# SysML-SFM Spezifikation

**zum Projekt / Produkt**

# ProcAppCom

**zum Thema / Modul**

# Modellierungsansatz

**GEFASOFT GmbH**
Dessauerstraße 15 · 80992 München · Tel. +49 (89) 125565 - 0
Fax +49 (89) 125565 - 180 · mailto:info@gefasoft.de · www.gefasoft.de

# Änderungshistorie

| Version | Datum | Autoren | Änderungen |
|---|---|---|---|
| 1 | 22.08.2017 | D. Schütz | • Erstellung des Dokuments |
| 2 | | | |
| 3 | | | |
| 4 | | | |
| 5 | | | |
| 6 | | | |
| 7 | | | |
| 8 | | | |
| 9 | | | |
| 10 | | | |

# Freigabe durch …

_____ _____

Ort, Datum Unterschrift

# Inhaltsverzeichnis

# 1 Einleitung

Die Grundlage des hier beschriebenen Modellierungsansatzes des Profils SysML-SFM (sprich: SysML für Shopfloor Modelle) bilden das Metamodell der Unified Modeling Language (UML, V2.0), das darauf aufbauende Profil der Systems Modeling Language (SysML, V1.4) und das ergänzende Profil AML4SysML, das die Verwendung von Konzepte der Automation Markup Language (AML) innerhalb von SysML sowie eine Transformation zwischen beiden Sprachen ermöglicht.

Bei der UML sowie der SysML handelt es sich um Sprachen, deren Metamodelle auf der Meta-Object Facility (MOF) beruhen, und die, wie MOF selbst, von der Object Management Group (OMG) standardisiert sind. Bei dem Profil AML4SysML, das die SysML für eine Verwendung von AML-Konzepten ergänzt, handelt es sich um das Ergebnis einer gemeinsamen Arbeit der zwei Forschungsgruppen um Prof. Lüder (OvGU Magdeburg) bzw. Prof. Wimmer (TU Wien).

Die hier zugrunde gelegten Sprachen werden hier nicht näher erläutert; es wird ein grundlegendes Verständnis von UML/SysML und deren Metamodellen und Profilmechanismen sowie der Sprache AML vorausgesetzt. Die Spezifikationen der Metamodelle von UML und SysML sind im Anhang dieses Dokumentes verlinkt. Ebenfalls befinden sich weitere Literaturangaben und Referenzen für zusätzliche Informationen im Literaturverzeichnis dieses Dokuments.

Die SysML-SFM besteht aus zwei Teilen: einem UML-Profil, in dem die definierten Stereotypen definiert sind (außerdem auch Stereotypen, die von der AML4SysML eingeführt wurden) und einem Bibliotheks-Modell, in dem verschiedene wiederverwendbare Modellartefakte (insbes. zu verwendende Datentypen) definiert sind. Sowohl das Profil als auch das Bibliotheks-Modell sind wurden mittels des Eclipse-Plugins Papyrus erstellt und liegen dieser Spezifikation als archivierte Projekte bei. Die aus AML4SysML übernommenen Stereotypen sind in Abbildung 1 dargestellt.

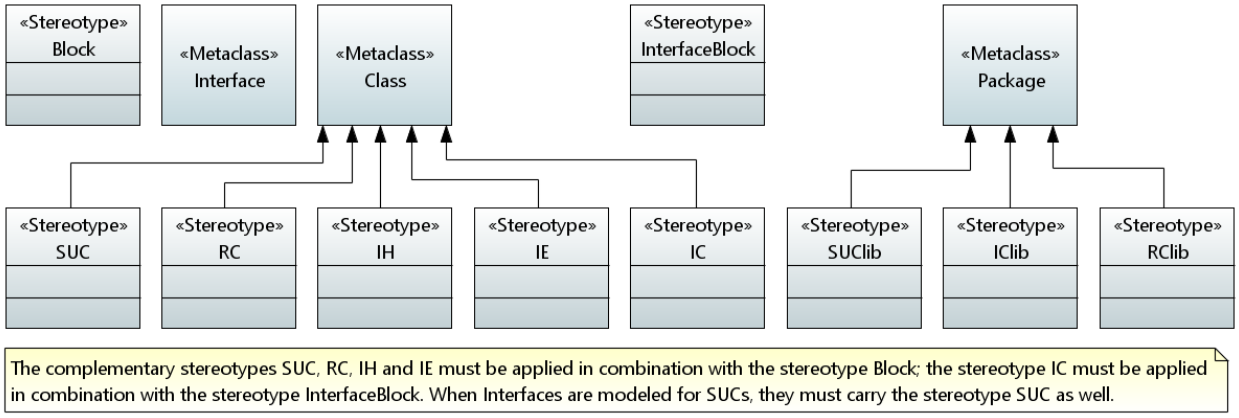


Abbildung 1: Aus *AML4SysML* für *SysML-SFM* übernommene Stereotypen

# 2 Genereller Aufbau eines Modells

Der Grundsätzliche Aufbau eines Modells (gekapselt durch das UML/SysML Konzept *Model*) sieht mehrere *Packages* vor, in die jeweils die entsprechenden Modellelemente eingruppiert werden. Die *Packages* repräsentieren innerhalb der SysML die verschiedenen Bibliotheks-Typen der AML, d.h. *SystemUnitClassLibrary (SUClib)*, *InterfaceClassLibrary (IClib)* und *RoleClassLibrary (RClib)*, und sind zu diesem Zweck durch die entsprechenden Stereotypen der AML4SysML gekennzeichnet. In

Gegensatz zu den Bibliotheks-Typen werden *InstanceHierarchies (IH)* in SysML mittels Blöcken modelliert, die ergänzend den entsprechenden Stereotyp tragen. Von diesen können in einem Modell beliebig viele existieren. Blöcke mit der Erweiterung *IH* liegen direkt in einem Modell und sind nicht in einem *Package* eingruppiert.

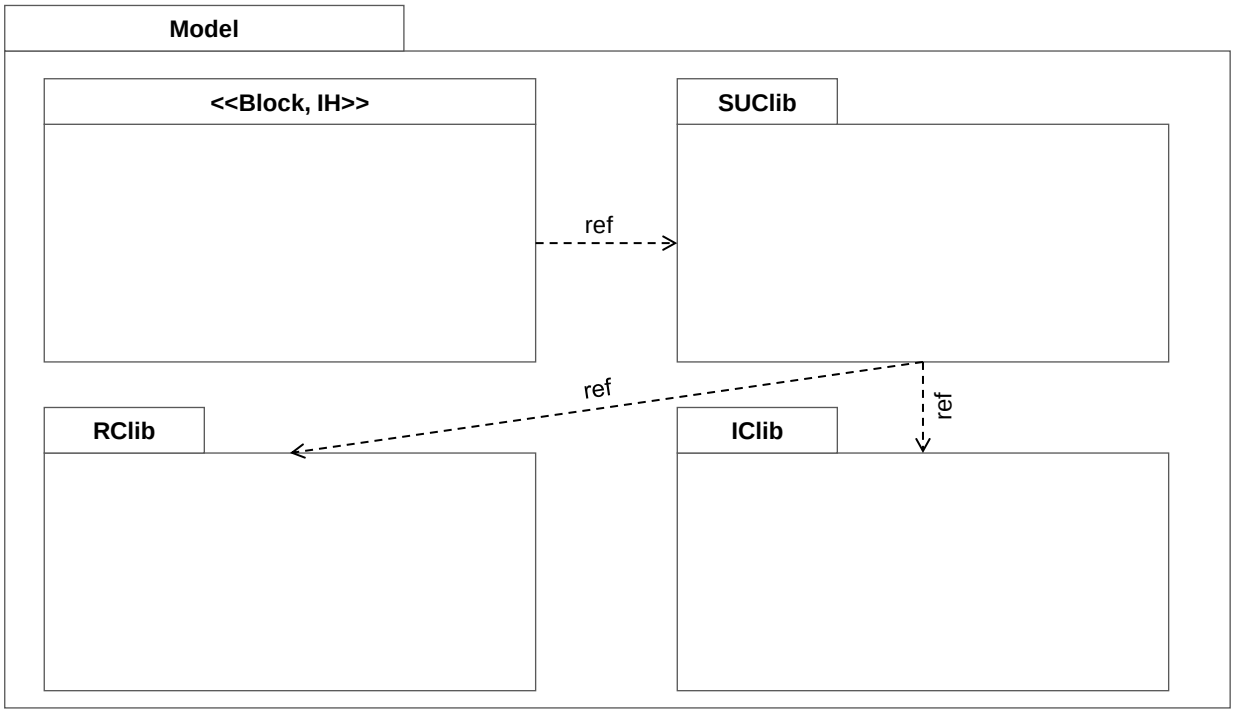


Abbildung 2: Schaubild für den grundsätzlichen Aufbau eines Modells

# 3 Nutzung objektorientierter Konzepte

Der Modellierungsansatz, der der SysML-SFM zugrunde liegt, unterstützt innerhalb der ersten Stufe der Modellierung, d.h. bei der Modellierung von Knoten als *SUCs* innerhalb einer *SUClib*, verschiedene Konzepte der objektorientierten Modellierung.

So ist es beispielsweise mittels **Vererbung** möglich, die Eigenschaften und Parts von einem *Block* (d.h. einer *SUC*) an einen anderen Block (d.h. eine *SUC*) zu vererben. Wie in der Objektorientierung üblich können in einem Block geerbte Eigenschaften wie **auch geerbte Eigenschaften überschrieben** werden. Ähnlich wie bei dem Überschreiben eines Methodenaufrufs in objektorientierten Programmiersprachen, ist in SysML dazu in einem erbenden *Block* ein Element mit gleichem Namen anzugeben und dann gegenüber der geerbten Eigenschaft abzuwandeln (bspw. eine Eigenschaft "Baujahr", die mit Datentyp *String* geerbt wird, dann im Kind-Block mit Datentyp *Int* überschrieben).

Ebenso lässt der Modellierungsansatz um die SysML-SFM die Modellierung von **Interfaces** zu, in denen Eigenschaften definiert werden können, die mehrere Blöcke tragen sollen. Da das Konzept *InterfaceBlock* der SysML im Rahmen von AML4SysML bereits mit der Abbildung des AML Konzepts *InterfaceClass (IC)* belegt ist (s. Abbildung 1), wird zur Definition von Eigenschaftsschnittstellen innerhalb von SysML-SFM das UML Konzept *Interface* verwendet. Da die objektorientierten Konzepte nur im Rahmen der *SystemUnitClasses (SUC)* eines Modells verwendet werden dürfen (vgl. Kapitel 4), wird der Stereotyp *SUC* auch auf das Metaelement *Interface*

angewendet und es gelten folgende Einschränkungen (zu Illustrationszwecken sind nach diesen Regeln unzulässige Beziehungen in Abbildung 3 in roter Farbe dargestellt):

- *Blöcke* mit dem Stereotyp *SUC* können nur von *Blöcken* mit dem Stereotyp *SUC* erben
- *Blöcke* mit dem Stereotyp *SUC* können nur *Interfaces* mit dem Stereotyp *SUC* implementieren

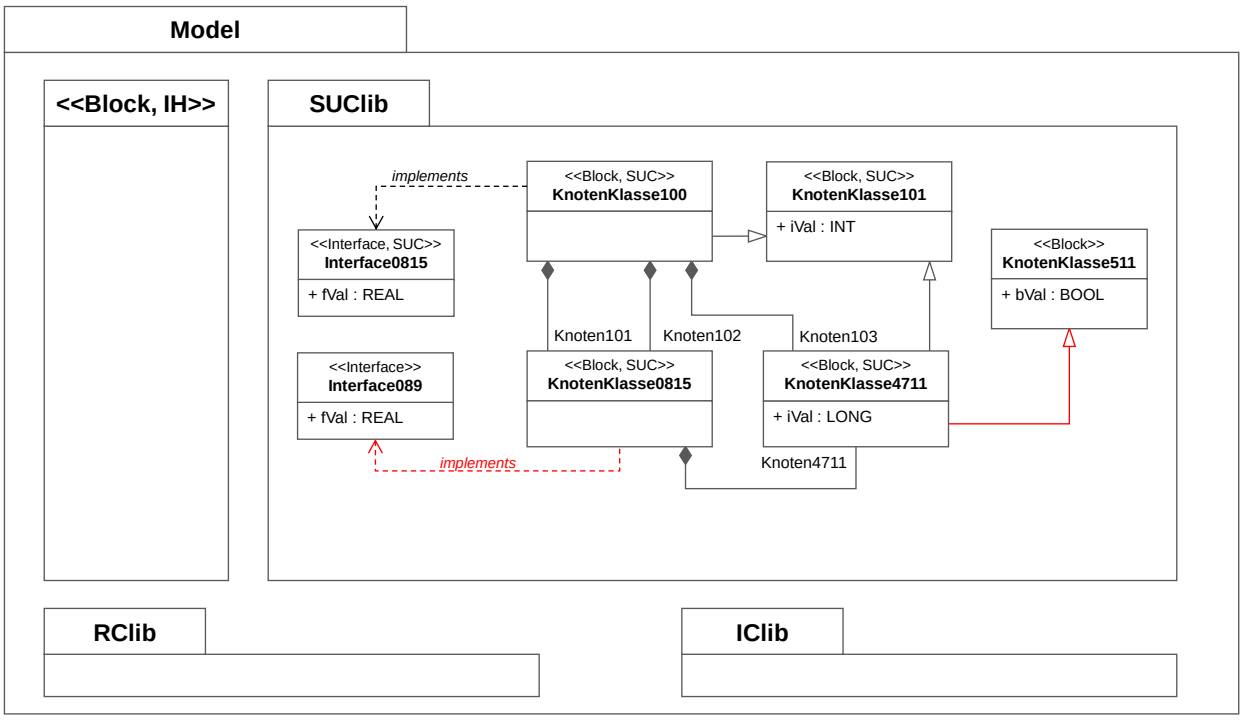


Abbildung 3: Vorgesehene objektorientierte Modellierungskonzepte innerhalb von SysML-SFM

# 4 Mehrstufiger Ansatz zur Modellerstellung

Bedingt durch die ergänzende Anwendung des Profils AML4SysML wird die Modellerstellung bzw. Modellbearbeitung in zwei Stufen aufgeteilt. In der ersten Modellstufe wird das Modell eines Shopfloors mit den objektorientierten Mitteln (d.h. Vererbungsbeziehungen, Interfaces, etc.; vgl. Kapitel 3) durchgeführt; verschachtelte Hierarchien werden durch Klasse-Instanz Beziehungen (d.h. mittels einer *Komposition* bzw. *composite association*) modelliert (vgl. Abbildung 4, rechts). Diese Stufe entspricht Aufbau und Pflege von *SystemUnitClassLibraries (SUCLib)* von AML. Dementsprechend werden in dieser Stufe der Modellierung auch die entsprechenden Stereotypen *SUC* ergänzend auf die SysML Konzepte (hier: *Block* bzw. *Interface*) angewendet.

Die zweite Stufe bildet der Aufbau der konkreten Instanz des Modells eines Shopfloors. Das innerhalb einer *SUCLib* erstellte Modell in Form einer oder mehrerer *SUC* wird dazu in eine *InstanceHierarchy (IH)* umgewandelt. Bei diesem Prozess werden Kopien der (dafür ausgewählten) *SUC*, modelliert auf Grundlage von *Blöcken*, erzeugt und in *Blöcke* umgewandelt, die den Stereotyp *InternalElement (IE)* tragen (vgl. Abbildung 4, links). Bei diesem Prozess werden sämtliche über *Interfaces* implementierte Elemente (sofern nicht überschrieben) sowie geerbte Elemente eines *Blocks* direkt diesem selbst hinzugefügt. Dies umfasst auch die in einer *SUC* bereits vergebenen Werte der jeweiligen Eigenschaften. Außerdem werden die in den *SUC* über Klasse-Instanz Beziehungen verschachtelten Elemente in als *nestedClassifier* verschachtelte Elemente umgewandelt. Dazu wird

für jede Instanz eines *Blocks* eine Kopie erzeugt, sodass in diesem *Block* spezifische Werte für *Properties* bzw. für die *TaggedValues* der angewendeten Stereotypen gesetzt werden können.

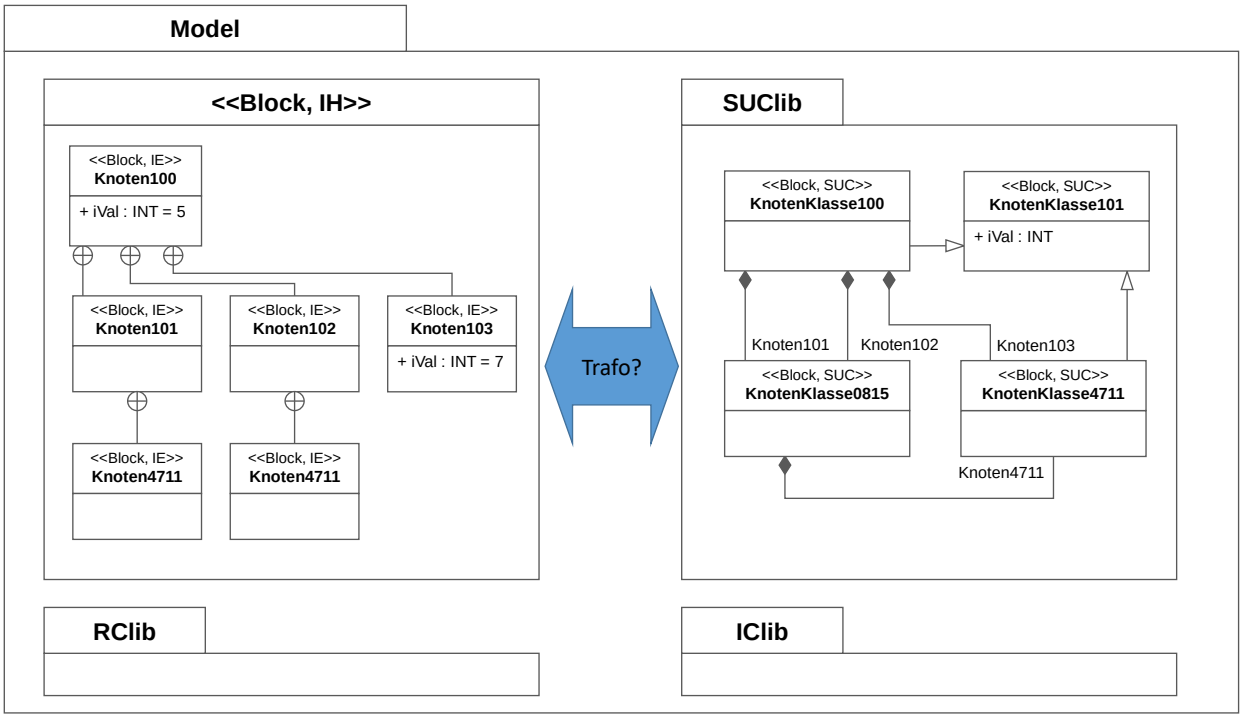


Abbildung 4: Schaubild für die zwei Stufen des Modellierungsansatzes

Für den Übergang zwischen der ersten und der zweiten Stufe ist eine (bi-direktionale) Modelltransformation auf Basis der Query-View-Transformation Language (QVT) vorgesehen. Diese benutzt ein Modell sowohl als Ein- als auch als Ausgang und nimmt die Übersetzung automatisch vor. Da diese Transformation noch nicht implementiert ist, muss die Übersetzung zwischen *SUC* und *IH* entweder manuell vorgenommen werden, oder aber die Modelle müssen zum jetzigen Zeitpunkt noch direkt in der *IH* erstellt werden.

# 5 Modellierung von Knoten-Hierarchien

Das grundlegende Konzept, das innerhalb von SysML-SFM für die Modellierung eines Knotens (die Bezeichnungen Knoten und Modul werden in diesem Dokument synonym für eine Maschine oder Maschinengruppe auf dem Shopfloor verwendet) verwendet wird, ist das Element *Block* der SysML. Für die Definition, um welchen Knoten-Typ es sich bei einem *Block* handelt, werden innerhalb des Profils SysML-SFM eine Reihe von ergänzenden Stereotypen definiert, mit denen die Modul-Typen abgebildet werden die in der Norm ISA S88 definiert sind (vgl. Abbildung 5).

In späteren Weiterentwicklungen des Modellierungsansatzes können auch andere Formen von Modul-Hierarchien und Modul-Typen (bspw. auch firmenspezifische) verwendet werden. Im Rahmen des Projekts ProcAppCom (Zieldomäne: Verfahrenstechnik) wird jedoch die Modul-Definition aus dem ISA S88 Standard fokussiert. Abhängig von der Stufe der Modellierung (Modellierung von *SUCs* oder *IHs*) sind zusätzlich die entsprechenden ergänzenden Stereotypen aus dem Profil AML4SysML an die *Blöcke* zu vergeben.

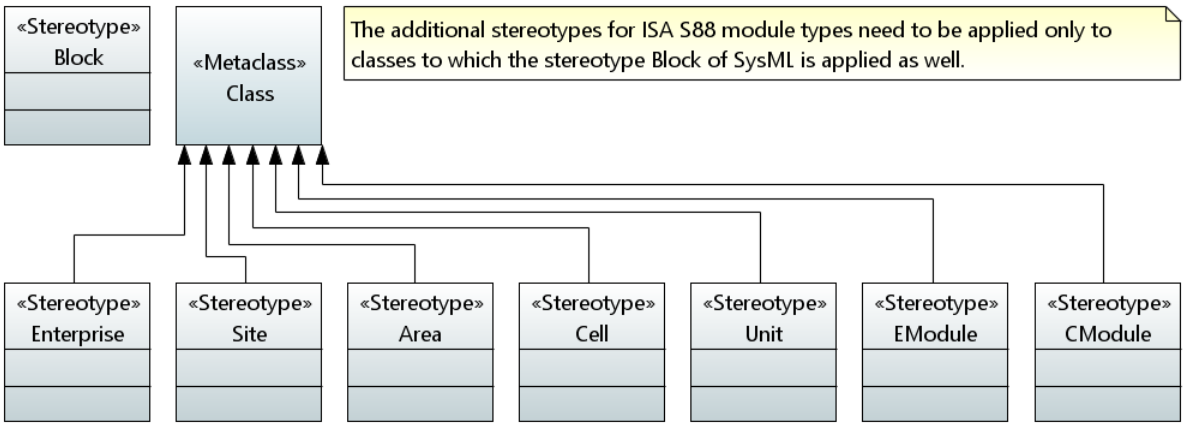


Abbildung 5: Sterotypen definiert zur Abbildung der ISA S88 Modul-Typen

# 6 Modellierung von Flussbeziehung zwischen Knoten

Für die Modellierung von Flussbeziehungen zwischen den Knoten eines Modells verwendet die SysML-SFM die Konzepte der SysML: *ProxyPort* (zur Modellierung der Schnittstellen eines Knotens) und *BindingConnector* (zur Modellierung der Verbindung zweier Schnittstellen). Für die Angabe eines konkreten Typs der Schnittstelle (bspw. Druckluftanschluss) sieht die SysML-SFM das SysML Konzept *InterfaceBlock* (gemeinsam mit dem AML4SysML Stereotypen *IC*) vor.

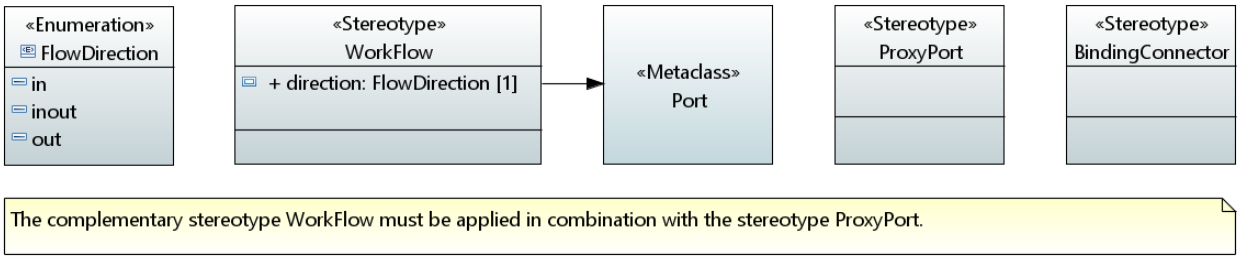


Abbildung 6: Profildiagramm mit den Stereotypen zur Modellierung von Flussbeziehungen

Die in einem Modell verwendeten *InterfaceBlöcke* werden, entsprechend der Konventionen der AML4SysML, in einer *IClib* (vgl. Abbildung 2) gesammelt und mit dem Stereotyp *IC* versehen und können dann als *type* eines *ProxyPorts* ausgewählt werden. Das Element *BindingConnector* schränkt die Modellierung von Flussbeziehungen insofern ein, als das nur Schnittstellen (=*ProxyPorts*) des gleichen Typs miteinander verbunden werden können.

Der zunächst wesentliche Anwendungsfall für die Modellierung von Flussbeziehungen innerhalb eines Shopfloor Modells liegt in der Abbildung von Prozessketten, d.h. Vorgänger/Nachfolger-Beziehungen zwischen Knoten bezogen auf den Fluss von gefertigten Produkten in ihren verschiedenen Stufen.

Für diese spezielle Art von Schnittstellen führt die SysML-SFM den Stereotypen *WorkFlow* ein, der auf die *ProxyPorts* eines Knotens angewendet werden kann. Die *InterfaceBlöcke*, die den jeweiligen Typ eines solchen *ProxyPorts* beschreiben, repräsentieren bzw. beschreiben dabei die verschiedenen Fertigungsstufen (Rohteil, Halbzeug, etc.) eines Produkts.

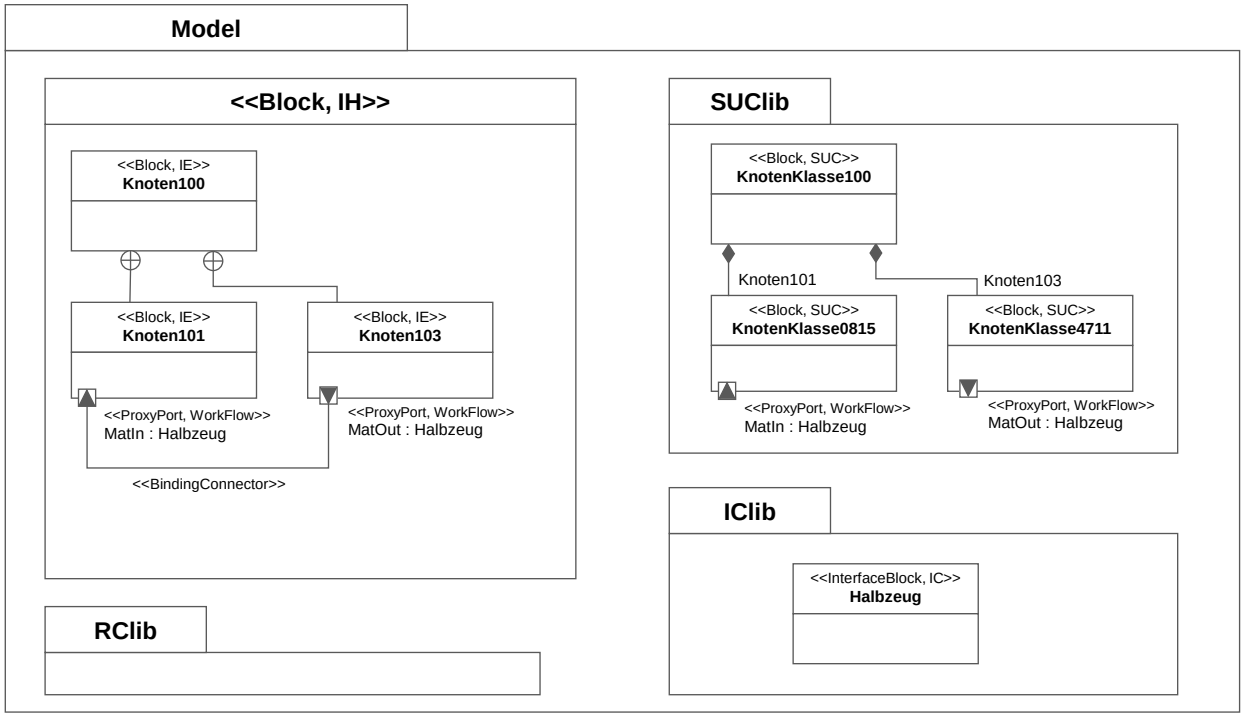


Abbildung 7: Beispiel der Modellierung von Materialflussschnittstellen

Da die Verbindungen der (Material-)Flussbeziehungen nur zwischen den konkreten Knoten-Instanzen modelliert werden sollen, zwischen denen die Beziehung besteht, und nicht für alle Knoten einer bestimmten Klasse, werden die Verbindungen (=*BindingConnector*) erst in der zweiten Stufe der Modellierung (vgl. Kapitel 4) zwischen den *InternalElements (IE)* einer *InstanceHierarchy (IH)* gezogen. Die Verletzung der Grenzen von Knoten ist bei der Modellierung von Flussbeziehungen nicht erlaubt; das heißt, dass keine Verbindungen (*BindingConnector*) über die Grenzen eines Blocks hinweg gezogen werden dürfen.

# 7 Modellierung der Eigenschaften eines Knoten

Zur Modellierung der Eigenschaften eines Knoten, bspw. des Herstellers eines Moduls oder der Versionsnummer eines Sensors, etc., wird innerhalb der SysML-SFM das Konzept der *Properties* mit elementaren Datentypen (*Bool, Int, Real, String*) sowie (nutzer-definierten) *Enumerationen* verwendet. Die elementaren Datentypen sind in dem Bibliotheks-Modell der SysML-SFM enthalten, das in jedem konkreten Anwendungsmodell importiert werden muss. Verschiedene hilfreiche *Enumerationen* sind dort ebenfalls bereits definiert. Bei diesen Eigenschaften kann es sich sowohl um Stammdaten handeln, die für einen Knoten im MES hinterlegt werden sollen, als auch um Projektierungsdaten, die für die Entwicklung relevant sind, aber nicht in die Datenbank des MES überführt werden sollen.

Um zu kennzeichnen, dass es sich bei der *Property* eines *Blocks* um ein für das MES relevantes Stammdatum eines Knotens handelt, führt die SysML-SFM den ergänzenden Stereotyp *SdScope* (für 'Stammdaten Scope') ein. Diese Kennzeichnung von Properties erlaubt bei späteren Generierungsprozessen die einfache Selektion und Transformation von Eigenschaften, die für die Stammdaten eines Knotens relevant sind. Zur Kennzeichnung von Eigenschaften, die für die Projektierung eines MES relevant sind aber nicht innerhalb der Stammdaten eines Knotens in der DB hinterlegt werden sollen, führt die SysML-SFM den Stereotyp *PrjScope* ein (für 'Projekt Scope').

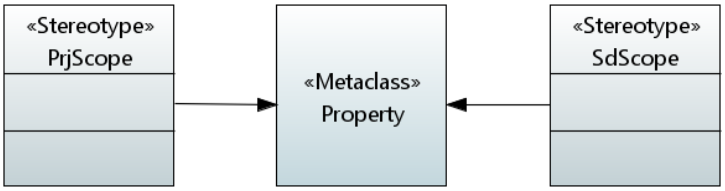


Abbildung 8: Stereotypen für die Eigenschaften eines Knotens

Durch die objektorientierten Konzepte der UML/SysML können in der ersten Stufe der Modellerstellung solche Eigenschaften von Knoten auch in *Interfaces* gruppiert bzw. zwischen *Blöcken* vererbt und in erbenden *Blöcken* (teilweise) überschrieben werden. Bei der Transformation zwischen der ersten und der zweiten Stufe der Modellerstellung werden, wie in Kapitel 4 bereits beschrieben, sämtliche Eigenschaften, die ein *Block* von anderen erbt bzw. über ein *Interface* implementiert, als direkte Eigenschaften des Blocks selbst umgewandelt. Es wird somit quasi eine spezifische Instanz erzeugt eines *Blocks* erzeugt, in der dann die spezifischen Werte für die Eigenschaften eines *Blocks*, bspw. der Hersteller eines Moduls, vergeben werden können (vgl. Abbildung 9). Die Vergabe von Werten erfolgt im SysML Modell über die Definition eines *DefaultValue* für die jeweilige *Property*. Im Falle von elementaren Datentypen werden diese über entsprechende Literale (*LiteralBool, LiteralInt, LiteralFloat, bzw. LiteralString)* angegeben; im Fall, dass eine *Property* als Datentyp eine *Enumeration* besitzt, kann direkt eines von deren Literalen ausgewählt werden.

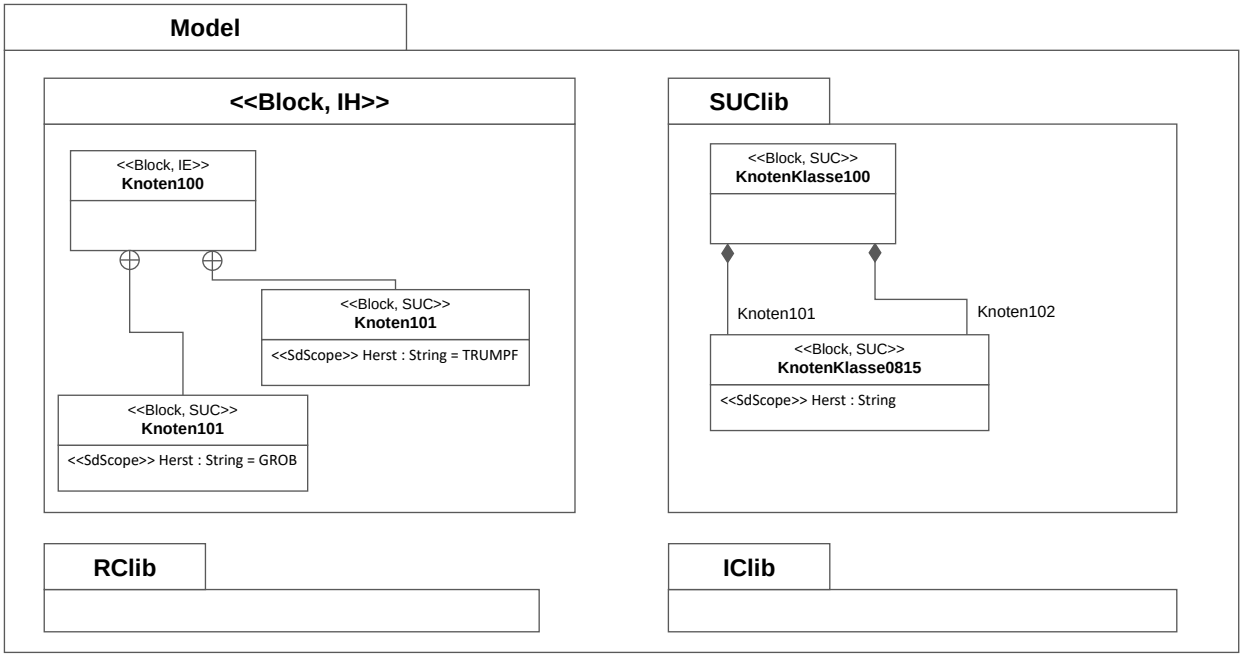


Abbildung 9: Beispielhafte Darstellung der Modellierung von Stammdaten (Werte erst in *IH*)

Da bei der Transformation von der ersten (*SUClib*) in die zweite (*IH*) Modellierungsstufe die Werte der *Properties* eines *Blocks* mit übernommen werden, ist es auch möglich innerhalb einer *SUClib* bereits Werte zu vergeben. Diese müssen dann innerhalb der daraus erzeugten *IH* nicht mehr gesetzt werden. Dies bietet sich insbesondere dann an, wenn mehrere Knoten die gleichen Eigenschaften und diese auch die gleichen Werte tragen (bspw. wenn mehrere Maschinen von einem Hersteller

sind). Hierfür kann, aufbauend auf den objektorientierten Konzepten, ein *Interface* definiert werden, das diese Eigenschaften definiert und dann auch bereits mit *DefaultValues* spezifiziert (vgl. Abbildung 10).

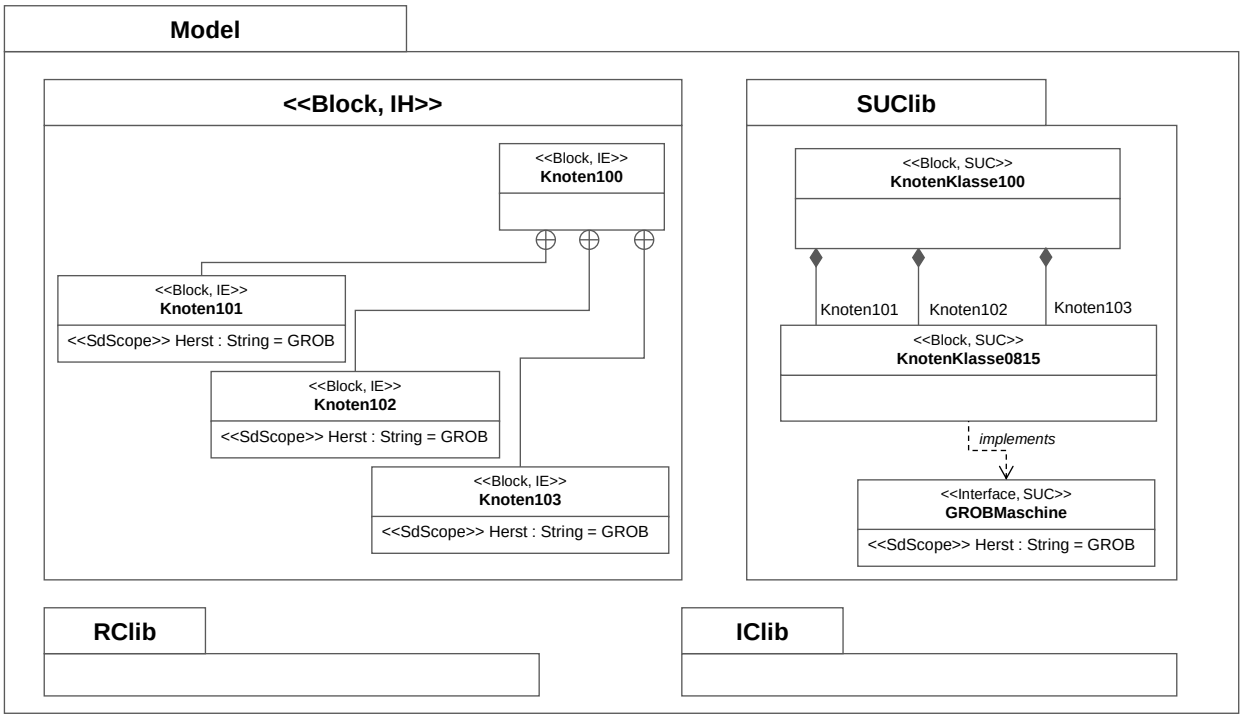


Abbildung 10: Beispielhafte Darstellung der Modellierung von Stammdaten mittels Interfaces

# 8 Modellierung der Variablen von Knoten

Ähnlich wie die Eigenschaften von Knoten, die als Stamm- oder Projektierungsdaten eines Knotens im Modell beschrieben werden, werden auch die Eigenschaften eines Knotens, die durch das MES zu überwachende / koppelnde Variablen definieren, in Form von *Properties* eines *Blocks* modelliert. Zur Kennzeichnung, dass es sich dabei um eine solche Variable handelt wird innerhalb der SysML-SFM der Stereotyp *VarScope* (für 'Variablen Scope') eingeführt (vgl. Abbildung 11).

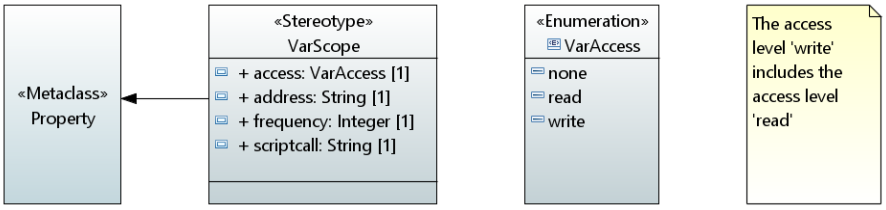


Abbildung 11: Stereotypen für die Variablen eines Knotens

Über die *TaggedValues*, die für diesen *Stereotyp* definiert wurden lassen sich für die zu koppelnden Variablen die Zugriffsrechte (*access : VarAccess*), die jeweilige Adresse (*address : String*) die Frequenz in Millisekunden mit der die Variable gelesen/geschrieben werden soll (*frequency : Integer*) sowie die

Verknüpfung zu einem Verarbeitungsskript (*scriptcall : String*) für den Wert der Variable. Für den Fall der Generierung eines SCC Projekts kann über die Vergabe eines Werts für den *scriptcall* der Aufruf eines Scripts codiert werden. Die vollständige Modellierung von Scripts innerhalb der SysML-SFM ist zum jetzigen Zeitpunkt zwar vorgesehen, jedoch noch nicht umgesetzt. Die Adresse einer Variable kann durch den Datentyp String flexibel sowohl absolut (bspw. auf einem AG) oder symbolisch auf einem OPC UA Server angegeben werden.

# 9 Modellierung der (Hardware-)Schnittstellen von Knoten

Die Modellierung der Hardware (bzw. deren Schnittstellen) von Knoten führt die SysML-SFM mehrere unterschiedliche Stereotypen ein. Alle diese Stereotypen leiten sich dabei von einem abstrakten Stereotyp *HardwareNode* (vgl. Abbildung 12) ab und können auch ergänzend zueinander gemeinsam mit dem Stereotyp *Block* angewendet werden, der einen Knoten im Modell beschreibt. Dadurch können für einen einzelnen Knoten auch mehrere Interfaces, je eines pro angelegtem Stereotyp, definiert werden. Über die *TaggedValues*, die vom abstrakten Stereotyp an die anderen Stereotypen vererbt werden, lassen sich dabei die IP-Konfiguration (*ipconfig : eIpConfig*) wie auch die IP-Adresse (*address : String*) spezifizieren. Von *HardwareNode* leiten sich zwei weitere Stereotype *MachineNode* (zur Modellierung von allgemeinen nicht näher spezifizierten Maschinensteuerung) und *ClientNode* (zur Modellierung von speziellen Maschinenclients) ab. Diese sind in der aktuellen Version des Ansatzes lediglich als Platzhalter für spätere Erweiterungen vorgesehen.

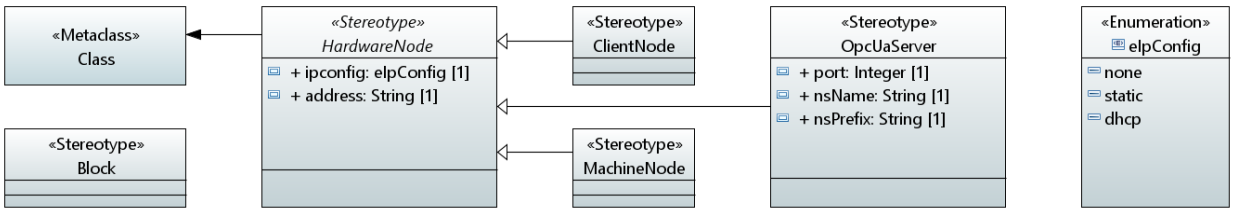


Abbildung 12: Stereotypen zur Modellierung der (Steuerungs-)Hardware von Knoten

Der derzeit einzig relevante Stereotyp zur Modellierung der Hardware eines Knotens ist *OpcUaServer*, mit dem Knoten gekennzeichnet werden können, deren Steuerungshardware einen OPC UA Server zur Kopplung zur Verfügung stellen. Mit diesem Stereotyp sind TaggedValues spezifiziert, mit denen der Port (*port : Integer*) spezifiziert werden kann, unter dem der OPC UA Server angesprochen werden kann. Darüber hinaus kann der Name des Namespaces (*nsName : String*) definiert werden, in dem die zu koppelnden Variablen eines Knotens lokalisiert sind. Außerdem kann ein Präfix (*nsPrefix : String*) vorgegeben werden, dass für die Adressen aller zu koppelnden Variablen gilt.

Hinweis: Sind in einer Hierarchie mehrere Knoten mit OPC UA Servern ineinander verschachtelt, wird die Annahme getroffen, dass die Variablen (VarScope) der Knoten (wenn nicht auf dem jeweiligen Knoten selbst) immer auf dem in der Hierarchie nächst-höher gelegenen Knoten mit einem OPC UA Server gekoppelt werden können. Sprich: Jeder Knoten mit OPC UA Server macht in der verschachtelten Hierarchie einen Zweig auf, und allokiert die Variablen, die in diesem Zweig modelliert werden.