

Implementierungsdokumentation

**Definition und Durchführung von
Messwertverarbeitung
für den Physikunterricht
auf Basis eines Raspberry Pis**

Version 1.0.0

David Gawron Stefan Geretschläger Leon Huck
Jan Küblbeck Linus Ruhnke

10. August 2019

Inhaltsverzeichnis

1	Ziel der Implementierungsdokumentation	3
2	Ausarbeitungsstand der Abnahmekriterien	4
2.1	Entwurf von Messkonfigurationen	4
2.2	Handhabung von Bausteinprototypen	5
2.3	Gewährleisten von Persistenz	5
2.4	Bereitstellung vorgefertigter Teile	6
2.5	Handhabung von Messläufen	6
2.6	Benutzbarkeit der GUI	6
2.7	Abgrenzungskriterien	7
3	Umsetzung des Entwurfs	8
3.1	Model	9
3.1.1	Ersetzen des Entwurfsmuster Erbauer durch eine Zuständigkeitskette	9
3.2	View	11
3.2.1	GUI-Paket	11
3.2.2	Menu-Paket	11
3.2.3	Configuration-Paket	12
3.2.4	BlockProperties-Paket	13
3.2.5	Exception-Paket	13
3.2.6	HelpAndOption-Paket	13
3.2.7	Model-Interface-Paket	13
3.2.8	Controller-Interface-Paket	14
3.3	Controller	14
3.4	Backend	14
3.4.1	SSH-Verbindung	14
3.4.2	Simulierte Verbindung	14
3.4.3	Nebenläufigkeit	15
3.5	Cache	15
3.6	File-Service	15
4	Realer Implementierungsablauf	15
5	Anhang	16
6	Glossar	18

1 Ziel der Implementierungsdokumentation

2 Ausarbeitungsstand der Abnahmekriterien

Im Folgenden werden die Muss-, Soll- und Wishkriterien aus dem Pflichtenheft herangezogen, das in der ersten Phase des Projekts entstanden ist. Es findet eine Bestandsaufnahme statt, inwieweit die Kriterien erfüllt sind.

Falls das Softwareprodukt ein Musskriterium nicht wie im Pflichtenheft beschrieben aufweist, so führt dieses Dokument detailliert die Ursachen und Gründe hierfür auf. Falls das Softwareprodukt ein Sollkriterium nicht wie im Pflichtenheft beschrieben aufweist, so beschreibt dieses Dokument zwar nicht in jedem Detail, aber hinreichend informativ die Ursachen und Gründe hierfür. Nicht umgesetzte Wishkriterien werden lediglich benannt, aber nicht hinterfragt.

2.1 Entwurf von Messkonfigurationen

Es fand ein Fallback statt. Die Messkonfigurationen werden nicht wie gewünscht graphisch durch ein Drag- and Drop Feld erstellt, sondern müssen textuell eingegeben werden. Dadurch verändert sich auch die Betrachtung, wie und ob die folgenden Kriterien überhaupt erfüllt werden können.

MK 1 Das Musskriterium „Hinzufügen eines Bausteins aus dem Prototypen-Feld zu der Messkonfiguration“ ist TODO

MK 2 Das Musskriterium „Anpassen von wichtigen funktionalen Bausteineigenschaften“ ist TODO

MK 3 Das Musskriterium „Löschen eines Bausteins aus der Messkonfiguration“ ist erfüllt, da der Benutzer die Textuelle Repräsentation eines Bausteins aus der Messkonfiguration entfernen kann.

MK 4 Das Musskriterium „Erstellen einer Verbindung“ ist umgesetzt. Der Benutzer kann ein Kanaltupel der Liste an Verbindungen hinzufügen und somit eine Verbindung der Messkonfiguration hinzu fügen.

MK 5 Das Musskriterium „Löschen einer Verbindung“ ist erfüllt. Der Benutzer kann eine Verbindung aus der Liste der Verbindungen löschen, in dem er das entsprechende Kanaltupel löscht.

SK 1 Das Sollkriterium „Undo-Redo-Funktion“ ist nicht umgesetzt. Die Messkonfiguration wird textuell erstellt und der Editor unterstützt keine Undo-Redo-Funktion.

WK 1 Das Wunschkriterium „Hinzufügen, Bearbeiten und Löschen von ergänzender Informationen zu der Messkonfiguration durch den Benutzer“ ist nicht umgesetzt.

2.2 Handhabung von Bausteinprototypen

SK 2 Der Benutzer ist in der Lage die Eigenschaften der Bausteinprototypen einzusehen. Jedoch ist die Ansicht auf das Anzeigen der *todo* beschränkt. Der Grund hierfür ist die Anbindung von dem Model an die GUI. Dadurch ist es aktuell nur möglich mit den allgemeinen Bausteinprototyp-Informationen zu arbeiten. Dementsprechend ist das Anzeigen der speziellen Eigenschaften, der Bausteinprototypen, nicht möglich.

SK 3 Das Kopieren der Bausteinprototypen ist möglich. Auch das Anpassen der allgemeinen Eigenschaften ist möglich. Jedoch können keine generischen Bausteinprototypen erstellt werden. Dafür wäre die Erstellung einer allgemeinen Vorläge nötig gewesen. Diese Erweiterung hätte dem Nutzer jedoch keine weitere Funktionalität geboten. Aus diesem Grund haben wir uns dazu entschieden die Funktion erst in einer eventuellen Erweiterung der Anwendung zu integrieren.

WK 2 Die Verwendung von erweiternder Software, über eine Schnittstelle, ist nicht mehr vorgesehen. Externe Software kann weiterhin zur Erstellung von Yaml-Dateien genutzt werden. Die so entstandenen Yaml-Dateien können über die Lade-Funktion der Anwendung aufgerufen und anschließend verwendet werden.

SK 4 Die Anwendung unterscheidet in ihrer jetzigen Form nicht zwischen Benutzerdefinierten- und System-Bausteinen. Deshalb ist es auch hier nur möglich die allgemeinen Eigenschaften der Bausteinprototypen zu ändern.

SK 5

2.3 Gewährleisten von Persistenz

MK 6 Das Kriterium ist erfüllt. Die verwendeten Bausteine und ihre Anordnung, also die Messkonfiguration, kann in einer Datei gespeichert werden.

MK 7 Das Kriterium ist erfüllt. Messkonfigurationen können aus Dateien geladen werden.

SK 6 Das Kriterium ist nicht erfüllt, da keine neuen Prototypen in der Anwendung erstellt werden können. (Siehe SK 3)

SK 7 Bausteine werden aus Dateien geladen, die jedoch nur außerhalb der Anwendung erstellt werden können.

2.4 Bereitstellung vorgefertigter Teile

2.5 Handhabung von Messläufen

2.6 Benutzbarkeit der GUI

SK 11 Die Funktionalität Aktionen per Drag-and-Drop durchzuführen ist in der aktuellen Version der Anwendung nicht implementiert. Die Gründe hierfür sind die lange Einarbeitungszeit in ein externes Editor-Programm, welches ermöglicht Objekte über Drag-and-Drop zu bewegen. Dies würde das Auswählen eines Editor-Programms beinhalten, die Einarbeitungszeit, die Implementierung und Integration dieses Programms beinhalten. Durch ein zu spätes Festlegen auf ein Editor-Programm, *JHotDraw* hat die Zeit für die Einarbeitung, Implementierung und Integration gefehlt. Deswegen haben wir uns aus Zeitgründen dagegen entschieden ein Editor-Programm zu implementieren. Leider entfällt dadurch eine grundlegende Funktionalität unserer Anwendung, welche bereits fest vorgesehen war und die Benutzung der Anwendung vereinfachen und verbessern würde. Als Weiterentwicklung hat dieses Kriterium eine hohe Priorität.

SK 12 Die Anwendung enthält ein Hilfe-Fenster, welche dem Benutzer eine kurze Beschreibung der Funktionalität der Anwendung und Information über die Anwendung bieten. Die Informationen zu den GUI-Elementen lassen sich jedoch nur aus dem Hilfe Fenster-Text auslesen und nicht interaktiv über Bedienung der GUI-Elemente.

SK 13 Die Anwendung bietet dem Benutzer Information über Fehler-Rückmeldungen über ein Fehlerfenster. Vor Fehlverhalten wird nicht gewarnt sondern nur reaktionär auf Fehler reagiert. Die Implementierung des Vorwarnen vor Fehlverhalten hätte sich durch konstante Analyse der Benutzereingaben als zeitintensiv und komplex erwiesen und wurde deswegen nicht umgesetzt.

WK 6 Die Anwendung ist in deutscher Sprache. Das Sprach-Paket lässt sich in der jetzigen Version nicht ändern.

WK 7 Die Festlegung auf ein Konfigurationsfeld, welche eine Konfiguration in schriftlicher Form erstellen lässt führt dazu, dass es keine visuelle Repräsentation von Bausteinen und deren Ein- und Ausgänge gibt.

WK 8 Die Anwendung legt kein festes Farbschema fest und somit ist das Farbschema nicht anpassbar. Die Implementierung ist vorgesehen, aber war zeitlich nicht umsetzbar.

WK 9 Die Anwendung legt eine feste Schriftgröße fest, die sich durch die feste Implementierung der GUI-Tools festlegt und ist somit nicht änderbar. Die Implementierung ist vorgesehen, aber nicht umgesetzt.

2.7 Abgrenzungskriterien

3 Umsetzung des Entwurfs

Während der Entwurfsphase wurden sowohl UML-Klassendiagramme als auch UML-Sequenzdiagramme erstellt. Zusammen mit der textuellen Beschreibungen der zu erstellenden Software-Elemente bildeten diese die Basis für die Produktion des Quellcodes während der Implementierungsphase.

In aller Regel lassen sich abstrakte Entwurfsinhalte während der Implementierung nicht in allen Details exakt umsetzen, was verschiedene Gründe haben kann. Bisweilen entpuppt sich auch eine andere Umsetzung als vorteilhafter. Die folgenden Abschnitte halten für jedes Softwaremodul die Abweichungen der Implementierung gegenüber den im Entwurf beschriebenen Strukturen fest. Des Weiteren enthalten sie die Gründe für diese Abweichungen.

3.1 Model

3.1.1 Ersetzen des Entwurfsmuster Erbauer durch eine Zuständigkeitskette

Das Paket „Model.BuildingBlockBuilder“ im Entwurf wurde durch das Paket „model.block“ ersetzt. Das dort verwendete Entwurfsmuster Erbauer erfüllte nicht die notwendige Anforderung, dass der Benutzer leicht eigene Versionen von Bausteinen in die Anwendung einfügen konnte. Darum wurde der Erbauer durch eine Zuständigkeitskette ersetzt. Hier gibt es keine Methode für jeden Baustein im Director, sondern es gibt nur eine Anzahl von Bearbeitern, die einen Block eines Types erstellen. Wenn also der Benutzer eine eigene Transformation erstellen will, kann er die .yaml Datei einer bereits vorhandenen Transformation kopieren und einige Parameter (außer Typ und subtyp) verändern. Die resultierende Transformation wird dann von der Anwendung als eine erkannt und kann dann auch dort verwendet werden. Dadurch entfallen alle folgenden Klassen des Entwurfs:

Builder

TransformationBuilder

RepresentationBuilder

XYRepresentationBuilder

TableRepresentationBuilder

SensorBuilder

VirtualSensorBuilder

PhysicalSensorBuilder

SnakeYamlParser

java.util.hashmap

sowie all diese öffentlichen Methoden in der Director Klasse:

createSensorFromYaml

constructTransformation

`constructXYRepresentation`

`constructNTimeRepresentation`

`constructDS18B20TemperatureSensor`

`constructBMPx80PressureSensor`

`constructINA219CurrentAndVoltageSensor`

`constructMMA8451Accelerometer`

`constructTransformation`

Statt dessen wurden folgende Klassen hinzugefügt:

`GeneralBlockKvProcessor`

`KvProcessor`

`SensorKvProcessor`

`PhysicalSensorKvProcessor`

`VirtualSensorKvProcessor`

`RepresentationKvProcessor`

`TableRepresentationKvProcessor`

`XYRepresentationKvProcessor`

`TransformationKvProcessor`

und die Methode `constructBuildingBlock` zum Director und zu jedem Bearbeiten die Methode `processKvPair` hinzugefügt. Dabei unterscheiden sich die Methoden der einzelnen Bearbeitern zwar nicht im Namen, aber in ihrer Funktion. Jeder Bearbeiter leitet entweder die Anfrage weiter oder erstellt einen Blocktyp und gibt ihn zurück.

3.2 View

3.2.1 GUI-Paket

MainWindow Die zentrale Instanz des GUI-Paketes, die Klasse MainWindow wurde durch Methoden zum Starten der Anwendung erweitert und fungiert somit als Initialisierungs-Modul unserer Anwendung. Dadurch wurde die Klasse um Attribute erweitert, welche für das Starten der Anwendung notwendig sind. Durch die Entscheidung für Swing als Framework-Tool, erbt die Klasse von der Klasse JFrame. Dadurch muss die Klasse durch Methoden zur Erzeugung des Anwendungsfenster erweitert werden. Ebenfalls werden alle anderen GUI-Objekte in dieser Klasse zu dem Anwendungsfenster hinzugefügt.

Erweitert wurde das GUI-Paket um die Klassen DataVisualisation, ConfigurationEditor und Editor.

ConfigurationEditor Die Klasse ConfigurationEditor stellt das Textfeld zum Schreiben von Messkonfigurationen dar. Durch die Entscheidung gegen ein Drag-and-Drop Editor stellt diese Klasse nun die Instanz dar, mit welcher der Benutzer eine Messkonfiguration entwirft.

DataVisualisation Die Klasse DataVisualisation stellt das Textfeld dar, in welcher Messdaten eines Messlaufs präsentiert werden.

Editor Die Klasse Editor benutzt das Framework JHotDraw, um einen Drag-and-Drop-Editor zu erstellen. In der jetzigen Version der Anwendung wird dieser Editor nicht eingebunden und nicht benutzt und liefert ebenfalls nicht die gewünschte Funktionalität. Für eine Weiterentwicklung ist der Editor jedoch ein grundlegender Baustein, um die Anwendung benutzerfreundlicher zu machen.

3.2.2 Menu-Paket

Das Menü-Paket wurde durch diverse Veränderungen am Entwurf, wie durch Bereitstellung von Framework- Klassen und Methoden vereinfacht, in dem mehrere Klassen zusammengefasst wurden.

PrototypeField In unserer Anwendung umfasst die Klasse PrototypeField ebenfalls die Klassen SensorBlockField, TransformationBlockField und RepresentationBlockField. Die Einzelnen Menüs der Bausteine werden als JTabbedPane, also als Tab-Fenster in dem Übermenü PrototypeField und enthalten die am Anfang initiali-

sierten Bausteine. Ebenfalls lassen sich hier über Drücken des "BearbeitenKnopfes" die Eigenschaften der Bausteine einsehen.

Dadurch entfallen in unserer Implementierung die Klassen:

SensorBlockField Enthalten in Klasse PrototypeField und somit entfällt eine separate Implementierung.

TransformationBlockField Enthalten in Klasse PrototypeField und somit entfällt eine separate Implementierung.

RepresentationBlockField Enthalten in Klasse PrototypeField und somit entfällt eine separate Implementierung.

FieldHandler Eine separate Implementierung von Handler-Klassen entfällt durch das Swing-Framework durch die Benutzung von vordefinierten Listener-Klassen.

Die Funktionalität der Buttons wurde in der Implementierung beibehalten, jedoch die Klassen zusammengefasst, da das Framework hierfür bereits Funktionalität vorgibt.

ButtonField Die Klasse ButtonField umfasst zusammen mit der Klasse MeasurementButtonField in unserer Implementierung Teile des Button-Paket und die Funktionalität. Die Klassen Button und deren Unterklassen sind somit als JButton-Attribut in dem ButtonField enthalten.

MeasurementButtonField Die Klasse MeasurementButtonField enthält die anderen Teile des Button-Pakets, welche den Messlauf beeinflussen, d.h Pause, Resume, Reset und SaveMeasurementData. Ebenfalls sind diese Knöpfe als Attribut der MeasurementButtonKlasse wieder zu finden.

Dadurch entfällt das komplette Button-Paket.

3.2.3 Configuration-Paket

Durch die Entscheidung Konfigurationen schriftlich aufzubauen und keinen Drag-and-Drop-Editor zu benutzen entfallen die visuellen Repräsentationen der Bausteine, der Verbindung, der Ein- und Ausgänge, sowie die zugehörigen Handler. Damit erfüllt das Configuration-Paket in der Implementierungsversion keine Funktionalität. Für eine Weiterentwicklung der Anwendung mit einem Editor, welcher Drag-and-Drop unterstützt ist das Paket wieder eine zentrale Instanz der Anwendung.

3.2.4 BlockProperties-Paket

Durch die Verwendung des Swing-Framework entfallen ebenfalls in diesem Paket die Handler-Interfaces, da das Framework diese Funktionalität in Klassen und Methoden liefert. Daher entfallen in der Implementierung die Interfaces BuildingBlockPropertiesHandler, TransBlockPropertiesHandler und ReprBlockPropertiesHandler. Für die Darstellung der Block-Eigenschaften sind die im Entwurf genannten Klassen implementiert. Durch die Darstellung der Eigenschaften als JFrame- Fenster werden die Klassen jedoch um die nötigen Attribute und Methoden ergänzt, um diese Funktionalität zu bieten.

3.2.5 Exception-Paket

Das Exception-Paket wurde fast vollständig aus dem Entwurf übernommen. Einzelne Methoden entfallen durch das Framework, wie z.B closeAll() in Klasse ExceptionWindowManager oder close() in ExceptionWindow. Ebenfalls wurde in der Klasse ConnectionExceptionWindow die Methode changeWireColor() entfernt, da durch unsere Entscheidung das Projekt von visuellen Komponenten zu kapseln, die Verbindung nicht mehr zu einer Instanz der Klasse Wire zugeordnet werden kann.

3.2.6 HelpAndOption-Paket

Das HelpAndOption-Paket wurde insofern verändert, dass die Interfaces HelpWindowHandler und OptionsWindowHandler aufgrund der Framework-Entscheidung entfallen. Die Klassen HelpWindow und OptionsWindow wurden durch Attribute und Methoden zur Anpassen an das Framework erweitert, damit die Fenster als unabhängige Instanzen angezeigt werden. Das OptionWindow wurde durch weitere Optionen erweitert, d.h System-Optionen, Fenster-Optionen, Messlauf-Optionen und Raspberry-Pi-Optionen.

3.2.7 Model-Interface-Paket

Das Interface ViewDirectoryInterface wurde nach Entwurfsvorgaben implementiert. Einzelne Methoden finden in der jetzigen Version der Anwendung keine Anwendung, sind aber für eine mögliche Weiterentwicklung notwendig, um bestimmte Funktionalitäten zu bieten.

3.2.8 Controller-Interface-Paket

Das Contoller-Interface-Paket wurde nach Entwurfsvorgaben implementiert. Einzelne Methoden aller Interfaces finden in der jetzigen Version der Anwendung keine Anwendung, sind aber für eine mögliche Weiterentwicklung notwendig, um bestimmte Funktionalitäten zu bieten. Bei einer Weiterentwicklung um ein Konfiguration-Editor, welcher Drag-And-Drop unterstützt, werden die Interfaces `IBlockAction` und `IConnectionAction` eine wesentliche Rolle zur Verbindung der zwei Module spielen.

3.3 Controller

3.4 Backend

3.4.1 SSH-Verbindung

Für die Kommunikation mit dem Raspberry Pi wird eine SSH-Verbindung verwendet. So können über das lokale Netzwerk Python-Skripte auf dem Raspberry Pi ausgeführt und Dateien kopiert werden.

Die SSH-Verbindung ist mithilfe der Bibliothek *SSHJ*¹ implementiert. Infolge dieser Technologieentscheidung sind folgende Klassen überflüssig geworden: `SystemProcessCommandLine`, `SshCommandGetSensorIds`, `SshCommandCopyFromPi`, `SshCommandCopyToPi`.

Die Aufgaben dieser Klassen werden stattdessen durch ihre vorgesehenen Oberklassen (`CommandGetSensorIds`, `CommandCopyFromPi`, `CommandCopyTo`) und direkt durch `SshToPi` erfüllt.

Nach außen ist der Zugriff auf das Backend über die Schnittstellen `IAccessToSensorInfo` und `IAccessToMeasurementRun` und die implementierenden Klassen `SensorInfoAgent` und `MRunAgent`, sowie die Klasse `PickupPointForBackendAgents` möglich. Diese wurden wie im Entwurf vorgesehen umgesetzt.

3.4.2 Simulierte Verbindung

Anstelle einer echten SSH-Verbindung kann das Backend auch simuliert werden. Dazu wurden die Klassen `ComToFile` und `FileCommandFactory` eingeführt. Es können dadurch Messdaten aus einer zuvor erstellten Datei ausgelesen werden.

¹<https://github.com/hierynomus/sshj>

Die Simulation des Backends ist nur eingeschränkt funktionstüchtig, da die Verbindung mit einem echten Gerät in der Entwicklung eine deutlich höhere Priorität hatte.

3.4.3 Nebenläufigkeit

Das Auslesen von Daten läuft parallelisiert ab. Dabei wird für jeden Sensor ein eigener Thread aufgebaut. (Nicht, wie im Entwurf angedeutet, für jeden Kanal.)

Jeder dieser Threads verwendet eine Instanz der Klasse `MeasurementRunnable`, welche die Klassen `CommandMRun` und `SshCommandMRun` aus dem Entwurf ersetzt. Über diese *Runnables* wird der Messlauf gesteuert (unter anderem pausiert und angehalten). Die Klasse `MRunThread` ist dabei überflüssig geworden.

3.5 Cache

3.6 File-Service

CsvService Die Klasse `CsvService` wurde in der Implementierung durch die Methoden zum Umbenennen und Exportieren von `CsvDateien` erweitern. Die im Entwurf gewollte Funktionalität wird durch die Implementierung umgesetzt.

PngService Die Funktionalität der Klasse `PngService` wurde wie im Entwurf beschrieben umgesetzt. Sie wird jedoch durch die Umorganisation der Projektfunktionalität und Projektstruktur nicht verwendet.

YamlService Die Klasse `YamlService` wurde nach der Entwurfsstruktur umgesetzt. Durch Anpassung an `SnakeYaml` wurden jedoch einige Parameter und Argumente verändert.

4 Realer Implementierungsablauf

Dieser Abschnitt führt auf, inwieweit der tatsächliche zeitliche Implementierungsablauf vom geplanten Ablauf abgewichen ist, und beschreibt die Ursachen und Gründe für diese Abweichungen. Abhängigkeiten zwischen den Implementierungsschritten und kritische Pfade stehen hierbei besonders im Fokus.

Von Abweichungen betroffene Softwareelemente werden nicht im Einzelnen aufgeführt, sondern es werden lediglich in Bezug auf die Abweichungsgründe die Gruppen der betroffenen Softwareelemente benannt.

5 Anhang

Woche	27	28								29								30								31								32							
Tag	07. Jul	08. Jul	09. Jul	10. Jul	11. Jul	12. Jul	13. Jul	14. Jul	15. Jul	16. Jul	17. Jul	18. Jul	19. Jul	20. Jul	21. Jul	22. Jul	23. Jul	24. Jul	25. Jul	26. Jul	27. Jul	28. Jul	29. Jul	30. Jul	31. Jul	01. Aug.	02. Aug.	03. Aug.	04. Aug.	05. Aug.	06. Aug.	07. Aug.	08. Aug.	09. Aug.	10. Aug.	11. Aug.					
Module																																									
Abstract																																									
Controller																																									
Interface																																									
ModelInformation																																									
MeasurementStatus																																									
View Controller Interface																																									
RobotAction																																									
BlockAction																																									
ConnectionAction																																									
Command Pattern																																									
CommandManager																																									
Command																																									
AddBlockToConfigCommand																																									
ModifyBlockConfigCommand																																									
RemoveBlockFromConfigCommand																																									
ExportBlockPropertiesCommand																																									
ImportBlockPropertiesCommand																																									
SaveConfigCommand																																									
LoadConfigCommand																																									
CreateChannelConnectionCommand																																									
ModifyChannelConnectionCommand																																									
StartRunCommand																																									
StopRunCommand																																									
ResumeRunCommand																																									
FilterService																																									
Services																																									
Transformance																																									
OnService																																									
OffService																																									
GUI																																									
Menus																																									
PrototypeField																																									
TransformationBlockField																																									
RepresentationBlockField																																									
FieldHandler																																									
Configuration																																									
ConfigurationFile																																									
BuildingBlockEventManager																																									
BuildingBlockView																																									
SensorBlockView																																									
TransformationBlockView																																									
RepresentationBlockView																																									
WidgetContainer																																									
AddWindowAndDragHandler																																									
RemoveWindowAndDragHandler																																									
ChannelConnector																																									
ChannelDescriptor																																									
Wire																																									
Building Block Properties																																									
BuildingBlockProperties																																									
BuildingBlockParametersHandler																																									
SensorBlockProperties																																									
TransformationBlockProperties																																									
LineBlockPropertiesHandler																																									
RepresentationBlockProperties																																									
HeadBlockPropertiesHandler																																									
Button																																									
Button																																									
ButtonPointer																																									
Exception																																									
ExceptionWindowManager																																									
ExceptionWindow																																									
BuildingBlockExceptionWindow																																									
ConnectionExceptionWindow																																									
GeneralExceptionWindow																																									
WidgetOption																																									
OptionWindow																																									
WidgetWindow																																									
WidgetWindowHandler																																									
OptionWindowHandler																																									
FacadeController View																																									
PackagePageControllerView																																									
RobotAction																																									
BlockAction																																									
ConnectionAction																																									
FacadeMapView																																									
FacadeObserverAndView																																									
Core																																									
MeasurementUnit																																									
MeasurementFunction																																									
MeasurementConfiguration																																									
BuildingBlock																																									
WidgetMessage																																									
WidgetRepresentation																																									
BuildingBlockStructure																																									
Transformation Logic																																									
Function																																									
TransformationLogic																																									
Representation Logic																																									
Representation																																									
WidgetRepresentation																																									
WidgetVisualization																																									
Sensor Logic																																									
Sensor																																									
PhysicalSensor																																									
VirtualSensor																																									
Channel Logic																																									
Channel																																									
ChannelState																																									
Connected																																									
NotConnected																																									
ValueReady																																									
InChannel																																									
OutChannel																																									
Building Block Builder																																									
Director																																									
Builder																																									
IntegerParameter																																									
SensorBuilder																																									
PhysicalSensorBuilder																																									
VirtualSensorBuilder																																									
TransformationBuilder																																									
RepresentationBuilder																																									
WidgetRepresentationBuilder																																									
LineRepresentationBuilder																																									
Facade Controller View																																									
PackagePageViewElements																																									
OptionInterface																																									
WidgetDataBuilderInterface																																									
ExceptionInterface																																									
MeasurementDataInterface																																									
WidgetHandler																																									
WidgetInfo																																									
WidgetManager																																									
Backend																																									
Measurement Logic																																									
AcquireDataUnitInfo																																									
ProcessDataUnit																																									
MeasurementState																																									
MeasurementAgent																																									
SensorInfoAgent																																									
PushEventForAgentBasedOnSub																																									
SubscribeForSub																																									
SubscribeChannel																																									
Command Factory																																									
CmdTaps																																									
CmdTurn																																									
CommandFactory																																									
GetCommandFactory																																									
CommandGetSensors																																									
CommandCopyCopyLeft																																									
CommandCopyCopyRight																																									
CommandCopyCopyFront																																									
CommandCopyCopyBack																																									
CommandMoveAndUp																																									
CommandMoveAndDown																																									
CommandMoveAndLeft																																									
CommandMoveAndRight																																									
Write																																									
Read																																									
Thread																																									
Variables																																									
Cache																																									
Cache Logic																																									
Mutex Timing																																									
Timing																																									
Cache																																									
BufferForOneSensorChannel																																									
TimesToWaitForPair																																									
ConnectionTerminatedAction																																									
TimeoutAction																																									
ErrorCalculation																																									
OutputCompleteAction																																									
ChannelDataPairAction																																									
ErrorCalculationChannel																																									
ChannelDataPairAction																																									
MutexCleaner																																									
MutexData																																									
TimeTask																																									

6 Glossar

JHotDraw JHotDraw ist ein Open-Source, Java-basiertes Framework zur Erstellung von grafischen Editoren. Durch die einfachere Unterstützung von Drag-and-Dop, als komplexere Frameworks eine gute Alternative..

todo todo.