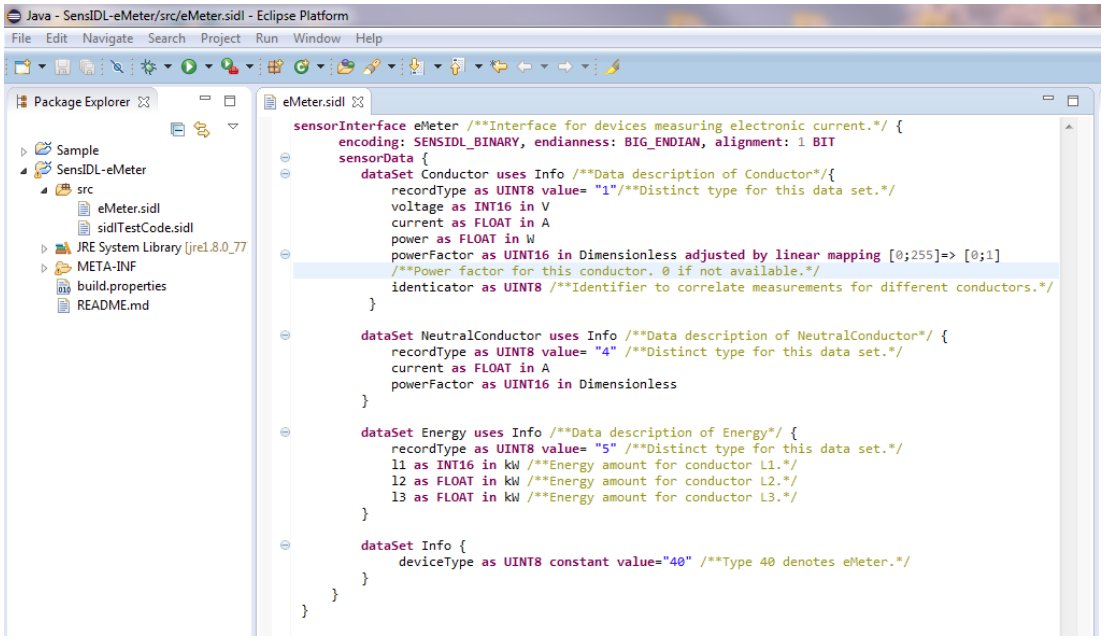

Abb. 2: Eclipse-basierter SensIDL-Editor mit Beispiel

Wie in jeder Programmiersprache werden auch in der SensIDL-Sprache einige Schlüsselwörter definiert (Beispiel in Abb. 2), welche dem Entwickler helfen die Sensorbeschreibung vorzunehmen. Dabei wird zunächst ein `sensorInterface` definiert, welches die Sensorschnittstelle repräsentiert. Innerhalb dieser Interfaces beinhaltet das sog. `sensorData` mehrere `dataSets`, welche die einzelnen Sensorinformationen wiederspiegeln. Jede Sensorinformation kann sich dabei aus verschie-

denen Variablen zusammensetzen, bei denen jeder Variablen neben einem Namen und einem Datentyp auch eine physikalische Einheit und/oder ein Gültigkeitsbereich bzw. Wert zugewiesen werden kann. Der SensIDL Editor unterstützt den Benutzer dabei sowohl durch Syntax-Highlighting als auch durch automatische Vervollständigung und stellt so sicher, dass der Entwickler schnell und effizient in der SensIDL-Sprache Definitionen schreiben kann. Ausgehend von dieser Beschreibung können dann über das Kontextmenü die verschiedenen Generatoren gestartet werden. Aktuell sind Generatoren für die Programmiersprachen C, C++, C#, Java und JScript verfügbar sowie ein Generator zur Erzeugung von Dokumentation der Schnittstellen.

Anwendungsbeispiel

Abb. 2 zeigt einen Screenshot des SensIDL Editors, um die Beschreibung beispielhaft an einem Strommesssensors genannt *eMeter* zu demonstrieren. Der *eMeter* liefert sowohl den Strom auf den einzelnen Leitern als auch die übertragene Energie als Sensorinformationen. Im `dataSet Conductor` werden einige weitere Möglichkeiten von SensIDL aufgezeigt. In der Deklaration des `dataSets` wird die Möglichkeit der sog. Vererbung aufgezeigt. Dabei bildet das Schlüsselwort `uses` die Möglichkeit gleiche Teile in den Sensorinformationen nur einmal definieren zu müssen. Zusätzlich wird deutlich, dass Variablen auch direkt ein Wert zugewiesen werden kann, wie bei `recordType`. Mit der Variablen `powerFactor` wird demonstriert, dass auch ein lineares Mapping bei SensIDL vorgenommen werden kann, was bedeutet, dass ein Byte übertragen wird, aber der mögliche Wertebereich von 0 bis 255 dann auf den Wertebereich 0 bis 1 abgebildet wird. Auch Möglichkeiten wie ein Offset zu setzen oder eine Skalierung vorzunehmen sind implementiert. Im Weitern können auch Methodendeklarationen mit entsprechender Sichtbarkeit definiert werden, um so auch für bestimmte Variablen nur die get- oder set-Methode erstellen zu lassen.

Ausblick

Das Projekt SensIDL befindet sich in der Endphase der initialen Projektförderung und wird danach als Community-getriebenes Open-Source Projekt weiterentwickelt. Neben der Anwendung in eigenen Entwicklungsprojekten ist auch eine projektübergreifende Kooperation mit dem *Eclipse Vorto* (https://eclipse.org/vorto) angestrebt, welches sich auf die Beschreibung von Geräten für das „Internet der Dinge“ und die Generierung von Integrationscode in Middleware-Systeme fokussiert. Eine Modelltransformation aus dem Vorto Projekt in SensIDL soll die Integrationsebene (Vorto-Generatoren) mit der Sensorimplementierung (SensIDL-Generatoren) nahtlos zusammenbringen. Zusätzlich sind noch weitere Schnittstellenanbindungen wie z.B. OPC-UA in Planung.

Autoren:

Nathalie Hipp hat an der Hochschule Furtwangen University in Villingen-Schwenningen Medical Engineering im Bachelor und Biomedical Engineering im Master studiert. Ihr Schwerpunkt lag auf der Programmierung und Simulation von software-basierten Systemen in der Medizintechnik. Seit 2015 ist Frau Hipp im Bereich Software Solutions des Hahn-Schickard-Instituts für Mikro- und Informationstechnik als wissenschaftliche Mitarbeiterin und Software-Entwicklerin angestellt. Ihre Arbeitsbereiche umfassen die Anwendungsgebiete Medizintechnik, Industrie 4.0 und Smart Home. Ihre besonderen Interessen liegen dabei unter anderem in den Forschungsthemen Digitalisierung in der Medizintechnik, Einsatz von Gamification für die Therapieunterstützung sowie Aufbau von Kommunikationsgateways zwischen verschiedenen Funkstandards im Heim- und Industrieumfeld.

Nathalie Hipp
Hahn-Schickard
Wilhelm-Schickard-Str. 10
78052 Villingen-Schwenningen
Nathalie.Hipp@Hahn-Schickard.de
Tel: 07721 943 186

Dr.-Ing. Christoph Rathfelder studierte Informatik an der Universität Karlsruhe und arbeitete seit 2007 im Bereich Software Engineering des FZI Forschungszentrum Informatik in Karlsruhe als wissenschaftlicher Angestellter. 2012 promovierte er am Karlsruher Institut für Technologie im Bereich der Qualitätsanalysen für verteilte Systeme. 2013 wechselte Dr. Rathfelder in den Bereich Sensoren und Systeme des Hahn-Schickard-Instituts für Mikro- und Informationstechnik in Villingen-Schwenningen und hat dann 2015 im neu gegründeten Bereich Software Solutions die Leitung der Anwendungsentwicklung übernommen. Dr. Rathfelder ist stellvertretender Vorstand des Smart Home & Living Baden-Württemberg Vereins sowie in verschiedenen nationalen und internationalen Arbeitsgruppen zu den Themen Industrie 4.0, Smart Systems und dem Internet der Dinge bei Microtec Südwest, der Bitkom sowie der European Plattform on Smart System Integration (EPoSS) tätig.

Dr.-Ing. Christoph Rathfelder
Hahn-Schickard
Wilhelm-Schickard-Str. 10
78052 Villingen-Schwenningen
Christoph.Rathfelder@Hahn-Schickard.de
Tel: 07721 943 161

Emre Taspolatoglu studierte Informatik am Karlsruher Institut für Technologie (KIT). Seine Diplomarbeit führte er im Forschungsbereich SE des FZI durch und beschäftigte sich mit den Coderelevanten Entwurfsentscheidungen während der Software-Evolution. Seit September 2014 ist er für das FZI Forschungszentrum Informatik als wissenschaftlicher Mitarbeiter im Forschungsbereich Software Enginee-

ring (SE) tätig und beschäftigt sich als Doktorand mit sicherheitsrelevanten Fragestellungen und Entwurfsentscheidungen auf der Ebene von Softwarearchitekturen sowie deren musterbasierten Analysen.

Emre Taspolatoglu
FZI Forschungszentrum Informatik
Haid-und-Neu-Str. 10-14
76131 Karlsruhe
taspolat@fzi.de
Tel: 0721 9654 618

Jörg Henß studierte Informatik an der Universität Karlsruhe und forschte seit 2009 am Lehrstuhl für Software-Design und Qualität am Karlsruher Institut für Technologie an Methoden zur Interoperabilität von Simulationen und dem Einsatz von modellgetriebenen Technologien. 2015 wechselte Herr Henß zum FZI Forschungszentrum Informatik und fungiert dort seit Anfang 2016 als Abteilungsleiter im Bereich Software Engineering (SE).

Jörg Henß
FZI Forschungszentrum Informatik
Haid-und-Neu-Str. 10-14
76131 Karlsruhe
henss@fzi.de
Tel: 0721 9654 630