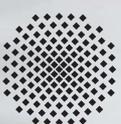
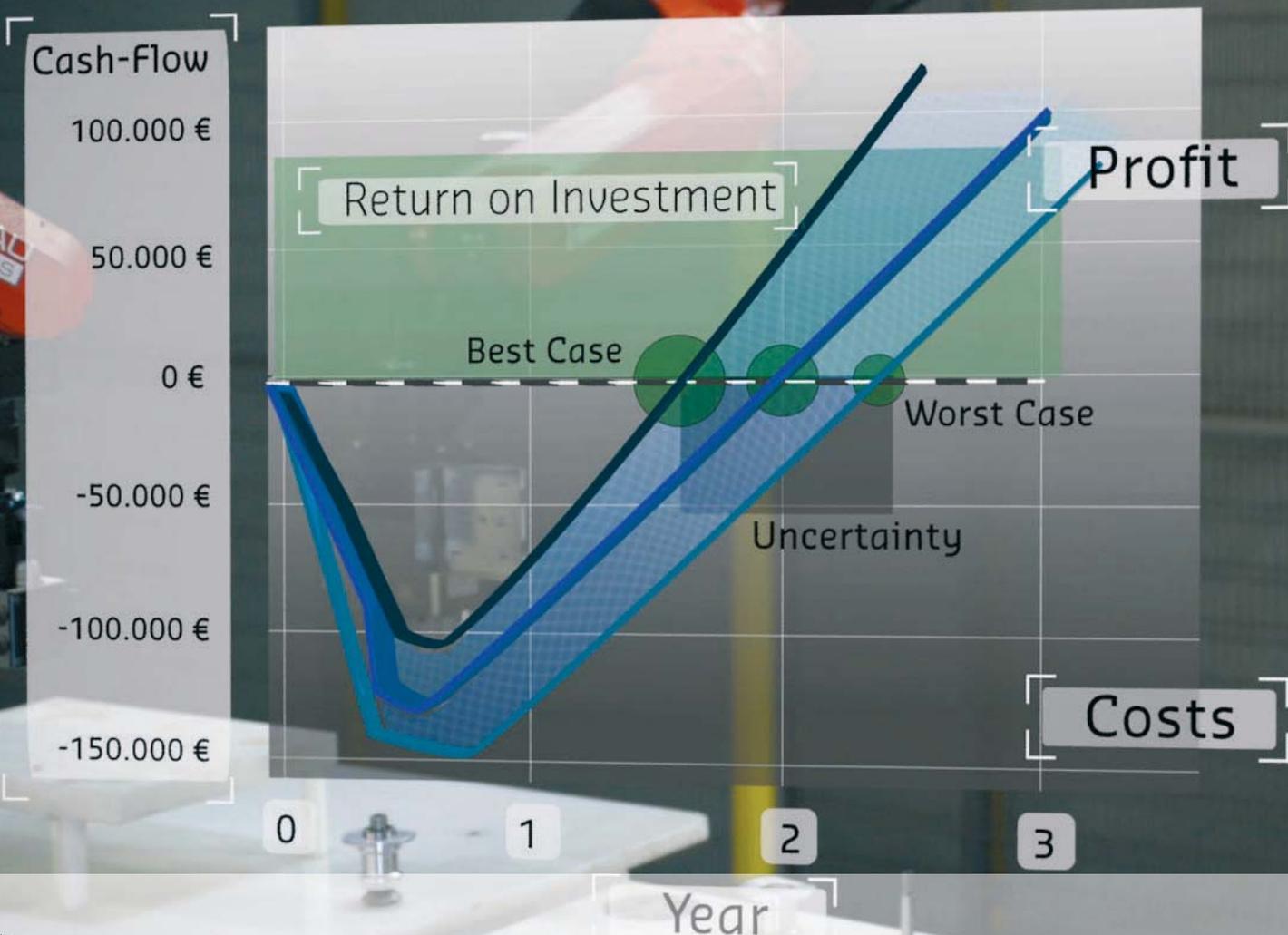


THOMAS DIETZ

Knowledge-based cost-benefit analysis of robotics for SME-like manufacturing



Herausgeber:

Univ.-Prof. Dr.-Ing. Thomas Bauernhansl

Univ.-Prof. Dr.-Ing. Kai Peter Birke

Univ.-Prof. Dr.-Ing. Marco Huber

Univ.-Prof. Dr.-Ing. Oliver Riedel

Univ.-Prof. Dr.-Ing. Dipl.-Kfm. Alexander Sauer

Univ.-Prof. Dr.-Ing. Dr. h.c. mult. Alexander Verl

Univ.-Prof. a.D. Dr.-Ing. Prof. E.h. Dr.-Ing. E.h. Dr. h.c. mult. Engelbert Westkämper

Thomas Dietz

**Knowledge-based cost-benefit analysis of robotics
for SME-like manufacturing**

Kontaktadresse:

Fraunhofer-Institut für Produktionstechnik und Automatisierung IPA, Stuttgart
Nobelstraße 12, 70569 Stuttgart
Telefon 07 11/9 70-11 01
info@ipa.fraunhofer.de; www.ipa.fraunhofer.de

STUTTARTER BEITRÄGE ZUR PRODUKTIONSFORSCHUNG

Herausgeber:

Univ.-Prof. Dr.-Ing. Thomas Bauernhansl^{1,2}

Univ.-Prof. Dr.-Ing. Kai Peter Birke^{1,4}

Univ.-Prof. Dr.-Ing. Marco Huber^{1,2}

Univ.-Prof. Dr.-Ing. Oliver Riedel³

Univ.-Prof. Dr.-Ing. Dipl.-Kfm. Alexander Sauer^{1,5}

Univ.-Prof. Dr.-Ing. Dr. h.c. mult. Alexander Verl³

Univ.-Prof. a. D. Dr.-Ing. Prof. E.h. Dr.-Ing. E.h. Dr. h.c. mult. Engelbert Westkämper^{1,2}

¹Fraunhofer-Institut für Produktionstechnik und Automatisierung IPA, Stuttgart

²Institut für Industrielle Fertigung und Fabrikbetrieb (IFF) der Universität Stuttgart

³Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW) der Universität Stuttgart

⁴Institut für Photovoltaik (IPV) der Universität Stuttgart

⁵Institut für Energieeffizienz in der Produktion (EEP) der Universität Stuttgart

Titelbild: © Fraunhofer IPA

Bibliografische Information der Deutschen Nationalbibliothek

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über www.dnb.de abrufbar.

ISSN: 2195-2892

ISBN (Print): 978-3-8396-1437-2

D 93

Zugl.: Stuttgart, Univ., Diss., 2018

Druck: Mediendienstleistungen des Fraunhofer-Informationszentrum Raum und Bau IRB, Stuttgart
Für den Druck des Buches wurde chlor- und säurefreies Papier verwendet.

© by **FRAUNHOFER VERLAG**, 2019

Fraunhofer-Informationszentrum Raum und Bau IRB

Postfach 80 04 69, 70504 Stuttgart

Nobelstraße 12, 70569 Stuttgart

Telefon 07 11 9 70-25 00

Telefax 07 11 9 70-25 08

E-Mail verlag@fraunhofer.de

URL <http://verlag.fraunhofer.de>

Alle Rechte vorbehalten

Dieses Werk ist einschließlich aller seiner Teile urheberrechtlich geschützt. Jede Verwertung, die über die engen Grenzen des Urheberrechtsgesetzes hinausgeht, ist ohne schriftliche Zustimmung des Verlages unzulässig und strafbar. Dies gilt insbesondere für Vervielfältigungen, Übersetzungen, Mikroverfilmungen sowie die Speicherung in elektronischen Systemen.

Die Wiedergabe von Warenbezeichnungen und Handelsnamen in diesem Buch berechtigt nicht zu der Annahme, dass solche Bezeichnungen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und deshalb von jedermann benutzt werden dürften. Soweit in diesem Werk direkt oder indirekt auf Gesetze, Vorschriften oder Richtlinien (z.B. DIN, VDI) Bezug genommen oder aus ihnen zitiert worden ist, kann der Verlag keine Gewähr für Richtigkeit, Vollständigkeit oder Aktualität übernehmen.

Knowledge-based cost-benefit analysis of robotics for SME-like manufacturing

Von der Fakultät Konstruktions-, Produktions- und
Fahrzeugtechnik der Universität Stuttgart zur Erlangung der
Würde eines Doktor-Ingenieurs (Dr.-Ing.) genehmigte Abhandlung

vorgelegt von
Thomas Dietz
aus Karlsruhe

Hauptberichter: Prof. Dr.-Ing. Dr. h. c. mult. Alexander Verl
Mitberichter: Assoc. Prof. Dr.-Ing. Klas Nilsson
Mitberichter: Prof. Dr.-Ing. Carin Andersson

Tag der mündlichen Prüfung: 12. September 2018

Institut für Steuerungstechnik der Werkzeugmaschinen und
Fertigungseinrichtungen (ISW) der Universität Stuttgart

2018

Acknowledgement

This thesis results from my work as research fellow and team manager at the Fraunhofer Institute for Manufacturing Engineering and Automation IPA. I thank Professor Dr.-Ing. Dr. h. c. mult. Alexander Verl for his supervision of my scientific work and his support and endorsement in writing this thesis. I thank Professor Dr.-Ing. Klas Nilsson and Professor Dr.-Ing. Carin Andersson for dedicating so much time to carefully co-reviewing my thesis. They provided very important impulses for my work and helped me far more than I expected.

I am grateful to all my colleagues at the department of Robot and Assistive Systems for interesting expert discussions, support and for the creative work environment at Fraunhofer IPA. A special thank goes to Junior-Prof. PD Dr.-Ing. Andreas Pott for his scientific mentoring, endorsement, constant reviewing of results and persistent advise during the work on this thesis and at Fraunhofer IPA in general. I am thankful to my boss Dipl.-Ing. Martin Hägele for his guidance, trust and for creating an environment for my personal growth. I thank my teammates M.Sc. Julian Diaz-Posada, Dr.-Ing. Milad Geravand, Dipl.-Ing. Alexander Kuss, Dr.-Ing. Susanne Oberer-Treitz and Dr.-Ing. Ulrich Schneider. I especially thank Luzia Schuhmacher for carefully reviewing this thesis and providing valuable comments for its improvement. Heide Kreuzburg provided invaluable support with administrative and organizational tasks for which I am extremely grateful.

I want to express my thanks to the SMERobotics consortium in which large parts of the research work of this thesis were carried out and the members of ARENA2036. In particular, I am grateful to Dr.-Ing. Max Hossfeld for his constant support and advice during the finalization of this thesis.

I am grateful to my parents for their ongoing support and personal advice throughout my work. I thank my wife Hyunjin Cho for her patience despite long nights and weekends of absence that were required for writing this thesis. Without her encouragement and loving support it would not have been possible to complete this thesis.

Stuttgart, September 2018

Thomas Dietz

Short Summary

Robot systems promise high potential for improvement of production processes in small and medium-sized enterprises (SME). Often, reliable data for the assessment of costs and benefits of robot systems is missing, because they are special machinery and because the required information is distributed among component manufacturer, system integrator and end-user. The problem of cost-benefit assessment of industrial robot systems in the face of uncertain and incomplete information is of high relevance for unlocking the potential of robotics in SME-like production environments.

In order to solve this problem, methods from business economics and from knowledge management in robotics are combined in a cost-benefit model for robot systems. This cost-benefit model builds on the established PPR-approach, which separates the concerns of product, process and resources. The PPR-approach is expanded with specific cost-relevant aspects. Domain specific costing models, borrowing from activity-based costing (ABC), for product, process, resources and cost information are proposed. These models and the meta-information required for their usage are collected in a knowledge base.

The input for the cost-benefit assessment is a description of the current planning status in AutomationML. This information is matched to the information in the knowledge base through reasoning. This allows to automatically build an overall cost-benefit model for the robot system. Missing or inaccurate information causes uncertainties. These uncertainties are modeled using interval arithmetics. The overall cost-benefit model is able to process interval numbers and hence to compute on the uncertain information. The interval representation offers an intuitive description of uncertainties in costs and benefits. An amortization graph allows to intuitively represent the results of the cost-benefit assessment for the entire life-cycle.

A test implementation of the cost-benefit assessment method allows an evaluation of its results for two realistic use cases. The resulting costs, benefits, and project planning are plausible and in agreement with real experience. Concluding, the cost-benefit assessment method has high potential to improve decision making in the development of robot systems.

Kurzzinhalt

Robotersysteme versprechen hohes Potential zur Verbesserung von Produktionsprozessen in kleinen und mittleren Unternehmen (KMU). Jedoch fehlen häufig die zur Bewertung von Kosten und Nutzen erforderlichen Informationen, da Robotersysteme Sondermaschinen sind und da diese Informationen auf Komponentenhersteller, Systemintegrator und Endnutzer verteilt sind. Die Problemstellung der Bewertung von Kosten und Nutzen von Robotersystemen unter Berücksichtigung von unsicheren und unvollständigen Informationen ist daher relevant, um das Potential der Robotik für KMU-Produktionsszenarien zu erschließen.

Um diese Problemstellung zu lösen, werden Methoden aus den Wirtschaftswissenschaften und aus dem Wissensmanagement in der Robotik in einem Kosten-Nutzenmodell kombiniert. Dieses Modell basiert auf dem bekannten PPR-Ansatz zur Trennung der Aspekte von Produkt, Prozess und Ressourcen, der um kostenrelevante Aspekte erweitert wird. Spezifische, an die Prozesskostenrechnung angelehnte Kostenmodelle werden entwickelt. Diese Modelle und ihre erforderlichen Metainformationen werden in einer Wissensdatenbank gesammelt.

Eingang der Kostenbewertung ist eine Beschreibung des aktuellen Planungsstands in AutomationML. Durch maschinelles Schlussfolgern werden diese Informationen mit den in der Wissensdatenbank hinterlegten Informationen abgeglichen. Hierdurch kann automatisiert ein Modell für Kosten und Nutzen des Robotersystems erstellt werden. Während des Abgleichprozesses werden fehlende oder unsichere Informationen mit Methoden der Intervallrechnung modelliert. Das resultierende Kostenmodell kann diese Informationen verarbeiten und bietet dadurch eine intuitiv verständliche Beschreibung der Unsicherheiten von Kosten und Nutzen des Robotersystems. Zur Darstellung der Ergebnisse wurde ein Amortisationsgraph entwickelt, der einen Überblick über den gesamten Lebenszyklus erlaubt.

Die Testimplementierung der entwickelten Methode erlaubt deren Evaluierung für zwei Anwendungsfälle. Die durch das Kostenbewertungssystem erzeugten Abschätzungen von Aufwänden, Nutzen und Projektplanung sind plausibel und in Übereinstimmung mit Erfahrungen aus der Praxis. Zusammenfassend hat das Verfahren zur Bewertung von Kosten und Nutzen hohes Potential, die Entscheidungsfindung bei der Entwicklung von Robotersystemen zu verbessern.

Contents

Abbreviations and symbols	ix
List of figures	xiv
List of tables	xvii
1 Introduction	1
1.1 Motivation	2
1.2 Objectives	4
1.3 Scope and applied method	5
2 State of the art and background	8
2.1 Accounting and capital budgeting	9
2.1.1 Management accounting	10
2.1.2 Capital expenditure budgeting	15
2.1.3 Cost considerations for flexible manufacturing systems . .	22
2.2 Uncertainties and related arithmetics	25
2.2.1 Classification of uncertainties	25
2.2.2 Methodical frameworks for analyzing systems under the influence of uncertainties	27
2.2.3 Mathematical descriptions of uncertainties	31
2.2.4 Conclusions on modeling of uncertainties	37
2.3 Cost modeling in engineering design	38
2.3.1 Costing methods in engineering and product development .	38
2.3.2 Estimation of software development efforts and times . . .	40
2.3.3 Development process for robot systems	45
2.4 Modeling of manufacturing systems	46
2.4.1 Information models used in plant engineering	47
2.4.2 Modeling of work efforts and resource consumption in production	48
2.4.3 Knowledge management in robotics	50
2.5 Conclusions from the state of the art	53
2.5.1 Methods from accounting and capital budgeting	53

2.5.2	Handling of uncertainties	54
2.5.3	Existing engineering models	55
2.5.4	Knowledge management	55
3	Problem analysis	57
3.1	Example applications for evaluation of the developed model . . .	57
3.1.1	Robotic GMAW welding	58
3.1.2	Machine tending robot for work piece handling	59
3.2	Scope of work and related terms	61
3.3	Analysis of the life-cycle of robot systems	62
3.3.1	Life-cycle model of the robot system	63
3.3.2	Involvement of stakeholders in the life-cycle	67
3.3.3	Cost and flexibility of robot systems	70
3.3.4	Interaction of system integrator and end-user	72
3.3.5	Alignment of system integrator and end-user interests in the realization of robot systems	74
3.4	Problem statement and breakdown	76
3.4.1	Usage scenarios of cost-benefit assessment and problem formulation	76
3.4.2	Breakdown of problem	79
4	Conception of the cost-benefit model	85
4.1	Overall costing system concept	85
4.1.1	Approaches to cost computation of robot systems	85
4.1.2	Assessment and selection of conceptual approach	88
4.2	Uncertainty-aware cost-benefit model	90
4.2.1	Types of cost	90
4.2.2	Structure of the cost-benefit model	91
4.2.3	Modeling of activities	102
4.2.4	Configuration of the cost model	111
4.2.5	Handling of uncertainty in the cost-benefit model	115
4.2.6	Scheduling and computation of the cost model	124
4.3	Knowledge management of cost data	128
4.3.1	Machine-specific knowledge	130
4.3.2	Cost ontology for machine independent knowledge	132
4.3.3	Handling uncertainties in knowledge management	143
4.4	Conclusion of concept	154
5	Activity models	155
5.1	General activity models in robotics	155
5.1.1	Product models for robot system realization	156

5.1.2	Process models for robot system realization	157
5.1.3	Product models for robot system usage	167
5.1.4	Process models for robot system usage	167
5.1.5	Resource models	169
5.1.6	Cost-pool models	173
5.2	Application specific activity models	175
5.2.1	Robotic GMAW welding	175
5.2.2	Machine tending robot for work piece handling	181
6	Implementation and parameterization	184
6.1	Implementation of cost modeling	184
6.1.1	Implementation of cost-benefit model	186
6.1.2	Run-time representation of the cost-benefit model	195
6.1.3	User interface	202
6.2	Knowledge management	204
6.2.1	AutomationML data management	205
6.2.2	Knowledge base	206
6.2.3	Knowledge interaction	208
7	Evaluation	210
7.1	Analysis of model behavior	210
7.1.1	Typical amortization behavior	210
7.1.2	Distribution of risk and uncertainties among stakeholders	222
7.1.3	Cost distribution for system realization	223
7.1.4	Task scheduling and project schedule	225
7.2	Model results and discussion for the handling use case	227
7.2.1	Amortization behavior	227
7.2.2	Cost distribution	228
7.2.3	Project schedule	230
7.3	Evaluation of the developed method for cost-benefit assessment	231
8	Conclusions and outlook	235
8.1	Contributions	237
8.2	Further research	240
	Appendix	243
	Bibliography	255

Abbreviations and symbols

Abbreviations

Abbreviation	Meaning
ABC	Activity-based costing
ANPV	Adjusted net present value
CAD	Computer aided design
CAx	Computer aided x
COCOMO	Constructive cost model
COTS	Commercial off-the-shelve components
DCF	Discounted cash flow
DLL	Dynamic-link library
DSI	Delivered source instructions
ERP	Enterprise resource planning
EVA	Economic value added
EWTF	European welding foundation
FAT	Factory acceptance test
FMS	Flexible manufacturing system
GMAW	Gas metal arc welding
GPK	Grenzplankostenrechnung, marginal planned cost accounting
IDE	Integrated development environment
inf	Infimum
IP	International protection marking
IRR	Internal rate of return
KPI	Key performance indicator
LCC	Life-cycle cost
Lvl	Level
mod	Modulus function
MOST	Maynard operation sequence technique
MTM	Methods-time measurement
MTM-UAS	MTM universelles Analysiersystem

Abbreviations and symbols

Abbreviation	Meaning
NPV	Net present value
OAT	One factor at a time
OLP	Offline programming
OWA	Open world assumption
OWL	Web ontology language
PLC	Programmable logic controller
PLM	Product life-cycle management
PMTS	Predetermined motion time systems
PPR	Product process resource
PPRC	Product process resource cost-pool
RCA	Resource consumption accounting
RDF	Resource description format
RDFS	Resource description framework schema
ROI	Return on investment
RTM	Robot time measurement
SLIM	Software life-cycle management
SLOC	Source lines of code
SME	Small and medium-sized enterprise
SPARQL	SPARQL protocol and RDF query language
STEP	Standard for the exchange of product model data
sup	Supremum
TCO	Total cost of ownership
TDABC	Time-driven activity-based costing
TRL	Technology readiness level
URI	Uniform resource identifier
VaR	Value at risk
W3C	World wide web consortium
WBS	Work breakdown structure
WPF	Windows presentation foundation
WPS	Welding procedure specification
XML	Extensible markup language
XP	Extreme programming

Symbols

Symbol	Unit	Meaning
a	m/s^2	Acceleration
\mathcal{A}	Depending on activity level	Activity level
\mathfrak{a}	-	Activity
\mathfrak{A}	-	Set of activities \mathfrak{a}
\mathcal{C}	Depending on driver	Cost driver
C	Currency units (e.g. €)	Cash flow (positive: cost)
c	Currency units per time unit	Cost rate
\mathcal{C}	1	Functional complexity
d	mm	Diameter
I	A	Electric current
F	Depending on quantity	Constant factor
f	Depending on scaling	Scaling factor
l	m	Position vector
L	m	Length
\mathcal{M}	-	Cost model
m	kg	Mass
m_L	kg/m	Mass per meter of length
o	%	Cost overhead in percent
\mathcal{P}	Depending on parameter	Model parameter
P	Set	Set of products
p	Set member	A manufactured product
\mathfrak{p}	$\mathfrak{p} \in \mathbb{N}$ from low priority ($\mathfrak{p} = 1$) to high priority ($\mathfrak{p} = 3$)	Priority of an activity \mathfrak{a}
\mathcal{P}	W	Power
q	%/100	Interest rate
r	$r \in \mathbb{N}$	Relative production volume
\mathcal{R}	1	Severity of requirements
S	m^2	Surface area
s	$s \in \mathbb{N}$	Lot size
t	s	Time duration
T	K	Temperature
U	V	Electric voltage
V	m^3	Volume
v	m/s	Speed
x	Depending on parameter	Value of a model parameter
X	Depending on quantity	Arbitrary quantity
ω	Depending on offset quantity	Offset

Subscripts

Subscript	Meaning
A	Activity
air	Pneumatic
ass	Assembly
Char	Characteristic
d	Time interval of one day
dev	Development
DSI	Delivered source instructions
E	Electric
ex	Executed
G	Electric grid
g	Gas
I	Time interval with unspecified duration
i	Index variable for summation
\mathcal{L}	Lot
L	Lot
l	Left
loc	Localization
M	Material
max	Maximum value
min	Minimum value
nex	Not executed
prec	Precondition
PS	Welding power source
r	Right
Ref	Reference
S	Seam
SLOC	Source lines of code
T	Technology
Th	Thermal
U	Unit
w	Time interval of one week
weld	Welding specific information
wo	Work
wp	Work piece

Prescripts

Prescript	Meaning
i	Level in the PPRC-model from product level ($i = 1$) to cost-pool ($i = 4$)

Indices

Index	Meaning
m	Number of cost drivers on level i
k	Number of models on level i
n	Number of activity levels for a certain model ${}_i\mathcal{M}_k$
r	Number of parameters for a certain model ${}_i\mathcal{M}_k$

List of Figures

1.1	Comparison of use of robot systems in mass production and small and medium lot size production.	2
1.2	Positioning of the scope of this thesis in terms of stakeholders and dimensions of cost and benefit.	6
2.1	Relevant knowledge domains for cost-benefit assessment of robot systems.	9
2.2	Overview of analyzed accounting methods.	10
3.1	Analyzed robot work cell for welding example.	59
3.2	Analyzed robot work cell for machine tending example.	60
3.3	Used life-cycle model for robot systems (adapted from VDMA 34160-06).	64
3.4	Interdependencies of cost model, involved stakeholders, and life-cycle of the robot system.	68
3.5	Comparison of cost aspects of manual and automated production.	71
3.6	Key stages of the realization process, their results and the used engineering tools in these stages	73
3.7	Relation of the addressed sub-problems.	80
4.1	Comparison of identified approaches for cost modeling in robotics.	89
4.2	Schematic structure of the life-cycle model.	93
4.3	Structure of the used cost-benefit model.	94
4.4	Classification of cash flows for system integrator and end-user view.	99
4.5	Differential cost-benefit assessment through comparison of an analyzed robot system with a benchmark.	101
4.6	Schematic of activity models and their inputs and outputs.	104
4.7	Combination of different sources of information during instantiation of the cost-benefit model.	113
4.8	Schematic of the process for model configuration.	113
4.9	Alternative scenarios of an event-triggered system of four processes with causality conditions (curved arrows).	119

4.10	Example of the NPV of a discrete cash flow of 1000 currency units occurring at the beginning of the second year.	125
4.11	Different domains of information required for the cost model.	130
4.12	Relation of top-level classes of the RoboCost ontology.	134
4.13	Resource classes of the RoboCost ontology.	137
4.14	Information related to members of the automation task class.	139
4.15	Information related to members of the model class.	140
4.16	Instantiation of activity model from serialized form.	141
4.17	Matching process between system composition and knowledge base in order to resolve uncertainties.	145
4.18	Flow chart depicting the handling of uncertainties in the cost model.	146
4.19	Specification of the similarity criterion in the knowledge base.	152
5.1	Detailed life-cycle during development, integration, commissioning and ramp-up of the robot system.	157
5.2	Weld seam parameters.	180
6.1	Architecture of the cost model implementation.	185
6.2	Top level class architecture.	187
6.3	Class architecture of different types of activity models.	188
6.4	Mechanism for data retrieval and provision by activity models.	189
6.5	Called functions and their tasks during instantiation of activity models.	197
6.6	Iterative process for cost-benefit model computation.	199
6.7	Computation pipeline and handling of events.	201
6.8	User interface of the costing software.	203
6.9	Query router and handling of SPARQL queries.	207
7.1	GMAW welded products used for the evaluation.	211
7.2	Simulated amortization behavior of the welding use case.	211
7.3	Amortization behavior of the GMAW welding use case from the point of view of the system integrator.	213
7.4	Amortization behavior of the GMAW welding use case from the point of view of the end-user.	214
7.5	Amortization curves of system integrator (top) and end-user (bottom) for overpricing.	216
7.6	Amortization curves of system integrator (top) and end-user (bottom) for underpricing.	217
7.7	Amortization curve for the case of high realization risk (overall amortization curve).	218

7.8	Amortization curves of system integrator (top) and end-user (bottom) with high realization risk.	219
7.9	Amortization curve with high usage risk (overall amortization curve).	220
7.10	Amortization curves of system integrator (top) and end-user (bottom) with high usage risk.	220
7.11	Runaway of project cost (overall amortization curve).	221
7.12	Amortization curves of system integrator (top) and end-user (bottom) with project runaway.	222
7.13	Distribution of cost between hardware cost and activity cost for the GMAW use case.	224
7.14	Distribution of cost between different life-cycle phases during design and integration for the GMAW use case.	225
7.15	Gantt chart of the realization of the GMAW use case.	226
7.16	Overall amortization view for the handling use case.	228
7.17	Amortization views of system integrator (top) and end-user (bottom) for the handling use case.	229
7.18	Cost distribution between hardware investment and activities for the handling use case.	230
7.19	Distribution of cost between different life-cycle phases during design and integration.	230
7.20	Project schedule for the handling use case.	232
7.21	Integration of the cost-benefit assessment into the development life-cycle of the robot system.	233

List of Tables

3.1	Conflict of interests of system integrator and end-user.	77
4.1	Machine-centric cost for robot systems.	95
4.2	Time-driven cost for robot systems.	103
4.3	Product model parameters and properties for typical processes. . .	107
5.1	Examples of parameters and properties for components.	156
5.2	Measures related to the functional complexity of the robot system in the concept and realization phase.	162
5.3	Functions of custom designed end-effectors for determination of functional complexity of the design process.	162
5.4	Functions of custom designed fixtures and work piece tables for determination of functional complexity of the design process. . . .	163
5.5	Typical values for constants for the acceptance testing model. . .	165
5.6	Resource parameters for maintenance.	169
5.7	Cost relevant parameters contained in the product model for GMAW seam welded parts.	176
5.8	Typical parameters contained in the welding procedure specifica- tion (WPS) of GMAW processes.	177
5.9	Cost relevant product parameters for handling tasks.	181
1	Overall requirements for the cost-benefit assessment tool.	244
2	Requirements for the cost model.	245
3	Requirements for knowledge management in the cost assessment tool.	246
4	Requirements for the integration of the cost assessment tool in the development life-cycle.	247
5	The SMERobotics role library for AutomationML.	248
6	Criteria for product complexity in the design phase.	249
7	Criteria for product complexity of the end-effector.	250
8	Criteria for product complexity of fixtures and jigs.	251
9	Parameters for the cost-benefit calculation of the GMAW use case.	252
10	Parameters for the cost-benefit calculation of the handling use case.	253

1 Introduction

During the emergence of industrial robot systems in the second half of the last century the automation of processes using industrial robots was seen as a universal cure for reducing manufacturing efforts. However, over time it became evident that, for many processes, robot systems do not form a basis for cost-effective solutions. Typical problems preventing cost-effective automation are high variant spectrums, small lot sizes, and interfusing processes that are easy to automate with processes that require significant development efforts for automation, handling of flexible parts being an example of the latter.

This is why most robot systems today can be found in mass production, especially in the car industry (Litzenberger, 2015). An increase of the use of robot systems in small and medium-sized enterprises (SME) is expected in the coming years. This is facilitated by modern information technology that allows reducing reconfiguration efforts of robot systems, in particular reprogramming efforts. These capabilities are essential for cost-effectively using robot systems in the production of SMEs because here reconfiguration is often carried out many times during the life of the robot system (see Figure 1.1). New types of robot systems are currently developed and implemented for these SME-type usage scenarios (Ajoudani et al., 2018; Krüger et al., 2009; Khatib, 2019). These robot systems feature new technologies such as direct human-robot collaboration, assistance to the worker and rich sensor systems for support of robot system reprogramming. They promise relief for small and medium-sized companies from manufacturing problems by offering more flexibility. These new SME-type robot systems potentially offer the ability to react more quickly to customer needs, changing demands, shorter product life-cycles and to reduce delivery times. Moreover, they promise to lower

the dependence on availability of skilled personnel, in particular in the face of impending demographic change. Furthermore, often cost savings can be achieved by reducing product variations through robot-based automation of processes.

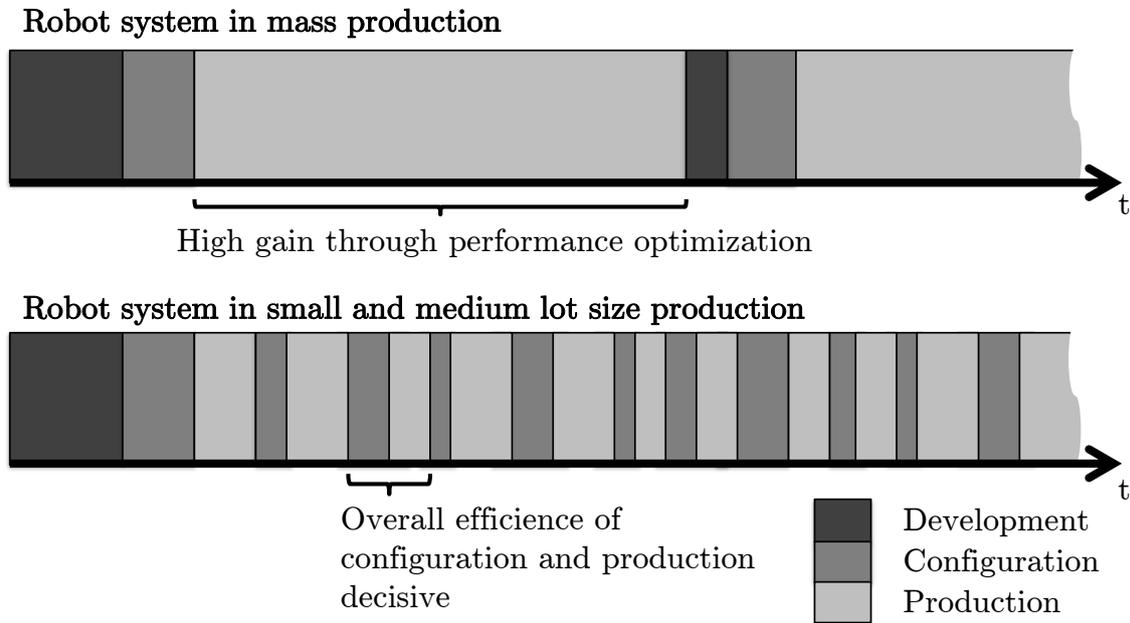


Figure 1.1 Comparison of use of robot systems in mass production and small and medium lot size production.

1.1 Motivation

As production scenarios and the robot systems carrying out the production task become more varied, the assessment of cost-effectiveness of those solutions needs to take into account additional factors, in particular flexibility and reconfigurability. This problem is exacerbated by several factors.

1. The lack of data for the assessment of advanced robotic solutions. The lack of historical data is a general hindrance for the widespread application of total cost models for the procurement of capital equipment (Hofmann et al., 2012). This factor is particularly important as robot systems are usually tailored solutions for one specific application scenario and therefore only built once or a few times.

2. The market for robot systems is fragmented and several, highly-specialized stakeholders, namely component suppliers, system integrators and end-users, are involved in building robot systems. This specialization on different aspects allows the creation of high-tech solutions, but leads to a lack of market consolidation in terms of economic and technical data.
3. New technical capabilities such as sensor based robots or advanced human-machine interfaces are costly to develop and integrate. The benefit of these features arises during the usage of the robot system and in particular if the application scope of the robot system changes. For a proper assessment of the benefits of such features in the development of the robot system, their effect over the complete life-cycle needs to be made transparent.
4. SMEs often lack experience and enterprise resource planning (ERP) systems with a sufficient depth and the resources to maintain them (Teittinen et al., 2013). Therefore, often the required data to assess the cost-effectiveness of a robot system prior to its implementation is lacking.
5. The risk associated with design decisions is not transparent in today's cost calculations. The term design decision here refers to choosing a particular system concept, component or construction principle of the robot system. If at all, several scenarios are computed to obtain a rough understanding of the sensitivity with respect to input parameters.

The fragmentation of the market (item 2 above) for industrial robotics has consequences on innovation. According to Coase, 1937, the emergence of companies is caused by an imbalance of internal costs and transactions costs. It is assumed that the robotics industry is stuck in this fragmented state as long-term relations between component supplier and system integrator lower transaction costs. This in turn prevents system integrators from growing sufficiently big to benefit from a lowering of internal costs resulting from direct access to internal component technology. The computation of different scenarios (item 5 above) is time-consuming as it is typically carried out manually. Often, the variation of parameters is not carried out systematically. Consequently, it is not sufficient for obtaining a good

understanding of the effects of design decisions on the cost-effectiveness over the entire life-cycle. A closed representation of the effect of variations in the input parameters would be beneficial in order to assess the risk associated with the system realization.

The motivation of this thesis is to alleviate the problems mentioned above by establishing methods and tools that allow the assessment of the cost-effectiveness and benefits of robot systems and their features. This motivation is the first problem statement concerning costs from a system-integrator point of view for SME-like robot systems.

1.2 Objectives

This thesis aims at modeling the typical cost structure of industrial robot systems over their complete life-cycle including development and commissioning efforts. The developed model and methods for its evaluation shall allow system integrators and end-users to:

- assess costs and benefits with uncertainties in input parameters and obtain an understanding how these uncertainties influence the profitability of the investment,
- use data from past projects for cost items that cannot be determined before building the robot system,
- make informed decisions on the design of a robot system and its required feature set for a specific application, and
- compare the economic benefit of a robot system to other alternatives, i.e. other automation solutions, manual production or outsourcing.

Three main components are required for achieving these objectives and overcoming the above mentioned limitations in current cost computations for robot systems:

1. A cost model that allows to model key economic factors (cost and benefits) and their relation to the composition and usage of the robot system.
2. Methods for coping with uncertainties in the entire cost-benefit model.
3. Methods for data management in order to combine design data of a specific robot system with data from past projects and project unspecific data. These methods are required in order to parameterize the cost model with as little manual data input as possible.

This thesis takes the approach of combining methods from two scientific fields. Firstly, it builds on cost analysis methods from business economics. These methods are combined with approaches for knowledge management and uncertainty handling from the field of robotics. The aim is obtaining a cost-benefit model that is sound with respect to costing terms, spans the entire system life-cycle, and is sufficiently detailed to model typical cost-causing activities in the development and usage of robot systems. For that purpose, different methods in business economics are analyzed with respect to their applicability and shortcomings in terms of modeling of specific robotic aspects. These methods are combined with approaches for structuring information and handling knowledge in the domain of robotics. The approach is to combine the high-level structure and formal rigor of economic costing methods with methods for knowledge handling and with the low-level system structure from robotics. This yields a cost-benefit model that allows the analysis of cost and benefits of different design alternatives in robot systems during the development process.

1.3 Scope and applied method

The main focus of analysis in this thesis is on system integration aspects and the work of the system integrator. The assessment addresses costs and benefits that can be directly quantified in currency units as they are directly related to consumption or saving of human resources, commodities or energy (Figure 1.2).

Due to this focus a strong emphasis is placed on modeling the activities for realization of robot systems. The views of other stakeholders and other dimensions of Figure 1.2 are also touched but handled in less detail. Operational aspects at the end-user are covered, but the focus here is obtaining a rough assessment of costs and benefits of the usage in production. This assessment allows covering the full life-cycle and therefore to offer decision support in early design phases. The applied level of detail for operational aspects at the end-user is much lower compared to existing work and common practice in operations research. The analysis of costs and benefits also includes information from component manufacturers as an input for modeling integration aspects and their costs. Component manufacturers are not targeted as users for the cost-benefit model though. The analysis does not address distal-monetary benefits, for example strategic advantages or benefits due to increase sustainability, for instance the decrease of environmental impact or the improvement of ergonomics.

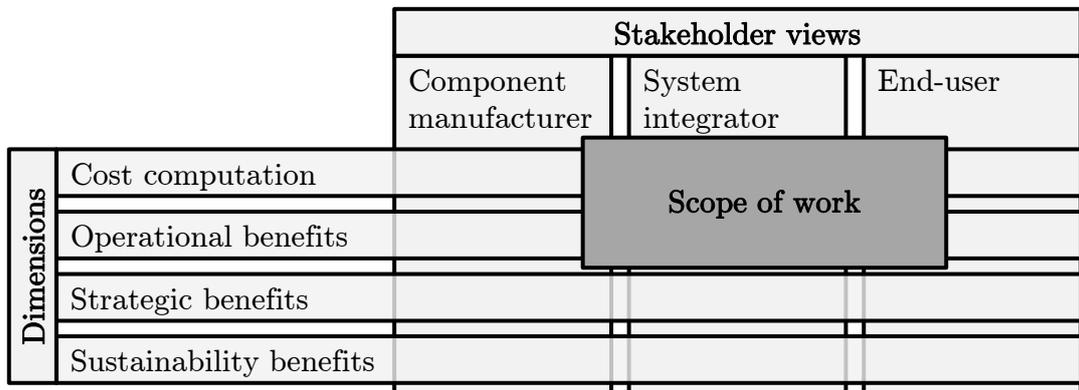


Figure 1.2 Positioning of the scope of this thesis in terms of stakeholders and dimensions of cost and benefit.

The intended usage scenario of the proposed cost-benefit assessment tool is decision support in early life-cycle phases of the robot system. In particular this concerns decision support during contract negotiation, design and integration. The aim is to analyze cost and potential benefits of using the analyzed robot system and reflect this information back into the design process to improve design and purchase decisions. The system integrator is the main user of the results. It can use the cost-benefit assessment tool to improve decision making in pricing

and system design. Additionally, it might use the model to convey benefits of the proposed robotic solution to the end-user.

The cost-benefit model quantifies benefits in terms of cost and cost-savings compared to an existing or alternative solution, which acts as a benchmark. It assumes a certain type, volume and mix of parts that needs to be produced by the robot system or the benchmark to compare the costs of both options.

The proposed cost-benefit model is evaluated by creating a benchmark implementation and by validating the results for two use cases that are typical for robot applications in SMEs - gas metal arc welding and machine tending. With these use cases, it can be assessed if the cost-benefit model produces plausible estimates for cost, benefits and realization time of the robot system. No formal validation of the cost-benefit model can be achieved with this approach. However, the validation with two use cases shows that it is possible to model the costs of components, integration and usage of SME-like robot systems with the proposed model structure, knowledge representation and uncertainty description.

This thesis is structured as follows. Chapter 2 gives an overview of the state of the art, and provides background required for an understanding of the developed cost-benefit assessment method. In Chapter 3, the problem of this thesis is analyzed in the face of the state of the art and requirements for the cost model and related components are given. The general design of the cost-benefit model developed here is introduced in Chapter 4. Chapter 5 lists typical model formulations for robot systems that allow to describe cost-causing activities during system integration and usage. The implementation details for the cost models and related components are outlined in Chapter 6. The results of the costing method for typical examples are shown in Chapter 7. The thesis is concluded in Chapter 8 and its main contributions are stated. An outlook for further research is given.

2 State of the art and background

The work reported here is multidisciplinary and combines methods from business economics and knowledge engineering in robotics. Figure 2.1 shows the fields of activity and domains that are used for achieving the aims of this thesis. Methods for assessing costs and benefits of robot systems are needed for obtaining quantitative results. Management accounting and capital expenditure budgeting reflect the managerial view on the problem and provide methodical grounding for cost-benefit assessment. Costs and benefits are computed by using models of the efforts caused by realizing and operating a robot system. Capital expenditure budgeting also employs simple models of cost creation, in particular for the use of purchased assets. Existing approaches in modeling of manufacturing systems and engineering design provide algorithms for modeling cost creation. In order to build these models, methods for describing and managing the knowledge on robot systems as a whole are required. The cost-benefit calculations need to reflect the inherent uncertainty. Hence, methods and mathematical formalisms for describing uncertainty are required for modeling as well as cost-benefit assessment. The state of the art in these domains of knowledge is introduced in the following sections.

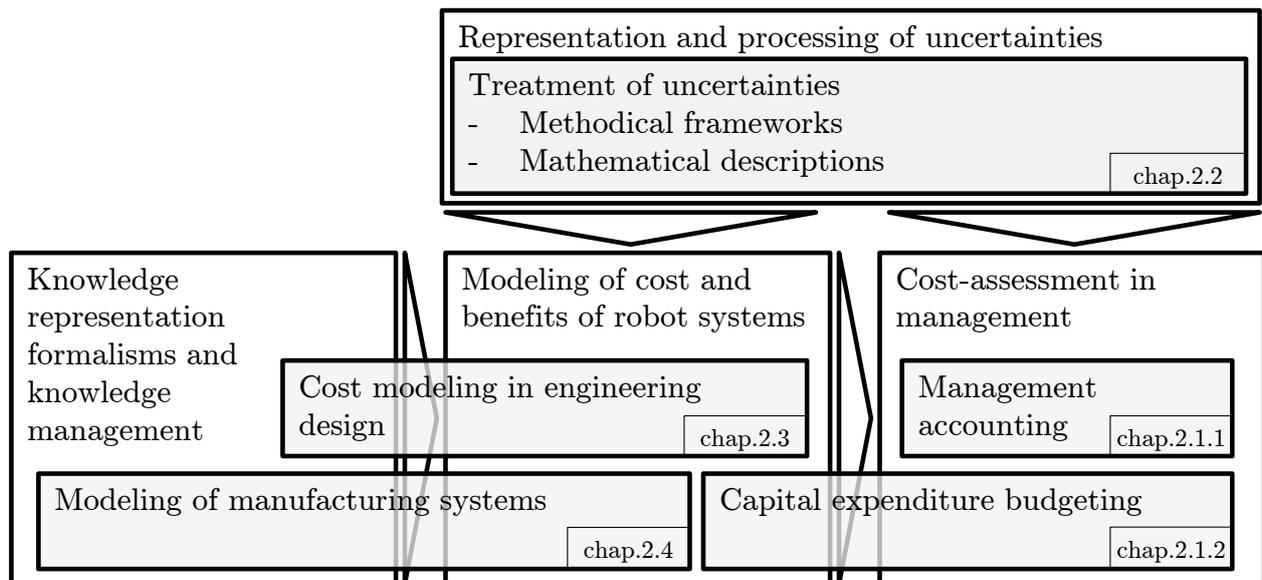


Figure 2.1 Relevant knowledge domains for cost-benefit assessment of robot systems.

2.1 Accounting and capital budgeting

Numerous approaches for cost-assessment in production exist. These approaches mainly stem from the fields of management accounting and capital budgeting. Management accounting is concerned with determining the production cost of products. Capital budgeting is applied for making investment decisions.

Figure 2.2 gives an overview of the methods that are introduced in the following sections. In management accounting traditional accounting methods and controlling frameworks are distinguished. Traditional accounting frameworks focus on breaking down costs to products and organizational units of the company and generating financial statements. Controlling frameworks are intended to support decision making with accounting information. In capital expenditure budgeting static and dynamic methods are distinguished. Dynamic methods take into account the time value of money, while this factor is neglected by static methods. The point of view of this thesis is more technical when compared to most work in business economics. Each subsection in this section summarizes remaining issues from this viewpoint.

Management accounting Monitoring organizational performance		Capital expenditure budgeting Assessment of investment decisions	
Accounting methods	Total absorption costing Allocation of all cost to cost centers	Static methods	Pay-off method Computation of time interval until invested money is paid back
	Marginal planned accounting (GPK) Only causally linked cost are allocated to products and services		Accounting rate of Return Calculatory yearly return of investment compared to bound capital
	Activity-based costing (ABC) Costs are allocated through the usage of cost-causing activities	Dynamic methods	Net present value (NPV) Discounting of all cash-flows resulting from investment to present
	Resource consumption accounting Hybrid of GPK and ABC where ABC is applied for causally linked cost where useful		Internal rate of return (IRR) Interest rate that leads to NPV of zero
Controlling frameworks	Target costing Cost limits are defined in product development through target price and target margin		Economic value added (EVA) Difference of profit and cost of capital
	Project accounting Usage of management accounting principles for project monitoring and decision making	Profitability index Quotient of NPV and initial investment sum	
	Balanced scorecards Systematic framework for assessing the company's performance	Discounted payback Time interval for which NPV is zero. Dynamic version of pay-off method.	
			Equivalent annuity method Yearly cost of owning an asset
		Life-cycle costing Costing framework that observes all life-cycle phases of an asset	

Figure 2.2 Overview of analyzed accounting methods.

2.1.1 Management accounting

A key problem in industrial companies is how costs are allocated to products such that the efforts caused by the production of these goods are reflected. While costs directly related to the product, for example material cost, can be easily distributed, this is not easily possible for indirect costs, for instance infrastructure costs. Similar problems occur for investment decisions. Here costs resulting from the investment have to be distributed to products. Management accounting is the discipline that is concerned with a solution for the dilemma of cost distribution.

Management accounting traditionally distinguishes total absorption costing practices and marginal costing practices. Total absorption costing practices allocate all types of cost to products and services. Marginal costing practices only alloc-

ate variable costs to the cost object, that are directly and causally linked to the product or service.

Total absorption costing

Total absorption costing assigns all costs including all overhead costs to cost centers. Indirect costs have to be distributed on the basis of a distribution key in a process called apportionment. In primary apportionment, indirect costs are distributed to single cost centers. In secondary apportionment, the costs of non-production cost centers are distributed to production-related cost centers. The main criticism of total absorption costing is that costs are not distributed based on a causal relation. Total absorption costing is less targeted towards making decisions in order to optimize the companies profit, but rather targeted towards external reporting as reported by Polejewski (2013a).

Marginal planned cost accounting

Grenzplankostenrechnung (GPK) (Kilger et al., 2012) or also called Marginal Planned Cost Accounting is commonly applied in Germany and increasingly popular in North America. Marginal cost refers to the increase in total cost in case of an increase of the produced quantity by one. In GPK, only indirect and direct cost that are linked to products and services are attributed to the product. These costs are used to determine the proportional cost per cost item, which gives a first indication regarding the products' profitability mostly suited for short term decisions. Fixed cost are not assigned to the cost objects but evaluated as a block in the profit and loss calculations. This addresses one of the main criticism with respect to total absorption costing where the inclusion of fixed costs leads to wrong decisions as these cost typically cannot be influenced on the short term. For instance, the distribution of fixed cost to products often results in negative contribution margins of products disguising the fact that elimination of this product will not increase profit if the respective capacity cannot be released. GPK has the main aim to facilitate short-term decisions, e.g. regarding marketing measures

or process decisions in manufacturing. This fact and the good standardization in common software are amongst the main criteria for success of GPK as reported by Sharman et al. (2004).

According to Friedl et al. (2009), the core elements of GPK are cost-type accounting, cost center accounting, product cost accounting and contribution margin accounting. Cost-type accounting distinguishes the proportional cost, i.e. costs that are causally linked to the products and services and fixed costs that cannot be assigned to single cost objects. Cost centers establish organizational elements on which actual cost and budgeted cost are regularly compared to evaluate performance. These cost centers also bear the managerial responsibility for these costs. Besides accounting on the cost center level, costs are controlled on the product level. Contribution margin accounting is concerned with determining the contribution of single products to fixed costs and profits. In a first step, the margin per product after deduction of proportional costs from revenues is computed. After that, fixed costs are deduced in the calculation en-block. The computation of fixed costs is often carried out on different levels, e.g. product, product group, organization. The implementation and maintenance of GPK systems causes considerable effort. According to Krumwiede (2005) this is primarily the case because they require detailed planning down to the level of single cost centers.

Activity-based costing

As non-fixed indirect costs constantly increased and eventually surpassed direct costs in particular due to an increase of complexity in production, new methods in costing emerged to address this problem. Activity-based costing (ABC) as a term is defined by Edwards (2012) as a costing approach centered around monitoring resource consumption for achieving final outputs. It assigns resources to activities and those activities to cost objects. The cost objects use cost drivers to model activity costs dependent on the outputs. According to Kaplan (1997), ABC helps to answer four key questions: why money on indirect and support resources is spent (activity dictionary); how much an organization is spending on each of its activities (cost drivers and unit costs); why these activities are performed; how

much of these activities are required for each product, service or customer. To set up an ABC system in a company, extensive data collection efforts are required for obtaining data on how much resources an activity consumes (Kaplan et al., 2004). Only this allows obtaining values for the unit cost of a cost driver. Cost drivers can either be transactional, i.e. counting the number of occurrences of an activity, duration-driven, i.e. counting the time effort for a certain activity, or intensity-driven, i.e. counting the time and intensity effort of an activity.

ABC emerged from total absorption costing and allocates costs related to the execution of activities depending on the degree of usage (Staubus, 1990). As outlined by Kaplan et al. (2008), its aim is to determine measures for cost and profitability of products and processes in order to support decision making. ABC takes the perspective that all costs are variable when seen in long perspective (Perkins et al., 2011). According to Neely et al. (1995), ABC in particular helped to shift the focus of cost accounting towards operative decisions rather than reporting. Therefore, ABC helps to deal with the shrinking percentage of direct labor cost when compared to product cost and increased complexity of the marketplace and production. ABC has in particular been popular in the United States, but was abandoned by many companies due to the effort for introducing and maintaining the system and a lack of standardization and commonly used software tools (Sharman et al., 2004). In Germany, the approach was adapted as Prozesskostenrechnung (Horváth et al., 2011) centered around processes as a collection of cost-causing activities that may span different cost centers.

Kaplan et al. (2004) further developed the approach of ABC into time-driven activity-based costing (TDABC). TDABC in particular addresses the high cost associated with the introduction and maintenance of original ABC systems. In TDABC, time-based unit costs are estimated based on the practical capacity of the employed resources. This time-based unit cost is then used together with time estimates for the activities to distribute costs to the activities.

Resource consumption accounting

Resource consumption accounting (RCA) (PAIB, 2009) is a combination of GPK and ABC. It has been developed in order to address common criticism regarding the cost and benefit of ABC. RCA establishes rules and limits on the use of ABC (Van der Merwe et al., 2002; Clinton et al., 2004). RCA utilizes the approach taken by GPK to distinguish proportional and fixed cost and only assigns cost to single cost objects that are casually linked (Perkins et al., 2011). Cost drivers in RCA can be traditional or activity-based integrating aspects of ABC (Clinton et al., 2004). According to Polejewski (2013b), RCA is characterized by three main principles. Firstly, RCA uses cost centers that are homogeneous with respect to the employed resources similar to GPK. Secondly, RCA uses a quantity-based model similar to ABC instead of directly distributing cost to products and services. Thirdly, like GPK, RCA distinguishes proportional and fixed cost and only proportional cost that are causally linked to products are distributed to cost objects. As the maximum capacity of resources is used as calculation basis, idle resources can be accurately identified when using RCA.

Further approaches in management accounting

Target costing is a marginal costing approach that aims at achieving cost properties of products or services that allow to achieve certain target profit margins in the market. The target cost is determined by subtracting the desired profit margin from the desired competitive market price.

Project accounting is an accounting approach used in project management. While standard management accounting methods aim at determining the financial performance of organizational elements of the company, project accounting aims at assigning direct and indirect costs to projects. The motivation of this thesis is highly aligned with the rationale behind project accounting as the commissioning and operation of an automation solution can be seen as development and procurement project.

Balanced scorecards is a method for holistic review of objectives in an organization. According to Kaplan et al. (1992), balanced scorecards offer managers the possibility to reflect the performance of a company with respect to different aspects. Most common are the perspectives financial, customer, internal business process, and learning or growth. Therefore, the balanced scorecard method leads managers to reflect the view of customers and investors on the company. Furthermore, it answers the question, in which areas the company must excel and how the company can improve and create additional value. The method is rather a tool for reviewing organizational performance than an accounting method or tool for measuring cost effectiveness.

Summary of management accounting tools

Management accounting tools offer a general, well grounded framework on how cost can be distributed to different cost objects. However, the methods do not supply dedicated knowledge on how the cost generation can be modeled accurately, in particular during the design phase of special machinery. Typically, methods from management accounting assume that the investment cost and performance parameters of the robot system are known. Therefore, they miss a critical life-cycle phase that is defining for the overall cost effectiveness of special machinery, as they do not look into the design and integration phase. As will be outlined in the following section, some capital expenditure budgeting methods offer a framework for analyzing cost and benefit in different life-cycle phases of a machine.

2.1.2 Capital expenditure budgeting

Capital expenditure budgeting aims at assessing investment alternatives in terms of the value created for the investing company. The question whether an investment into new production equipment is favorable from an economic point of view is a central decision problem for all industrial companies. The technical advantages of new machinery have to be assessed in economic terms and compared to alternatives to justify the investment. This field has been researched top-down for

a long time with a strong focus on operational aspects. Typically, static methods and dynamic methods are distinguished in capital expenditure budgeting. Contrary to static methods, dynamic methods take into account the present value of cash flows by discounting future cash flows with a calculatory interest rate. Dynamic methods, in particular internal rate of return and net present value, are most popular in industrial practice (Ryan et al., 2002; Pike, 1996; Graham et al., 2001). Apart from this differentiation, the focus has been shifted on the entire life-cycle as costs during usage are defined in early phases of an investment project. In this context, life-cycle costing (LCC) and total cost of ownership (TCO) have emerged to reflect the change in scope of capital expenditure budgeting.

Static methods

The comparative cost method compares the cost associated with different investment options. The comparison either compares the cost accruing during a certain time interval or the cost per output piece. Typically, the comparative cost method takes into account the asset depreciation, cost of capital and usage cost. The profit comparison method is similar to the comparative cost method, but compares the profit resulting from different investment options. The pay-off method, a static method, is commonly applied in order to determine the time interval for an investment required to pay back the invested money. The Accounting Rate of Return (ARR) computes the return of an investment by comparing the yearly income related to an investment with the bound capital. ARR does not consider the time value of money. As static methods offer limited expressiveness due to the failure to recognize the time value of money, they have become uncommon in industrial practice (Ryan et al., 2002). However, static methods are still employed to obtain quick forecasts for expected returns of investments.

Dynamic methods

Two main decision rules are applied in dynamic capital budgeting: Net Present Value (NPV) and Internal Rate of Return (IRR). Both approaches rely the dis-

counting of cash flows over time. With the Discounted Cash Flow (DCF) method, each cash flow is discounted up to a common time base in order to reflect the different values of cash flows depending on their time of occurrence. The discounting of a cash flow C in year i with the commonly used exponential method follows the formula:

$$\text{NPV} = \frac{C_i}{(1 + q)^i}, \text{ where } q \text{ is the discount rate.} \quad (2.1)$$

NPV The NPV decision rule is very common in industrial practice (Ryan et al., 2002). For the NPV, all cash flows related to an investment are discounted to the present with a calculatory interest rate q . The calculatory interest rate should reflect the company's cost of capital plus a risk adjustment representing the inherent risk of the project regarding future cash flows (PAIB, 2008). An investment with a positive NPV creates value in the company. With the Certainty Equivalent Method individual cash flows are adjusted to risk free cash flows applying an individual risk assessment. These risk free cash flows are then discounted with a calculatory interest rate not including project risk. NPV is an absolute measure as it indicates the absolute value in currency units that is generated by an investment.

IRR The Internal Rate of Return (IRR) is the interest rate that leads to an NPV of zero. The IRR is closely tied to the opportunity cost of the investment decision. The funds to be used for investment could also be invested at an assumed discount rate if the investment is not made. Therefore, the return of the investment has to surpass this assumed discount rate before being of interest. In contrast to NPV the IRR is a relative measure as it indicates the profitability of an investment with respect to the invested sum. Hence, when a small investment with a higher IRR is compared to a large investment with a lower IRR the NPV of the large investment might still be higher. This is because, despite its higher profitability, the smaller project might generate less absolute value due to its smaller size.

EVA The Economic Value Added (EVA) is the difference of the profit, i.e. net operating profit after tax, and the costs for capital and therefore a measure for the increase of company value for the owner (Mäkeläinen, 1999). It is

a measure for the value created by an investment, e.g. a machine. While originally being a financial performance metric, EVA can also be applied to investments as it puts the returns achieved from an investment in relation to the cost of financing this investment.

PI The Profitability Index (PI) denotes the quotient of the present value of all future cash flows and the initial investment in the asset. A PI of 1 indicates break even. PI is in particular useful for ranking different investment options.

Discounted Payback When using the Discounted Payback method the payback of an investment as the point in time at which the sum of all cash flows becomes zero is computed. The Discounted Payback method discounts future cash flows making it a dynamic method.

EAC The Equivalent Annuity Method or Equivalent Annual Cost (EAC) refers to the yearly cost of owning an asset over its entire life span. It is computed by dividing the NPV by the present value of the annuity factor. EAC is in particular useful when assessing different investment options with differing life spans and similar risk.

Dynamic methods from capital expenditure budgeting form an established and accepted basis for the evaluation of cost and benefits of capital intense investments like robot systems. The above mentioned methods can be applied for accumulating costs and benefits that can be measured as cash flows along the life-cycle of the robot system by discounting them to a common point in time and adding them. In particular the NPV and IRR decision rules are established and well known by many decision makers. The NPV is favored as a measure, because the aim of this thesis is the comparison of the absolute value generated by an investment.

Life-cycle costing

A key question in capital expenditure budgeting is how costs are structured so all relevant cash flows are taken into account. Traditional approaches typically focus on the direct investment cost and compare that cost to the return that is

obtained from using the investment. Life-cycle cost (LCC) oriented approaches shift the focus of investment decisions away from the directly visible costs such as purchase price to an assessment of the total cost occurring due to the use of the investment over all phases of the life-cycle. This includes for example engineering costs, costs for adaptation of the production site, ramp-up costs, maintenance and repair cost as well as disposal cost. LCC allows the purchasing company to get an understanding of real costs of an invest decision and to make decisions that optimize the cost and benefits in a holistic way.

Total cost of ownership (TCO) (Ellram et al., 1998) expands the focus of LCC to the interaction between the investing company and suppliers (Hofmann et al., 2012). TCO expands LCC by including indirect costs such as cost for supplier selection, qualification of suppliers, transportation and receiving inspection. According to Ellram et al. (1998), TCO is most frequently used for capital intense purchase decisions such as the investment in manufacturing equipment. Other uses include make or buy decisions, repetitive purchases, selection of components for own products, and process optimization by identification of the real cost of certain process steps. The initial effort to set up TCO is often cited as a mayor hindrance. Zachariasen et al. (2011) propose that TCO models should take into account the relation of supplier and purchasing company, as many of the indirect costs depend on this relationship. Ferrin et al. (2002) conducted a study on the use of TCO in organizations. They suggest that core cost drivers that are relevant for TCO calculations for all commodities and products exist while the models need to be augmented by cost drivers specific for the respective purchased item.

When talking about cost-assessment tools for automation, different viewpoints need to be distinguished:

- Decision-making support for investment decisions from the view of the end-user.
- Decision-making support for machine utilization from the viewpoint of the end-user.

- Decision-making support for design decisions for the machine development of the machine-producer.

Most costing approaches take the view of the end-user to support his investment decision. Several tools and methods specifically targeted at robotics and automation exist for this decision support problem. In the European research project SMErobot, an LCC tool (Lerch et al., 2010) was developed to assess the cost effectiveness of robot solutions in SME environments, in particular in comparison with manual production. This tool captures data on:

- operational aspects of the SME, for example number of shifts, days of operation and work times,
- data on the existing production of the SME (i.e. typically using manual labor),
- cost parameters related to purchasing of the robot system,
- operating costs of the robot system, and
- quality aspects of the manufacturing process.

The splitting of input data into different categories is meant to facilitate input by different experts from different companies as one expert can be assigned to each domain depending on his specific area of expertise. The tool supports the analysis of sensitivity for different financing options of the robot system. The approach is directed at decision support for machine procurement.

Fraunhofer IPA and ISI conducted the study Effirob (Hägele et al., 2011) on the cost-effectiveness of service robot solutions. For this purpose, a dedicated LCC tool and an underlying method were developed. The approach differentiates service oriented and industrially oriented scenarios. For service oriented scenarios, the performance is usually measured as the result of this service, e.g. cleaning performance in cleaned surface or inspection performance in meters per hour. In industrial scenarios, the output is usually the result of an industrial process, e.g. assembled parts.

The German Engineering Federation (VDMA) developed an LCC specification for machinery (VDMA 34160-06, 2006) and a computation tool implementing this specification. The tool is dedicated to machine suppliers for presenting the economic data of their machinery to prospective customers and justify price premiums that may pay off quickly during machine usage. The applied life-cycle model contains preparatory tasks, machine operation, and further utilization of the machine. The specification contains an extensive listing of relevant cost types for the computation of life-cycle costs for machinery. Therefore, the user has to supply input data on many cost items throughout the life-cycle of the machine. The specification takes a machine buyer's view and does not cover system development and installation efforts that are typical for robot systems. This LCC tool does not track uncertainties in input cost data and is not connected to engineering models of the machinery.

The German Electrical and Electronics Manufacturers' Association (ZVEI) (Birkhofer et al., 2012) points to the costs that are caused due to the different life-cycles of automation systems, their components and subcomponents as the life-cycle of automation systems is usually considerably longer than that of their components. Due to reuse, the life-cycle of automation systems is becoming longer which in turn requires a life-cycle based approach to assessing investments.

Preventive and scheduled maintenance contribute significantly to the cost over the life-cycle of a system (Lauven et al., 2010). Fleischer et al. (2007) combined life-cycle costing methods with Monte-Carlo methods (see also Section 2.2.2) in order to estimate maintenance cost over the entire machine life-cycle. In particular, the cost for corrective maintenance over the life-cycle is hard to determine and depends on different reliability measures. Mannuß et al. (2012) propose the execution of a combined design and availability FMEA (CDA-FMEA) to generate the respective input for the life-cycle model.

Most LCC methods do not take into account that design decisions made in early life-cycle phases have a strong influence on cost in subsequent life-cycle phases. This dependency is particularly strong for special machinery where numerous design parameters exist. Furthermore, LCC methods typically require the provi-

sion of large amounts of input data and fall short in integrating knowledge from different stakeholders and in handling uncertainties.

Conclusions about capital expenditure budgeting methods

Both static and dynamic capital accounting methods focus on how cash flows and value transfers influence the profitability of investments, but do not track how the height of these cash flows is connected to technical properties of the purchased asset. This view works in case that there is a discrete number of alternatives with different price performance trade-offs, for example when purchasing a machine tool and different manufacturers offer machines in the range of interest. However, in special machinery, there are countless decisions regarding the nature of the machine that are taken in the design phase. All these decisions influence the required investment, risk, and also performance of the machine in production. To properly assess the optimal investment alternative for special machinery, an integration of methods from capital budgeting into the design process of robot systems is required as proposed by this thesis. For this purpose, the NPV method seems to be most suitable as it is straight forward to compute and widely used.

2.1.3 Cost considerations for flexible manufacturing systems

Flexible manufacturing systems (FMS) are manufacturing systems that allow the reaction to changes in product or market in a flexible manner. FMS have been a very active area of research in conjunction with the computer integrated manufacturing (CIM) approach during the 1980s. The evaluation of the cost effectiveness for FMS has similar challenges to the evaluation of the cost effectiveness for robot systems. An active area of research has been to find optimal strategies for use of FMS in coexistence with traditional manufacturing systems.

Rezaie et al. (2007) proposed an objective function and constraints for the optimization of the implementation strategy of FMS. Their model includes costs for

implementation and for machine reconfiguration for new products. However, the model does not contain a cost assessment for the FMS during the usage phase. Ramasesh et al. (1997) proposed a methodology to take into account flexibility of production solutions in discounted cash flow (DCF) analysis. An objective function and constraints are used for simulating the NPV of the optimal cash flows. Different levels of flexibility that are subject to a probabilistic distribution of the manufacturing environment are taken into account in the analysis. The authors classify flexibility as product-mix flexibility, volume flexibility, material flexibility, machine flexibility, and innovation flexibility with respect to process and product.

Kulatilaka (1988) argues that typical evaluation methods such as NPV are not suitable for evaluating highly flexible systems as they do not reflect the proper value of flexibility. The author proposes a dynamic model to incorporate the value of flexibility in the capital budgeting process. For this purpose, different operating modes, switching cost and profit functions are defined for the FMS. Using an analogy from financial options, the value of flexibility can be determined and optimal switching decisions can be made over the life-cycle of the FMS. Kulatilaka notes that design and cost assessment have to be conducted concurrently as operating modes depend on design decisions taken for the FMS.

Koren et al. (1999) highlight the productivity limitations of FMS and propose reconfigurable manufacturing systems (RMS) as a new alternative where reconfiguration of the plant and machines are designed into the system. They argue that the ability to upgrade the system at later times and the extended life-cycle helps to reduce overall life-cycle cost.

Fine et al. (1990) derive a model for the trade-off between cost and flexibility in FMS which consists of a two-stage stochastic problem. The first stage is an investment decision concerning uncertainty in the actual product demand. In the second stage, the manufacturing operations are optimized with known product demand under the constraints imposed by the investment decision. Fine and Freund carry out a sensitivity analysis with respect to the investment cost of the system and demand fluctuations. This analysis shows that the correlation

between risk and demand of different products is complex and hard to predict by an expert without decision support.

Kaplan (1986) notes that FMS are hard to justify using capital expenditure budgeting methods such as DCF. One reason identified by him is unrealistically high assumptions regarding profit rates of the investment which are unjustified by the cost of capital and the neglect of intangible benefits. Kaplan proposes that for FMS projects with negative NPV, the managers should evaluate how much they expect the intangible benefits to be worth per year and recalculate the NPV using this figure.

Aggarwal et al. (1991) propose to use a method called Adjusted Net Present Value (ANPV) for capital budget decisions regarding FMS. ANPV takes into account intangible benefits such as strategic advantages, better reactivity to customer demands and increase of market share. However, no details on quantification of cash flows for these intangible benefits are given. They argue that most importantly the implicit assumption of NPV methods of no response by competitors does not hold and a potential loss of market share in case of no reaction regarding flexibility should be included in the capital expenditure budgeting decision.

Concluding, most of the found approaches for evaluating the cost effectiveness of FMS focus on the optimal strategy for investing in and using FMS. The work on FMS seems mostly address general, generic measures of flexibility and not to discuss specific approaches for creating this flexibility on technical levels and their implication on cost effectiveness. For the problem at hand, one major challenge resides in making design decisions that govern the level of flexibility and reconfiguration cost of the robot system. This aspect was found to be inadequately addressed in existing approaches for FMS. The existing work on FMS can contribute to the objective of this thesis by providing metrics for the evaluation of design decisions.

2.2 Uncertainties and related arithmetics

Uncertainties and their relations play an important role in many disciplines of engineering, finance, and economics. In the following, common approaches for classifying uncertainties and risk are presented. Furthermore, methodical frameworks for treating uncertainties in system analysis and decision making as well as mathematical descriptions of uncertainties are introduced.

2.2.1 Classification of uncertainties

One common distinction of uncertainties found in literature is between so-called epistemic and aleatory uncertainties (Helton et al., 1996). Epistemic uncertainties relate to not knowing the value of a quantity that is assumed to have an exact value. Uncertainties stemming from an inherent randomness of the underlying quantity are called aleatory uncertainties. Epistemic uncertainties are strongly related to the concept of risk. In the design and operation of robot systems, both types of uncertainties are typically mixed. For example, the exact cost of certain resources, such as test parts or labor are well defined, but typically can only be determined with limited level of certainty. Therefore, these quantities are subject to a certain level of epistemic uncertainty. The effort for certain activities is related to the ability of the integrator of identifying problems during integration or hardware faults which are inherently random. Therefore, the required integration effort is subject to aleatory uncertainty.

Nilsen et al. (2003) study model uncertainty, i.e. the description of uncertainty introduced through the deviation of real world and system model, in the context of risk analysis. They offer two interpretations of risk modeling. Firstly, the classic interpretation views the uncertainty in model inputs as aleatory uncertainties stemming from the inherent randomness of the system. Secondly, the Bayesian point of view interprets the uncertainties in inputs as purely epistemic, stemming from the lack of information about the system. While the term model uncertainty is hence of no meaning in the Bayesian interpretation, the authors conclude that

the close analysis of model uncertainty diverts attention from the main objective in risk analysis.

Kennedy et al. (2001) classify uncertainties in computer models with respect to their sources. The first found source are uncertainties related to parameters, i.e. the inputs of the model. The second source is the inadequacy of the model itself to describe the modeled process. A further source of uncertainties are residual values stemming from deviations in the modeled process itself in repeated executions. Additionally to inaccurate parameters, uncertainties might stem from model inputs, that cannot be accurately specified (parametric variability). Lastly, uncertainties might result from observation errors, which refer to inaccuracies in reference measurements of the modeled process.

Lawson (1988) gives an overview of different interpretations of uncertainty in the context of economic analysis and carves out two main interpretations of uncertainties. One views uncertainties as an object of knowledge or also as properties of reality. Another view treats uncertainties as a type of knowledge. The first interpretation appears to be closely related to the definition of aleatory uncertainties, while the latter corresponds to epistemic uncertainties.

Beraldi et al. (2013) introduce two approaches towards accounting for risk in capital budgeting. The first approach aims at managing the risk of a certain set of investment decisions by introducing a risk objective function. The second approach involves the definition of a reactive strategy in order to correct investment decisions as soon as information on uncertain parameters evolves. This thesis follows the first approach and focuses on the decision in favor or against one particular automation project.

Different approaches for describing uncertainties are reported in literature (Helton et al., 2004; French, 1995). The most common descriptions are probability theory, evidence theory (also referred to as Dempster-Shafer Theory), possibility theory, and interval analysis. In contrast to the other approaches, interval analysis does not rely on an understanding of the structure of uncertainty. This is strength and weakness of interval analysis as it makes the method very easy to use, in particular

when relying on expert estimates, but also limits the explanatory power of this method.

2.2.2 Methodical frameworks for analyzing systems under the influence of uncertainties

Uncertainties can be incorporated into system analysis and decision making in different ways. This section presents common approaches from system modeling and decision making for treating uncertainties.

Monte-Carlo Methods

Monte-Carlo Methods (MCM) are widespread in treating uncertainties. One of the first proponents of numerical MCM was Hertz (1964) in the 1960s. MCM basically refer to carrying out random experiments using a model of a real-life system running on the computer (Kroese et al., 2014). This allows treating uncertainty in input parameters due to inherent risk and analyzing inherently random systems. MCM uses the law of large numbers (see also Section 2.2.3) to find expected values of the analyzed measures.

MCM involves randomly generating a large number of input sets for the analyzed model from a defined probability distribution. These inputs are then used to solve the model through deterministic computations. The results of these random experiments allow to compute expected values for the model results due to the law of large numbers. This procedure involves a large number of computations of the model for each change in the input distributions. MCM has been applied in capital budgeting as reported by Smith (1994).

Sensitivity and uncertainty analysis

Sensitivity analysis and uncertainty analysis deal with the question how uncertainties in the output of a given model are related to uncertainties in its in-

puts (Saltelli et al., 2008). Given model outputs $\vec{y} = [y_1, \dots, y_n]$ and model inputs $\vec{x} = [x_1, \dots, x_m]$, uncertainty analysis deals with the question how big the uncertainty in \vec{y} is depending on a known amount of uncertainty in \vec{x} . In contrast, sensitivity analysis answers the questions how strongly the single elements in \vec{x} drive the uncertainty in \vec{y} (Helton et al., 2006). Sensitivity and uncertainty analysis are conceptually and computationally closely related.

To assess the sensitivity of the cost model with respect to uncertainties and risk, a nominal value is used to compute the cost model. In the following, the input parameters are varied, for example through MCM. This uncertainty analysis gives an indication how the model reacts to variations in input parameters (Saltelli et al., 2008). The uncertainty analysis forms the basis for sensitivity analysis that quantifies how the model reacts to variations of a specific parameter. For this purpose, meaningful probability distributions for the input parameters have to be found. This often causes tremendous effort and belongs to the most important parts of sampling-based uncertainty analysis (Helton et al., 2006). The model itself is then run several times while inputs are generated from the probability distributions identified in the previous step. The sensitivity is either assessed manually through scatter plots or through various sensitivity measures based on the distribution of outputs while fixing particular input values (Saltelli et al., 2008). This method is hardly accessible for non-expert users and requires significant computational effort as the model has to be computed many times.

Different approaches towards sensitivity analysis are distinguished by Saltelli et al. (2004). One major distinction is between global methods, which span the entire input parameter space of the model and local analysis which tries to compute partial derivatives of the outputs with respect to the inputs to determine local model sensitivity (Homma et al., 1996). In so called one factor at a time (OAT) analysis, one input is modified and the output changes are determined. In regression analysis, parametric curves are fitted into the model results whose coefficients give an understanding of the related sensitivity.

Campolongo et al. (2007) argue that the OAT approach is the most common form of sensitivity analysis. According to the authors, OAT minimizes computational

effort, but lacks sophistication. The authors note that especially global methods for sensitivity analysis require significant computational resources. To this day, quantitative methods for sensitivity analysis can only be applied to low dimensional models due to the high computational cost associated with them (Iooss et al., 2015).

Scenario analysis

Scenario analysis examines the uncertainty of future developments by devising alternative scenarios of what might happen (Huss, 1988; Ringland, 1998; Von Reibnitz, 1988). In this sense, scenario analysis does not try to draft one consistent picture of the future. It rather presents possible future developments, which are sometimes called alternative worlds. Scenario analysis does not extrapolate from past data but rather attempts to draw alternative pictures of the future based on core changes and developments (Postma et al., 2005). Swart et al. (2004) argue that quantitative scenario analysis should be complemented by qualitative scenario exploration in order to identify additional influence factors and hard measurable factors.

Real options analysis

Real options analysis (Myers, 1977) is a method to evaluate different action alternatives in a decision problem, in particular if these action alternatives occur in the future. Real option in this context refers to the possibility of taking a specific action or decision, such as expansion of operations or stopping of projects. This reflects the possibility for management to react to changes in the market environment. Real options analysis is often contrasted with the static consideration of capital budgeting methods such as NPV. The application of real options analysis typically leads to higher NPVs, in particular for projects with higher flexibility. Luehrman (1998) relates these higher NPVs to the value of deferring decisions, the related interest on unused capital and the additional information gained by the delay regarding the economic soundness of the option. In this context, future

action alternatives are valued using an analogy from finance utilizing the Black-Scholes model for pricing of options (Black et al., 1973) with NPV and variance as inputs. Borison (2005) distinguishes different approaches of real options analysis and criticizes that underlying assumptions and conditions are very different and often not named explicitly.

Beta analysis

In beta analysis, Beta is a measure for volatility in quantities, e.g. a model output, compared to a benchmark (Sharpe, 1963). Higher beta values than 1 indicate higher volatility compared to the benchmark. Therefore, beta analysis allows assessing systematic risk associated with that quantity. Beta analysis is used in investment portfolio risk analysis where the performance of financial markets is used as a benchmark.

Beta is computed from least square regression analysis (Tofallis, 2008). Hence, beta analysis evaluates the risk associated with a quantity by comparing its past volatility with respect to a benchmark. This also means that new information is not taken into account as beta is purely derived from past data. Furthermore, the relative nature of beta is useful when comparing investment options, but does not lead to absolute statements regarding cost effectiveness.

Value at risk

Value at risk (VaR) is a measure that specifies the amount of money that an investment portfolio might lose over a specified time horizon with a given probability. It is a statistical measure to quantify potential loss of a portfolio (Linsmeier et al., 2000). The observed time horizon or holding period and the risk level are decisive for VaR measures. The holding period is the time horizon over which the loss is computed under the assumption that the positions in the portfolio are held, i.e. not adapted to market movements. The risk level is the probability for the losses of the portfolio over the holding period to exceed VaR.

VaR is often used in finance for risk management, but also sometimes used in non-finance applications (McNeil et al., 2005). VaR was proliferated through the need of monitoring a company's risk in times of increased complexity in the financial market due to frequent currency fluctuations and interest changes. VaR is traditionally computed with one of three methods, either Historical Simulation, Delta-normal Method or Monte-Carlo Simulation (Linsmeier et al., 2000).

2.2.3 Mathematical descriptions of uncertainties

The above outlined methods for analysis of uncertainty offer guidance on how uncertainties in models can be determined and assessed. As a foundation, they require mathematical frameworks for the description of uncertainties. Furthermore, these mathematical foundations are required in order to process uncertainties in different models and input quantities in order to determining the combined uncertainty of synthesized models. These mathematical descriptions are introduced in the following.

Probability theory

Probability theory is concerned with analyzing random, non-deterministic phenomena. It is closely related to the handling of aleatory uncertainties. Probability analysis is based on the analysis of historic data in order to predict future development assuming a stable environment. It uses random variables, i.e. variables whose value depends on possible outcomes.

The concepts of expected value and variance are central to probability theory and its mathematical description. The expected value of a random variable is the average of results if many experiments are carried out. The variance of a random variable is a measure of how far the results spread from the expected value.

Two theorems are of high importance for probability theory. The one theorem called the law of large numbers states that the average value of observations will approach the expected value when a large number of experiments is carried out.

This theorem thus states that conclusions regarding probability can be made from large samples of experiments. The other theorem called the central limit theorem states that the sum of independent and identically distributed random variables will approach the normal distribution, if variance and expected value are well defined. This holds even if the added random variables themselves are not normally distributed. It can be inferred from the central limit value theorem that a description of uncertainties in costing models using normal distributions is possible under the assumption that the related random variables are independent and identically distributed. The normal distribution of a random variable depending on other random variables can be computed under the assumption of independence of the input variables with defined rules (Stahel, 2002). This allows the propagation of probabilities through a computation model.

Possibility theory

Possibility theory is an alternative to probability theory in dealing with uncertainties. Like evidence theory (see below), it utilizes two concepts to describe likelihood: necessity and possibility (Helton et al., 2004). However, while evidence theory is closely related to probability theory, possibility theory is closely related to fuzzy sets and uses membership functions to assign possibilities to classifications (Zadeh, 1978).

While probability theory only uses one set to describe uncertainty, possibility theory uses two concepts to describe the likelihood of an event A (Dubois, 2006). The possibility Π is the maxitive set-function and expresses in how far a certain state of affairs is possible in the eye of an agent assessing potential states of affairs. A possibility Π of 0 means that a state is rejected as impossible while a possibility Π of 1 means that the state is assessed to be totally plausible. If the possibility of all states is assumed to be 1, this expresses complete ignorance. Necessity is the minitive set-function and defined as $N(A) = 1 - \Pi(\bar{A})$, where \bar{A} is the complement of A . Hence, the concept necessity contains the information of the possibility of not A regarding the occurrence of A . For example if $\Pi(\bar{A}) = 0$, i.e. not A is

impossible, implies that A is certain, hence $N(A) = 1$. In general, it holds true that $N(A) \leq \Pi(A)$ (Smets et al., 1998).

It has been realized that fuzzy sets, as an application of possibility theory, are an extension of interval analysis (Dubois et al., 2000). In particular, the usage of methods from interval analysis in fuzzy set theory for handling overestimation has been an area of research (Moore et al., 2003). Possibility theory has been applied to capital budgeting decisions. Tsao (2012) uses possibility theory to compute uncertain NPV of investments where the uncertainty is derived from linguistic expressions linked to fuzzy sets. Chiu et al. (1994) use triangular fuzzy sets in order to capture the uncertainty in cash flows and compute a fuzzy NPV. They further explore different decision-making strategies for project selection based on the fuzzy NPV. Damghani et al. (2011) developed a decision support system for investment decisions taking into account uncertainty. Fuzzy sets are used to express uncertainty in key parameters of different investment options and solve the associated capital budgeting problem. Huang (2008) uses fuzzy numbers in order to model uncertainty in investment sum, cash-flows and available capital, and studies the resulting capital budgeting problem. Based on that, a decision model for capital allocation is proposed.

Bayesian probability

Bayesian probability is an interpretation of probability as the degree of belief associated with a hypothesis (De Finetti, 2017). Bayes' theorem is central to Bayesian probability. It allows to update the probability assigned to a hypothesis in the light of new information in a process called Bayesian inference. Bayes' theorem reads

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}, \text{ where:} \quad (2.2)$$

- A is the hypothesis under scrutiny,
- P is the probability function,

- $P(A|B)$ is the probability of the hypothesis (A) in the light of new information (B) and hence the result of interest, also called posterior probability,
- $P(B|A)$ is the probability of observing the new information (B) given that the hypothesis holds,
- $P(A)$ is the base probability before new information is taken into account, also called prior probability,
- $P(B) \neq 0$ is the probability of the observation, also called marginal likelihood or model evidence.

Two views on Bayesian probability, the objectivist and subjectivist view, exist in literature (Thompson, 1990). The subjectivist view interprets Bayesian probability as personal belief in the respective hypothesis. The objectivist view interprets Bayesian probability based on relative frequency as an extension of logic.

According to Thrun (2002), Bayesian filters are the most common approach in probabilistic robotics in particular for problems like state estimation where hypotheses regarding the current state of the robot have to be continuously combined with sensor readings.

Evidence theory

Evidence theory (Dempster, 1967), also referred to as Dempster-Shafer Theory or the theory of belief functions, establishes a general framework for reasoning with uncertainties. In evidence theory, lower (belief) and upper (plausibility) bounds for the truth of a statement are established (Helton et al., 2004). The main properties of evidence theory according to Beynon et al. (2000) are that the belief not given to a proposition does not need to be attributed to its negation and that measures of uncertainty can be assigned to overlapping sets. Evidence theory can be seen as a generalization of Bayes' theory. Xu et al. (2006) hence report that evidence theory is more flexible in decision making compared to traditional Bayes' theory. Hilhorst et al. (2008) report the use of evidence theory for the selection of an IT-infrastructure. The authors show that the application of evidence theory

allows for the incorporation of non-financial criteria in the decision process leading to different decisions compared to application of pure real options analysis or NPV.

Interval analysis

Interval analysis is an extension of arithmetics of real numbers \mathbb{R} to the treatment of interval numbers \mathbb{I} . An interval number x_I is specified by a lower bound a (infimum) and an upper bound b (supremum) as

$$x_I = [a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}. \quad (2.3)$$

As for example outlined by Daumas et al. (2009), the basic laws in interval analysis of two interval numbers $x_I = [a, b]$ and $y_I = [c, d]$ can be described as

$$\text{Summation:} \quad x_I + y_I = [a + c, b + d] \quad (2.4)$$

$$\text{Differentiation:} \quad x_I - y_I = [a - d, b - c] \quad (2.5)$$

$$\text{Multiplication:} \quad x_I y_I = [\min(ac, ad, bc, bd), \max(ac, ad, bc, bd)] \quad (2.6)$$

$$\text{Division:} \quad x_I / y_I = [a, b][1/d, 1/c], \text{ if } 0 \notin [c, d]. \quad (2.7)$$

Exponential functions x_I^z with $x = [a, b]$ and $z \in \mathbb{R}$ can be computed as

$$x_I^z = [\min(a^z, b^z), \max(a^z, b^z)]. \quad (2.8)$$

For strictly monotonous increasing functions, the computation of their interval value is trivial, for example

$$\text{Natural logarithm:} \quad \ln(x_I) = [\ln(a), \ln(b)], \quad (2.9)$$

$$\text{Square root:} \quad \sqrt{x_I} = [\sqrt{a}, \sqrt{b}]. \quad (2.10)$$

Interval arithmetic guarantees that if the real input values lie inside the specified input intervals also the result will lie inside of the output interval obtained through interval arithmetic. Therefore, this approach allows obtaining strict bounds for best and worst case economic performance under the assumption that the expert setting the input intervals has chosen the right interval bounds.

When applying interval arithmetics to equations using above rules, great care has to be taken in case of multiple occurrences of the same variable in one equation. If these cases are not explicitly handled, excessively conservative bounds result as different occurrences of the variable are interpreted as different interval numbers. This effect is known as interval identity or overestimation. For example, consider the square of an interval variable $y_I = x_I^2$. This can be interpreted as $y_I = x_I^2 = x_I x_I$. When using this interpretation, applying equation (2.6) and inserting $x_I = [-1, 1]$, the resulting bounds of y_I are $[-1, 1]$. This result is excessively conservative as the worst case combination of bounds implied by equation (2.6) is not possible in this case as x_I is multiplied with itself. The better result interval is $y_I = [0, 1]$ as this reflects smallest interval that contains all possible result values.

Interval analysis is widely applied in computational numerics and kinematics, see for example Jaulin (2001) or Pott (2007). Matthews et al. (1990) report the use of interval analysis for the assessment of different investment options in utility economic analysis. Jerrell (1997) proposes the use of interval arithmetics to capture the effects of uncertainty on economic input-output models that describe the interdependencies between different branches of the economy. The approach is judged particularly useful when an assessment using confidence intervals is not possible.

Different authors report on the use of interval numbers instead of exact real numbers in order to reflect uncertainty. Mustajoki et al. (2006) apply intervals to multi-attribute value tree analysis (MAVT) to account for uncertainties of differential weighting of objectives and values of rating functions. They point out that great care has to be taken in choosing meaningful interval boundaries to receive meaningful results. Their approach allows to carry out strict worst case analyses.

Dietz et al. (2013a) propose to use a combination of interval arithmetics and ABC to model the cost creation in the engineering design of robotic systems. This approach is extended by Dietz et al. (2015) to use information structured according to the PPR-approach.

Authors in risk management point out that interval theory is fundamentally different from probabilistic approaches, as it does not contain a structured representation of likelihood. This is also the reason that the results of interval analysis are easy to represent as misunderstandings regarding the underlying concepts of likelihood are avoided.

2.2.4 Conclusions on modeling of uncertainties

Methodical frameworks for analyzing uncertainty, like for example Monte-Carlo methods, often rely on repeated execution of the analyzed models under variation of input parameters. This threatens to slow down analysis and hence to make it less reactive for the user. It exists a wide range of methodical frameworks and mathematical descriptions for uncertainties. These frameworks and descriptions are highly applicable to this thesis and offer a sound foundation for the description of uncertainties in the cost-benefit assessment in industrial robotics. The theories of probability, Bayesian probability, possibility and evidence offer highly expressive mathematical descriptions. However, practitioners in robotics and production, like for example managers and planners, might have problems with specifying the required parameters and with interpreting the results as these measures require some level of knowledge of the underlying theories. On the contrary, interval arithmetics offers a less expressive description that can be intuitively understood by practitioners in robotics. With interval arithmetics great care has to be taken to avoid overestimation. Overestimation might lead to overly conservative results that prevent any meaningful analysis. Due to the intuitive nature of interval arithmetics and the straightforward integration into existing models, interval arithmetics appears to be a very interesting choice for the representation of uncertainties in cost-benefit assessment of robot systems.

2.3 Cost modeling in engineering design

This section gives an overview of approaches relating to cost engineering in product development and development of automation systems.

2.3.1 Costing methods in engineering and product development

Several components of a robot system require detailed mechanical and electrical engineering. This includes in particular the mechanical and electrical design of the entire work cell as well as custom-made components such as end-effectors, fixtures, jigs, and material feeding solutions. The development and engineering processes are closely related to product development for which several costing approaches exist.

The issue of cost assessment has been actively studied in particular from the product point of view. Layer et al. (2002) differentiate between quantitative and qualitative approaches. Qualitative approaches are capable of ranking different options. However, they are not capable of quantifying how much a certain option is superior to an alternative. Quantitative approaches establish some metrics to compare different decision options in product design. Roy et al. (2001) report in an example from the aerospace industry that a combination of quantitative and qualitative approaches leads to better results than employing one of these approaches individually. This holds in particular true if also indirect development cost, for example for discussing a design problem with colleagues, should be included in the assessment. As this thesis aims at quantifying benefits from investments in automation and neglects distal-monetary factors, it focuses on quantitative approaches.

Asiedu et al. (1998) distinguish three types of quantitative costing approaches. Firstly, statistical models take a top-down approach, building on past data and employ data analysis techniques such as regression analysis to arrive at parametric cost functions. Secondly, analogous models derive cost information based on

the degree of similarity between product characteristics. Analogous models hence also build on past data from comparable products. Thirdly, generative-analytical models take a bottom-up approach by trying to model the product creation process and the related resource consumption. Amongst these three quantitative approaches, only generative-analytical models offer the possibility to easily add new processes without the need of historic data (Layer et al., 2002) as desired for robot systems. Furthermore, a lack of integration in the design process is cited which can also be considered true in plant engineering.

Agyapong-Kodua et al. (2011) state that main shortcomings of current cost models are the lack of focus on the process side, i.e. the manufacturing required to produce a specific product feature. Furthermore, they state that cost models need to be virtually executable to support decision making and need to be able to capture a wider range of products due to the increasing product spectrum in manufacturing. They suggest that cost models are required that integrate aspects of cost engineering, enterprise cost modeling, discrete event modeling and dynamic modeling. Quintana et al. (2011) apply the statistical costing approach to decision making in machine tool selection. The addressed market of vertical high speed machine tools is much more homogeneous compared to the market of robot systems, because the latter are special machinery.

Niazi et al. (2005) developed a decision model for cost estimation in product design. In this context, analytic models are of interest for the design of production systems, as they link product properties to manufacturing processes and the employed resources. Tseng et al. (2000) suggest a combination of feature-based costing and ABC. With feature-based costing, a specified manufacturing activity and its intensity are assigned to certain product features. ABC is used for modeling the cost of the required production processes and takes the values assigned to the product features as input. Koltai et al. (2000) apply the principles of ABC to FMS to cope with the allocation problem for high overhead costs of FMSs. Also process specific cost-models have been proposed, e.g. for machining (Ehmann et al., 1997) or welding (Masmoudi et al., 2007; Dolgui et al., 2004). The estimation of design effort for general development projects has been addressed by Bashir et

al. (1999), Pahl et al. (2007) and Chandrasegaran et al. (2013), but not specifically for robot systems.

A commonly found model for design efforts stemming from the mechatronic industry is the model proposed by Bashir et al. (2001) in which the design effort t_{wo} is expressed as a function of product complexity \mathcal{C} and severity of requirements \mathcal{R} :

$$t_{\text{wo}} = a \mathcal{C}^b \mathcal{R}^c, \quad (2.11)$$

where a , b and c are parameters of the model. The product complexity is expressed in terms of included functions and severity of requirements is rated, typically on a scale from 1 to 3.

The above analysis makes clear that many costing approaches in product development exist. Most approaches try to make manufacturing costs transparent to the design engineer in order to support design decisions, thereby leading to lower product cost. The developed models often focus on the production of mechanical features in order to support product design and manufacturing process selection. Significantly fewer research addresses the modeling of cost of the design process itself.

2.3.2 Estimation of software development efforts and times

Mechatronic systems and cyberphysical systems increasingly contain embedded software in order to implement complex functions and to ensure that the product can be updated and upgraded during its life-cycle. Likewise, the effort for developing software plays an increasingly important role in the overall cost of realizing robot systems.

Traditional, global estimation methods

The estimation of effort and cost in software development has been an active area of research for a long time. Different models are established in software cost estimation (Boehm, 1984; Boehm et al., 2000; Putnam, 1978a). The most commonly applied approaches are model-based techniques and methods based on expert judgment. While research on model-based methods dominated in the past, the interest in non-parametric models, more specifically expert judgment and analogy measures has increased (Jorgensen et al., 2007).

Methods based on expert judgment are in particular useful, when quantified data from past or comparable projects is missing. The downside of these cost estimation techniques is the high reliance on the experts' competence to make valid estimates. According to Boehm et al. (2000), the two most common expert judgment approaches are the Delphi method and the Work Breakdown Structure (WBS). The Delphi method (Helmer, 1965) is a method to produce forecasts based on the predictions of a panel of experts. In a first round, the experts give an estimate without discussion, independent of each other. The results are collected and distributed to all participants for a second estimation round in which all experts are aware of the estimates from the first round. The WBS method performs a breakdown of the software product to be developed. Breaking down the work means that the software product is decomposed into different single functional components, subcomponents and finally development activities. The aim of this breakdown is to obtain activities that are sufficiently small so the effort can then be estimated by an expert. From these estimates, the overall effort for the software development is then aggregated bottom-up (Tausworthe, 1979).

Model-based approaches typically use a measure for the amount of written code as an input. Common input measures are source lines of code (SLOC), delivered source instructions (DSI) or function points. These measures are then transformed into associated cost or a quantification of development effort. In the following, the details of some common model-based approaches are outlined.

Putnam's Software Life-cycle Model (SLIM) (Putnam, 1978b) specifies the software equation

$$n_{\text{SLOC}} = a_{\text{T}} t_{\text{wo}}^{1/3} t_{\text{dev}}, \quad (2.12)$$

where n_{SLOC} is the number of source lines of code, a_{T} is an adjusting factor for environment or technology factors that is derived from historic development data, t_{wo} is the development effort going into the project in terms of man years and t_{dev} is the delivery time or overall development time of the software project. While the model was originally aimed at the opposite task of estimating the size of software projects after development, Putnam's software equation can also be used for the estimation of development effort (Leung et al., 2002).

The well-established Constructive Cost Model (COCOMO) approach uses the delivered source instructions (DSI) as a cost driver. Furthermore, COCOMO distinguishes different types of projects based on their complexity of implementation. Small, informal projects with well-trained staff are referred to as organic project. Projects with medium complexity are called semi-detached projects. While the development of highly complex systems with formal project management procedures is referred to as embedded project. Automation software projects can typically be classified as organic projects as the overall complexity of the software package is rather low due to the local installation at the robot cell.

According to the COCOMO approach, the applied effort t_{wo} for the software project can be computed as

$$t_{\text{wo}} = a \left(\frac{n_{\text{DSI}}}{1000} \right)^b, \quad (2.13)$$

where n_{DSI} is the number of delivered source instructions, i.e. the number of lines of program code including comments and blank lines up to 25% of the code¹ and a as well as b are coefficients related to the respective project type.

¹The COCOMO model typically counts in thousands of delivered source instructions. Hence the division of n_{DSI} by 1000.

The development time for the project t_{dev} can be computed using the relation

$$t_{\text{dev}} = c(t_{\text{wo}})^d, \quad (2.14)$$

where c and d are constants to be found in literature. For organic projects, the following constants $a = 2.4$, $b = 1.05$, $c = 2.5$ and $d = 0.38$ are suggested by COCOMO (Boehm, 1984). The required number of developers n_{dev} can be computed from the development time and required effort $n_{\text{dev}} = t_{\text{wo}}/t_{\text{dev}}$. The COCOMO model does not account for team size and related management and communications efforts and can therefore only be seen as a rough estimate. As software development efforts for automation components and systems are typically carried out by rather small teams, this estimate appears to be a good starting point. Model-based approaches are applied in this thesis, as it is the aim to obtain quantified cost estimates without relying on expert estimates for the single knowledge domains involved in systems engineering. In particular, the good and verified base of parameter values for COCOMO methods is beneficial.

Estimation methods for agile software development

Agile software development methods have received strong attention in recent years and have been adopted by a large number of companies (Abrahamsson et al., 2009). The estimation of software development efforts for agile methods is carried out differently compared to traditional software development, because agile development methods are highly iterative (Cohn, 2005). Usman et al. (2014) report that in practice expert judgment methods are predominantly used for software development effort estimation of agile software projects. The most commonly applied parametric methods build on story points and use case points as size metrics.

Planning poker is an iterative group estimation technique in which the estimators individually present estimates for a certain user story and then discuss differences in their estimates until they converge (Cohn, 2005). One study (Mahnič et al.,

2012) indicated that estimates obtained through planning poker tend to be over-optimistic if the estimation is carried out by non-experts. In this study, the accuracy of the estimates was found to improve with the level of expertise of the estimators.

Another common method for estimation of software development efforts in agile software development is the use case point method (Schneider et al., 2001). The use case points are similar to functional points used in traditional software effort estimation techniques. The use case point method employs different weighting factors for use case complexity, number of actors involved in the use case, the technical complexity and environmental considerations. Kusumoto et al. (2004) developed an automatic use case measurement tool that allows automating large parts of use case estimation. This measurement tool assesses use cases specified by the user in XML. The assessments are based on an experience database.

Abrahamsson et al. (2007) proposed the use of an incremental estimation model for estimation of development efforts in extreme programming (XP). This estimation model is built after every iteration in the agile process and uses information from previous iterations to improve accuracy. The developed estimation model outperformed traditional estimation models. This was in particular true in early development phases, which also corresponds to the intended use case scenarios of the cost-benefit assessment system researched here.

Bhalerao et al. (2009) compare estimation methods targeted at traditional and agile development. They argue that both types of methods use the same type of quantitative information for estimation. According to the authors most estimation methods targeted at agile software development are too much based on experience. Further, an algorithm for the estimation of software development efforts in agile development called Constructive Agile Estimation Algorithm (CAEA) is proposed. This algorithm evaluates the project based on different criteria (domain, configuration, performance, complexity of processing, data transactions, ease of operation, number of sites and security).

The research on software effort estimation techniques for agile development pro-

cesses is a relatively young area of research. In comparison to traditional software effort estimation approaches the discussed cost estimation approaches are much more integrated into the development process and become a part of the agile development itself. The experience based nature of most current approaches makes them harder to integrate into an automated cost-benefit assessment. Furthermore, many automation software projects for application software development by the system integrator are very small projects that often do not have any formalized development process at all. However, as software becomes an increasingly important part of robot systems it will become an interesting research questions how those estimation methods can be integrated into cost-benefit assessment.

2.3.3 Development process for robot systems

Development of special machinery is a complex process in which many, often contradicting and hidden requirements need to be traded off in order to obtain a good solution. One major problem is the management of the inherent complexity of this development process. Systems engineering offers processes to cope with this complexity.

The design of a robot system typically starts with a conceptual phase in which the functions and related solution principles are defined. The conceptual design of the robot system is vital as the layout and components of the machine are defined in this phase. The design often follows processes from systems engineering, such as VDI 2221 (VDI 2221, 1993). It includes the definition of functions, the definition of solution principles for these functions, the overall concept including the planning of the layout and used components to be integrated. During the conceptual design phase, feasibility studies in simulation or experiment might be required to validate critical functionality.

The concept design phase is a vital part of the creation of robot systems and highly depends on the novelty of the system and the experience of the integrator with similar systems. The Genefke Scale (Nielsen, 2012) attempts to measure these aspects related to robot system complexity. It measures the complexity of a

robot installation in five categories from standard solution to cutting edge research application. In the context of industrial applications, the first three categories of the Genefke Scale are of interest:

Category 1: Off-the-shelf solutions with single robots in which the robot does not carry out an actual manufacturing process, i.e. handling robots (see Section 3.1.2 for an example).

Category 2: Standard solutions that require adaptation to the specific tasks, multiple robots and process robots, i.e. a welding robot installation (see Section 3.1.1 for an example).

Category 3: Specialized solutions that have never been realized in a similar setup comprising all technological elements and require demonstration of core components.

Concluding, systems engineering offers a good methodical framework for the development of complex systems to which robot systems belong. However, there appear to be few development process frameworks that are specifically tailored for the development of robot systems. The Genefke Scale offers a useful taxonomy with respect to the complexity of the realization task of a robot system.

2.4 Modeling of manufacturing systems

One key problem for the introduction of elaborate costing methods such as ABC is that they need to be tailored for the specific application case and the relevant data needs to be available (Staubus, 1990). The largest part of the system cost is defined in the early engineering phases before the system is set-up. If that assumption holds true, a lot of cost relevant data is already implicitly defined in the engineering models of the production system. The aim here is building a bridge between these engineering models and the cost model of the manufacturing system. For this reason, the state of the art for engineering models is relevant to this thesis and outlined in the following.

2.4.1 Information models used in plant engineering

Cost engineering in automation is closely linked to plant engineering as these two processes constitute an iterative cycle for optimization of plant designs. The cost engineering process should be as closely as possible integrated in plant design in order to provide effective decision support. For this purpose, it needs to draw information from and provide information to engineering formats in plant engineering, i.e. common, machine readable file formats used in plant engineering for storing and exchanging data. For the structuring of manufacturing systems, the so called three view concept, or PPR-concept is commonly applied in practice. In the three view concept the data on the manufacturing system is structured into data relevant to products, processes, and resources (PPR) (Schleipen et al., 2009):

- The product domain, i.e. what is to be manufactured
- The process domain, i.e. the actual manufacturing activity that is carried out on the work pieces that are to form the product
- The resource domain, i.e. the means of production that are employed to carry out the manufacturing task

AutomationML is a format dedicated to exchange of data between different plant engineering tools, e.g. CAD, PLM and electric circuit design. The AutomationML file format (AutomationML, 2013; Draht, 2010) is built on the PPR-approach. The format uses CAEX (IEC 62424, 2016) for capturing the hierarchical structure of the production system and its components, COLLADA for geometric data and PLCopen XML for the description of the logic information. AutomationML features role libraries to define abstract characteristics that are described by the CAEX file and to give the described objects a semantic meaning.

An alternative standard for the exchange of product information is the Standard for the Exchange of Product Model Data (STEP), which is specified in ISO 10303 (ISO 10303, 2011). It supports the modeling of data from mechanical and electrical design including tolerances, machine topology, material properties,

and process structure. However, STEP does not offer facilities to model the behavior of machinery. The sheer size of the standard resulted in fragmented implementations.

Process specification language (PSL) is a high-level framework for the description of processes. It is applicable for different domains, for example for manufacturing and business processes, but was developed with a strong focus on manufacturing processes. PSL is standardized in ISO 18629. The main aims for its development are supporting communication of process data between different software systems and supporting automatic reasoning on processes, for instance for the evaluation of process consistency and compliance to process rules. Logic elements for the process description are defined in an ontology and support automatic reasoning.

It can be concluded that powerful information models and related file formats for the modeling of manufacturing systems have been developed in the past. In fact, the developed information models and file formats are so broad that few or no tools exist that are feature complete with respect to the theoretical possibilities of the file formats. These models and file formats are an important starting point for the modeling of information on robot systems in this thesis.

2.4.2 Modeling of work efforts and resource consumption in production

In order to reflect the cost in the entire life-cycle of a robot system, the necessary resources for the usage of the robot system have to be modeled. This refers to the manual work efforts required for certain production tasks and to the consumption of commodities and energy, for example electricity consumption or consumption of process media.

Elaborate models exist for the design of manual workplaces in order to estimate times required for certain production activities. Predetermined motion time systems (PMTS) are methods designed to analyze manual work processes in industrial production. PMTS methods typically use catalogs with standard times in

order to derive an estimate for the required work times. Methods-Time Measurement (MTM) is the most widely used PMTS method. Several variations of MTM exist. MTM Universelles Analysier-System (MTM-UAS) is the most recently developed variation. Compared to standard MTM, the MTM-UAS method uses a coarser level of aggregation centered around complete operations rather than fundamental motions. Maynard Operation Sequence Technique (MOST) is another commonly used PMTS method operating with time measurement units. Paul et al. (1979) applied the general approach of the MTM method to robotic applications and hence created Robot Time Measurement (RTM). A similar approach was taken by Wygant et al. (1987) who created Robot MOST, which measures time of robotic processes using MOST. The set of methods published and researched by the German Verband für Arbeitsgestaltung, Betriebsorganisation und Unternehmensentwicklung (REFA) is targeted at determining nominal work times from time measurements taken in the real production. The method is part of many labor agreement contracts in Germany.

Apart from work efforts, also the behavior and consumption of energy and commodities caused by the means of production employed in the robot system need to be modeled. The physical basics of modeling manufacturing systems such as kinematics and dynamics are well known and respective models can be found in standard literature. In the context of cost modeling, in particular models relating to the production output of the employed machine and models related to the consumption of resources are relevant. Dietmair et al. (2008) analyze the energy consumption of machine tools and develop a model for forecasting the energy consumption related to a specific production task. The study shows that machine tools consume a significant amount of energy even if the machine is not actually processing parts. Therefore, the overall energy efficiency highly depends on the usage scenario of the machine, i.e. how many and what type of work pieces are processed. Chemnitz et al. (2011) conducted a study on the energy efficiency of industrial robots using two different robots. The study shows that the energy consumption characteristics are highly dependent on the robot model. Furthermore, the energy consumption is highly trajectory dependent. Slow trajectories do

not necessarily have an optimal energy consumption compared to more dynamic trajectory profiles.

Existing models allow simulating the consumption of basic resources such as work time or electric energy during production. This allows determining an important cause of cost for the life-cycle assessment of production systems.

2.4.3 Knowledge management in robotics

An important challenge for solving the addressed research problem is finding methods to effectively manage cost relevant knowledge from different stakeholders participating in the development, commissioning, and operation of industrial robot systems. This requires to autonomously integrate information from different sources and to reason based on this information. This section introduces related methods for knowledge management as commonly used in robotics research. Normally, the usage of the introduced methods aims at realizing additional technical features in robot systems. Knowledge management methods are in particular used for achieving a greater degree of autonomy of the robot system. Here, these methods will be applied to the task of cost estimation in robotics.

Nowadays, ontologies are widely used in robotics research in order to model knowledge in human and machine readable form. What sets the application of ontologies apart from common industrial engineering tools is the modeling of knowledge by pure logic without the creation of artificial constraints. Domain specific ontologies are available for service robotics in the KnowRob ontology as described by Tenorth et al. (2013) and others (Lim et al., 2011; Chella et al., 2002). The IEEE currently develops a set of ontologies targeted at service and industrial robotics and automation (Schlenoff et al., 2012; IEEE 1872-2015, 2015). Lohse (2006) designed a domain ontology and related method for automated planning of modular assembly systems. The wide usage of ontologies and related tools in robotics promises that a transfer of these methods to plant engineering and cost estimation is fruitful once an ontology for this domain will be created. Specifically,

ontologies as described in the following do not depend on XML or programming languages.

The foundation for the use of ontologies and reasoning are technologies from the Semantic Web, which are increasingly used in robotics as outlined by Stenmark et al. (2015). Commonly used examples are Resource Description Format (RDF) (W3C, 2004), Web Ontology Language (OWL) (W3C, 2012), and SPARQL Protocol and RDF Query Language (SPARQL) (W3C, 2008). The development of RDF, OWL and SPARQL has been coordinated by the World Wide Web Consortium (W3C) which aims at adding explicit semantic content to webpages (Berners-Lee et al., 2001; Shadbolt et al., 2006).

RDF represents knowledge as triplets, consisting of subjects, objects and predicates connecting subjects and objects. RDFS (Resource Description Framework Schema) (W3C, 2015) extends RDF and introduces basic concepts for ontologies such as the concept of classes and the relation of resources to these classes. RDF and RDFS are commonly used for semantic descriptions and the basis for the creation of ontologies, e.g. in the common OWL format. OWL introduces additional concepts for the representation of ontologies that are motivated from limitations in expressiveness of the formats RDF and RDFS on which OWL builds (Staab et al., 2009). However, this results in an undecidable language, because RDFS has some very powerful modeling primitives, allowing all possible combinations of OWL primitives. Therefore, the three sub-languages, OWL full, OWL DL, and OWL-lite were developed. While OWL full uses the full language capabilities, OWL DL and OWL-lite impose additional restrictions to ease reasoning. OWL is the technical foundation for reasoning in many research applications in robotics and one of the basic tools for increasing the autonomy of robot systems. The OWL formats allow defining concepts and their relation and therefore represent semantic information in computer readable form.

Different storage frameworks for RDF or comparable graph data exist. The Sesame framework (Sesame, 2013) contains a servlet that allows the storage and querying of RDF knowledge in a database. Other frameworks (Jena, 2015; Newman, 2015; CubicWeb, 2015; Neo4j, 2015) possess similar features. SPARQL is a

query language for RDF and the query language for the Semantic Web. It allows to query for triple patterns and features conjunction and disjunction. SPARQL supports federated queries, i.e. queries that are distributed to multiple endpoints with gathering of the results from these endpoints. Through this feature, SPARQL allows the integration of knowledge from different data sources. While SPARQL is the principal query language for RDF (Bailey et al., 2009), alternatives exist. The RQL (Karvounarakis et al., 2003) language and its branch SeRQL allow both queries for data and schema. Furthermore, Algae, TRIPLE, and Xcerpt are languages that can be used to query RDF data according to Bailey et al. (2009). Dedicated query languages for OWL emerged more recently and are less developed. OWL-QL (Fikes et al., 2004) being the most known query language for OWL.

Several authors proposed using and tailoring technologies from the Semantic Web for applications in robotics. Persson et al. (2010) describe an approach to convert data from AutomationML to RDF triplets and make them accessible via SPARQL endpoints. Stenmark et al. (2013) report the potential for the use of knowledge bases from different projects and present different architectures for knowledge management. They favor a distributed approach to knowledge that allows to separate local and global knowledge. The authors point out open research issues in the fields of reliability, consistency and access. The same authors argue in favor of using knowledge bases to simplify the programming process of industrial robots. In their work, they outline the required ontologies and knowledge integration services for this application (Stenmark, 2017). Abele et al. (2013) transform CAEX plant models into OWL ontologies by the use of a base ontology following a suggestion by Runde et al. (2009) to perform consistency checks through SPARQL queries. La Rocca (2012) proposes the use of knowledge-based approaches in CAD design. The author argues that an information gap hinders the widespread application of these approaches in engineering practice. Kuss et al. (2016) explore currently used approaches for knowledge management in robotics and develop a concept for using manufacturing knowledge for automatic program generation. The authors propose to expand the PPR-approach through technology models that contain detailed information on physical properties of manufacturing pro-

cesses. This approach is detailed by Kuss et al. (2017) for gas metal arc welding and for automatic feature recognition of weld seams.

The use of ontologies and in particular OWL for their description form a very powerful basis for the management of knowledge in SME-like production scenarios. In particular the open-endedness and extensibility of ontologies allows to react to changing environments and production tasks as often found in SMEs. This approach conforms to the Open World Assumption (OWA) that complete knowledge cannot be obtained that holds true in dynamic SME-like production scenarios. The domain knowledge for the cost-benefit assessment of robot systems does not exist yet and has to be created.

2.5 Conclusions from the state of the art

Subsequently, the conclusions from the state of the art are drawn. The potential from using the presented methods for the cost-benefit assessment of robot systems and their shortcomings are discussed. The resulting gap to be filled to answer the addressed research problem is elaborated.

2.5.1 Methods from accounting and capital budgeting

Methods from accounting and capital expenditure budgeting offer holistic and methodically sound approaches for the identification of costs resulting from the purchase and use of automation systems. However, there are major shortcomings from applying these methods for decision-support in early design phases of robot systems:

- The methods do not offer details on how to compute actual costs in the single life-cycle phases of robot systems. The cost drivers and model details have to be tailored for each specific application field resulting in significant effort for the user. In particular, the methods do not include details on the aspect of system integration and the dependence of costs and benefits on

the technical properties of the robot system. However, this is vital because robot systems are typically special machinery and involve many design and integration activities.

- The methods require large amounts of input data. However, reliable cost data is often missing in SMEs and cannot be measured with reasonable effort. Furthermore, the required information is distributed over different stakeholders involved in design and operation of robot systems. The presented methods lack approaches for collecting and consolidating this distributed knowledge.

ABC offers a well-researched and established approach to distribute machine costs to single products. However, also ABC does not address the aspects of producing and maintaining the machine itself. Therefore, there is the need for a generic model for robot systems, that connects the life-cycle driven approaches from LCC with the activity-driven approach from ABC. This generic model has to include the handling of uncertainties and the management of knowledge from different stakeholders and projects.

2.5.2 Handling of uncertainties

Cost-benefit assessment is usually carried out in early design phases of the robot system where few information is available on the design of the robot system. Furthermore, the volatility in the order behavior of customers increases as product life-cycles become shorter and inventory is reduced. This lack of information and inherent volatility of the market cause uncertainties in input data to play a major role for cost assessment of robot systems. Different approaches based on probability, evidence, possibility, and interval theory are described in literature. The results of interval theory lack information on the likelihood distribution. However, results and required inputs of interval theory are most intuitively understandable for non-expert users when compared to other approaches of describing uncertainty. The mathematical foundations of all presented approaches are well-understood and application examples for a large variety of fields exist.

2.5.3 Existing engineering models

A wide body of research aimed towards modeling manufacturing costs of products exists. However, these models are typically highly specific to one domain, e.g. the machining of parts. While these model mostly address production cost, also models aimed at capturing development and engineering efforts exist. In particular the prediction of software development efforts has been an active area of research. There are several models that can also be used in the context of robot systems. The Genefke Scale offers a measure for classification of robot systems with respect to their implementation complexity.

Several file formats address the processing of engineering data in the design of products and automation systems. AutomationML is focused on the handling of plant design data in automation. The format is easily extensible and therefore predestined for the exchange of information between design tools and the cost-benefit assessment.

Only generative-analytical models are capable of modeling the details of the development process of robot systems. This is the case because robot systems, being special machinery, are heterogeneous in terms of application areas and features. This prohibits the use of analogous models and statistical models as past data is not significant in terms of the overall system cost. However, on the component level of the robot system statistical and analogous models might be applicable as the market is much more homogeneous.

Concluding, a large number of domain specific models exist and are available for integration into a common framework for cost-benefit modeling of robotic systems.

2.5.4 Knowledge management

Much research in knowledge management in robotics exists. This research typically builds on technologies of the Semantic Web in order to describe knowledge in

machine readable form. The Semantic Web technologies offer a powerful framework for representing and processing knowledge in machine and human readable form. Today, these technologies are mostly applied in robotics for extending the level of autonomy of robot systems. However, it seems possible to apply these approaches in order to automate parts of the cost-benefit assessment for a specific robot system. How to do that while including the missing system integration point of view is believed to be an important unsolved research problem.

3 Problem analysis

In this chapter, the problem of economics of advanced industrial robot systems is analyzed in detail with respect to the state of the art. The chapter aims at establishing a more detailed understanding of the addressed research problem, its single parts and the requirements on the solution. First, a set of exemplary application scenarios for the evaluation of the results is defined. These application scenarios reflect the specific problems and requirements identified in the following analysis. Subsequently, the scope of the thesis is defined and related terms that are used throughout the thesis are introduced. In the following, the typical life-cycle of robot systems is analyzed. Here, the processes that are characteristic to the different life-cycle phases are introduced and the involvement of the different stakeholders is discussed. Based on this analysis, the problem addressed by this thesis is defined and broken down into sub-problems. This problem statement also explicitly highlights the expected main contributions of this thesis.

3.1 Example applications for evaluation of the developed model

The evaluation of the cost-benefit assessment method developed here will be carried out based on two application scenarios. In this section, these two application scenarios are introduced. With handling and welding, these application scenarios span a significant portion of the market for industrial robots (Litzenberger, 2015). The two application scenarios also reflect two fundamentally different uses of robots to be captured by the cost-benefit assessment tool:

1. Use of robots for the main manufacturing operation: In the welding scenario, the value creating activity is carried out by the robot. The processing speed and the part output of the robot system therefore are primarily defined by the performance characteristics of the robot, its peripherals and the welding process.
2. Use of robots in production with fixed machine cycle time: In the handling scenario, the cycle time of the robot system is governed by the machine tended by the robot. The robot's operation is synchronized to the machine operation. Therefore, in this scenario the production output of the robot system is only secondarily defined by the performance characteristics of the robot system.

3.1.1 Robotic GMAW welding

Gas metal arc welding (GMAW) is a widespread joining process and applied in many industrial sectors such as building construction, ship building, or car manufacturing. GMAW is a very universal process that can be performed for different parts using the same process tool. However, usually specialized fixtures are required that are typically more complex when using welding robots compared to manual welding.

The overall process is highly complex involving several process steps and diverse technologies, such as producing the weld joint, i.e. the welding process itself, finishing, and machining operations for intermediate processing of the weld joint. Preparatory tasks, such as tack assembly and the insertion into the jig, are out of scope of the example. These tasks are assumed to be neutral with respect to manual and automatic execution.

The analyzed work cell is a welding work cell for SMEs. The cell consists of a steel support structure on which a 32 kg payload robot, a two-axis work piece positioner, a GMAW welding source, the control cabinet for the robot, an air filtering system, and protective fencing are mounted (Figure 3.1). The work cell is highly standardized, but welding itself is a complex process requiring accurate

3.1 Example applications for evaluation of the developed model

path following and parameter adjustment. Therefore, the robot system is a system of category 2 on the Genefke Scale (see Section 2.3.3). For the GMAW use case, the cost savings for the replacement of manual labor through a robot system compared with the upfront investment cost are to be analyzed by the cost-benefit assessment tool.

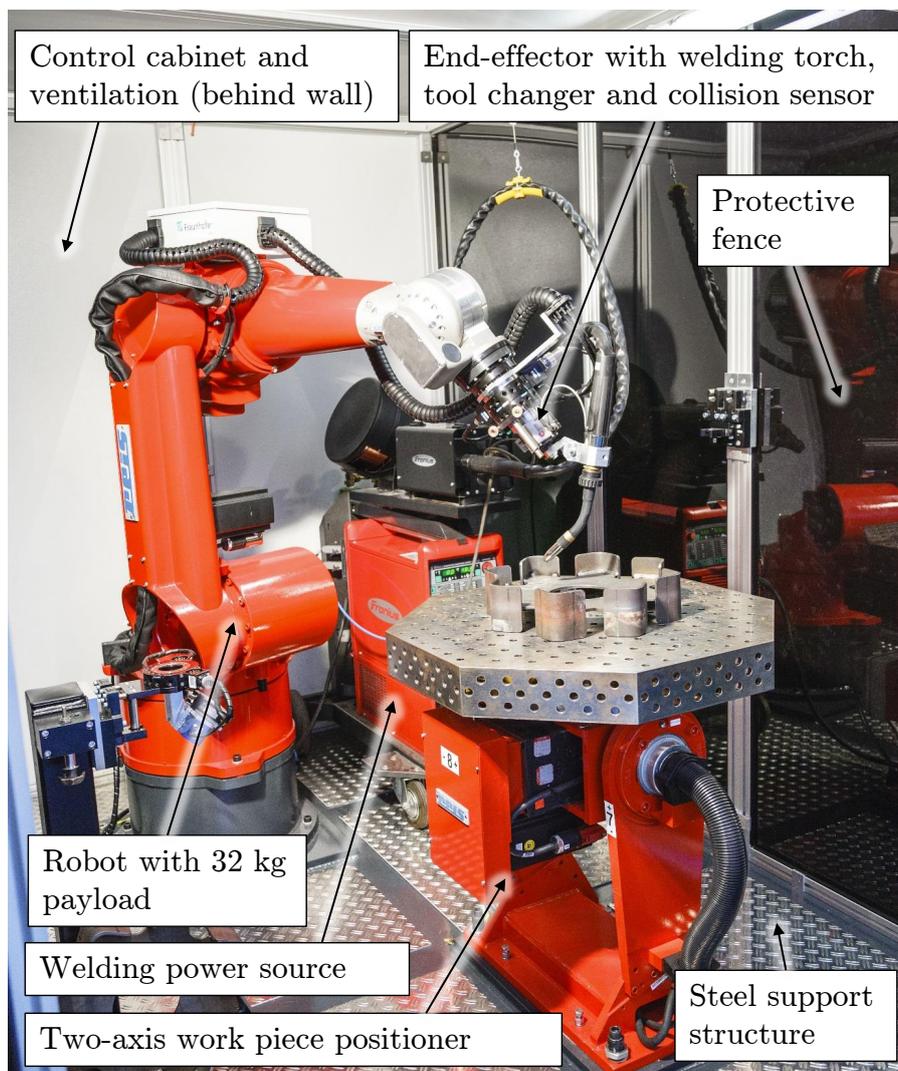


Figure 3.1 Analyzed robot work cell for welding example.

3.1.2 Machine tending robot for work piece handling

Handling applications are by far the most widespread application for industrial robots. In the year 2014, 49% of the shipped robots and 43% of the operational

3 Problem analysis

stock of robots were used in handling applications (Litzenberger, 2015). A handling application is analyzed in this thesis, because of this high importance in terms of market share. Most handling robots are used in packaging goods or feeding and unloading parts from production machinery or fixtures. The latter case of machine tending is the example selected here.

The analyzed work cell feeds cast shafts to a machining center for post-processing. The shafts are placed by a human worker in a pattern table (low-cost automation device). The robot picks the parts from there and inserts them into the machine. The collet chuck of the tended machine has self-centering properties, hence a highly accurate placement of the parts by the robot is not required. The cycle time of the robot system is determined by the process of the machining center and the robot is synchronized to this cycle.

The robot system requires very little application-specific adaptation. Therefore, it is considered to be a system of category 1 on the Genefke Scale (see Section 2.3.3). Figure 3.2 shows the analyzed work cell without the machining center.

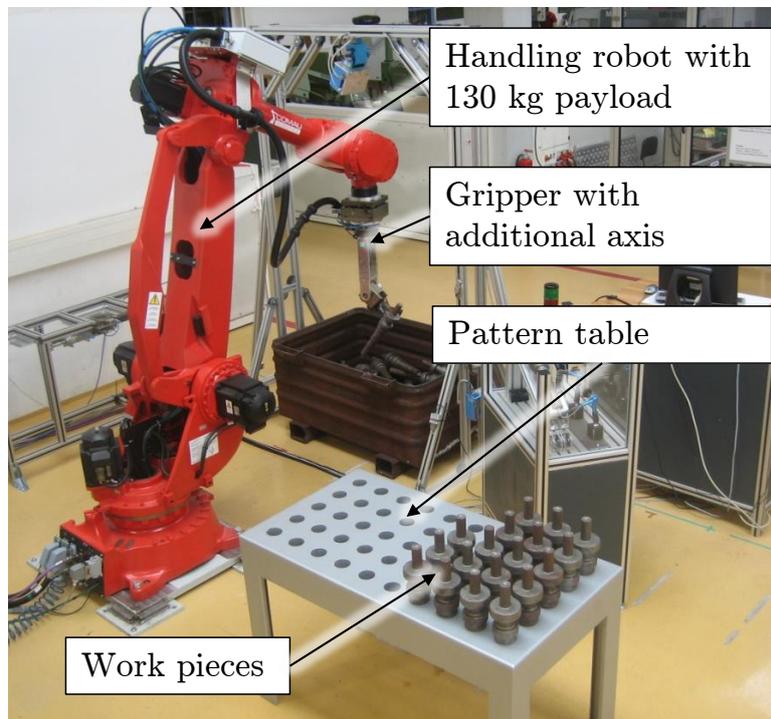


Figure 3.2 Analyzed robot work cell for machine tending example.

The main task for cost-benefit assessment of the handling work cell is to estimate the realization costs of the robot system and comparing this cost to the reduction

of manual labor. Additionally, the fulfillment of the cycle time of the robot system needs to be checked to ensure that the automation solution is not the rate-determining step. The resource consumption of the robot system, in particular electric and pneumatic energy, is expected to be negligible compared to personnel cost and maintenance cost of the robot system. Nevertheless, the cost-benefit assessment tool should be capable of computing these efforts and will take them into account.

3.2 Scope of work and related terms

This thesis focuses on the cost effectiveness of robot systems. In this context, a robot system is defined as a machine used for industrial production involving at least one industrial robot according to ISO 8373. Robot systems are usually custom built, tailored to specific applications and hence special machinery. It is expected that results of this thesis can also be transferred to other automation systems.

The cost effectiveness of robot systems always needs to be evaluated with respect to the application of the robot system. The term application in this context refers to the life-time production task of a robot system. It is characterized by the manufactured products, the properties of these products, the executed process steps and the technologies employed in these steps, as well as the production volume and lot-sizes. The term product here refers to the actual product produced by the robot system during its use. In early life-cycle phases, such as design and integration, the term product refers to the robot system itself.

A process is defined here as a work task or sequence of work tasks adding value to a product. A process takes a certain amount of time for execution. An example is the joining of two parts through welding which might include the process steps: seam preparation, tack welding of the assembly, root welding, root weld curing, welding of the seams and finishing. Each of these single process steps employs specific technologies, e.g. gas metal arc welding (GMAW) for the root welding.

A technology in this context is a specific method involving a physical principle for modifying the product at single steps during manufacturing. A technology therefore does not include any time aspect. In order to carry out processes and provide the required capabilities for the involved technologies, an adequate set of resources is required. A resource refers to a machine, person or tool involved in carrying out one or several steps of the manufacturing process by applying one or several technologies to the product.

To understand specific aspects of the realization and the usage of robot systems, the different stakeholders in the market characteristic to the robotics industry have to be considered. A component supplier is a company that delivers commercial off-the-shelf (COTS) products with minor degrees of customer adaptation and limited integration support that are intended to be integrated into a robot system. The system integrator purchases automation components and integrates these components into a robot system, i.e. a machine for production use. The end-user deploys a robot system in order to use it as a resource for the manufacturing of products. The specific role of a company always has to be defined based on a concrete automation project. Some companies might act towards their customers as component suppliers or system integrators or both depending on the requirements of the customer in the project. Furthermore, a company whose main business model is the supply of automation components might act as a end-user in a project that concerns its own production.

3.3 Analysis of the life-cycle of robot systems

A wide range of factors have to be considered to evaluate performance of a robot system along its entire life-cycle. The life-cycle model for robot systems that is used throughout this thesis is introduced here. Typical tasks and the involvement of the different stakeholders as well as development tools are analyzed in the remainder of this section. The ability of the different stakeholders to contribute required information to the cost-benefit assessment is scrutinized in order

to identify how different sources of information should be integrated into the model.

3.3.1 Life-cycle model of the robot system

An assessment of the cost effectiveness of a robot system can only be useful when it includes a significant part of the system life-cycle. This is because a higher effort during conceptual design, development, and integration might pay-off very quickly in production due to improved performance or higher versatility of the resulting robot system. The lack of proper life-cycle assessment in many capital expenditure decisions in robotics leads to sub-optimal investment decisions. This problem is further exacerbated by the fragmented nature of the robotics market. A system integrator has the incentive of being contracted for the realization of an automation solution. Since competition usually is based on the purchase price of the system, this often leads to system solutions that have a low purchase price but are sub-optimal for the end-user in terms of the overall life-cycle cost. As will be shown in subsequent sections, this fragmentation also causes problems as the required information for cost-benefit assessment is distributed amongst component suppliers, system integrators and end-users. This requires the integration of different knowledge sources in order to obtain a closed assessment of cost and benefit.

Figure 3.3 shows the life-cycle model used throughout this thesis. The model is an adaptation of the life-cycle model for the machine industry proposed by VDMA 34160-06 for robot systems. The individual phases are explained and motivated in the following.

Development and integration of a robot system

During the development and integration phase, the automation project is set-up, the robot system is planned and components are purchased or developed. This life-cycle phase usually starts with first negotiations between end-user and

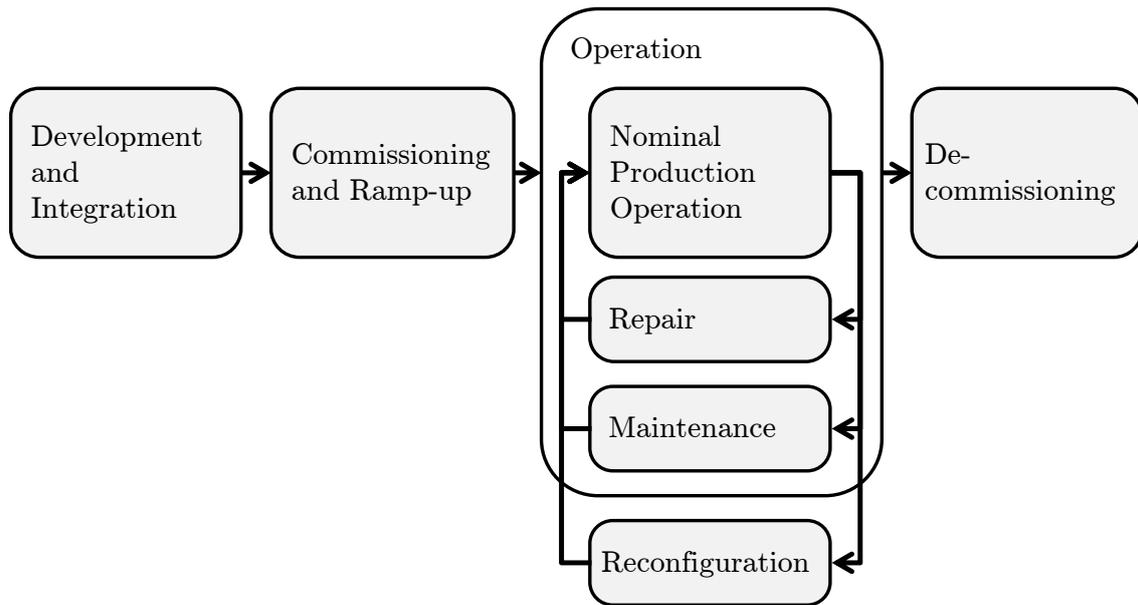


Figure 3.3 Used life-cycle model for robot systems (adapted from VDMA 34160-06).

system integrator and ends with the factory acceptance test (FAT) of the partly integrated system. Activities in this phase usually take place at the site of the system integrator before the robot system is sent to the production site of the end-user. Typical cost-relevant activities are engineering hours, labor hours for integration and software development efforts. These efforts mostly occur on the side of the system integrator who has the main responsibility for the project in this phase.

The development phase is decisive for the performance of the robot system over the entire life-cycle as the core properties of the robot system are defined here. The proposed cost-benefit model is mostly used to guide decisions in the development phase. Therefore, its usage has to be integrated with typical development processes. These development processes and the role of the different stakeholders in this phase are described in detail in Section 3.3.2.

Commissioning and ramp-up of production

The installation and ramp-up phase integrates the robot system into the production of the end-user. It starts with the installation of the automation equipment at the end-user's site. Further, it includes the initial operation of the complete system in the final production scenario and the ramp-up of production to the targeted production volumes. During the end of this life-cycle phase the benefits of using the robot system begin to materialize.

This phase is characterized by a gradual shift of responsibility for the robot system from the system integrator to the end-user. Likewise, the focus shifts from set-up cost to cost resulting from operation. Typical cost items occurring during installation and ramp-up are labor cost for integration and problem solving, for example for programming, interfacing with other machines in production, and program correction. Further, a transient decrease of the production output or increase of scrap might result from ramp-up of production, which induces additional direct cost and opportunity cost for the end-user. The high risk that the integration of the robot system might affect running production makes the ramp-up phase critical for the end-user. However, efforts and required time for problem solving are often not considered properly in cost calculations for robot systems. Another important cost factor during installation and ramp-up is the training of end-user's personnel for the operation of the robot system.

Operation of a robot system

In the operation phase, the robot system is used by the end-user in nominal production for a specific range of products. This life-cycle phase also includes changeover to other, already tested products that can be managed without the need to supplement the capabilities of the robot system. In this case, changeover can be handled by a machine operator or production worker.

During operation of the system, the benefits of using the robot system materialize. The increase in productivity, output, or quality causes cost savings that constitute

the benefit of using the robot system. The prior life-cycle phases were concerned with estimating the cost associated with building the robot system and bringing it into operation. These costs can be evaluated on an absolute basis, i.e. what efforts are required for bringing the robot system into production. For the evaluation of the benefits of the robot system in terms of cost savings a benchmark for comparison is required. This aspect will be further addressed during concept development for the cost-benefit assessment tool in Chapter 4.

Maintenance and repair

Regular inspections including maintenance activities keep the capabilities of the robot system intact. Repair restores the capabilities after breakdown. These activities interrupt the operation of the robot system. Cost considerations for maintenance and repair are significantly different from the nominal operation. This is the case, because during repair and maintenance the robot system becomes again the product of the executed work tasks. In this phase, the end-user typically also relies on support and services from an external partner such as the system integrator or the robot manufacturer. Maintenance and repair are therefore considered as a separate life-cycle phase. Furthermore, this phase has to be considered thoroughly as the machine will not be available for production resulting in additional opportunity costs. This fact might cause additional cost as the production of adjacent machines and processes might be interrupted as well.

Reconfiguration of a robot system

During the reconfiguration phase, the capabilities of the robot system are augmented to be able to cope with changed conditions, such as new types of products, new technologies, or changed production volume. Reconfiguration is usually carried out by the system integrator or a specialized department of the end-user. Therefore, it is not considered to be part of the nominal operation. Typical cost items are personnel cost for programming and redesign of fixtures and tools. Currently, reconfiguration is relatively rare, because the associated cost is often in a similar

range to purchasing a new robot system. But, due to developments in automated programming, cognition, and agile means of production, reconfiguration of existing robot systems is increasingly becoming cost-effective. Ease of reconfiguration is a key area of research and development for bringing the advantages of automation to new application areas and for maintaining the cost-effectiveness in existing areas which are subject to increasingly volatile market conditions.

Decommissioning

The robot system might become unusable for the end-user, e.g. because the product became obsolete and the use for a new type of product is no longer feasible, a newer machine with better cost-benefit ratio is available, or the robot system is worn out and does no longer fulfill the required quality standards. Consequently, the robot system will be decommissioned. This life-cycle phase is mentioned for completeness although it does not significantly contribute to the overall life-cycle cost of most robot systems. The reason is that robot systems typically neither contain hazardous components with expensive disposal nor include components that can be resold for a significant price. Decommissioning is therefore neglected in the following.

3.3.2 Involvement of stakeholders in the life-cycle

Industrial robots are by definition (ISO 8373, 2012) universal, programmable manipulators and therefore fulfill no predefined function. The central task in industrial robotics is building a production machine with the robot as basic component. This task includes carrying out engineering and integration activities. Examples are the design and integration of customized peripheral components such as end-effectors, fixtures, and feeders. Further tasks concerning the overall robot system are safety analysis, component integration, and overall performance optimization.

3 Problem analysis

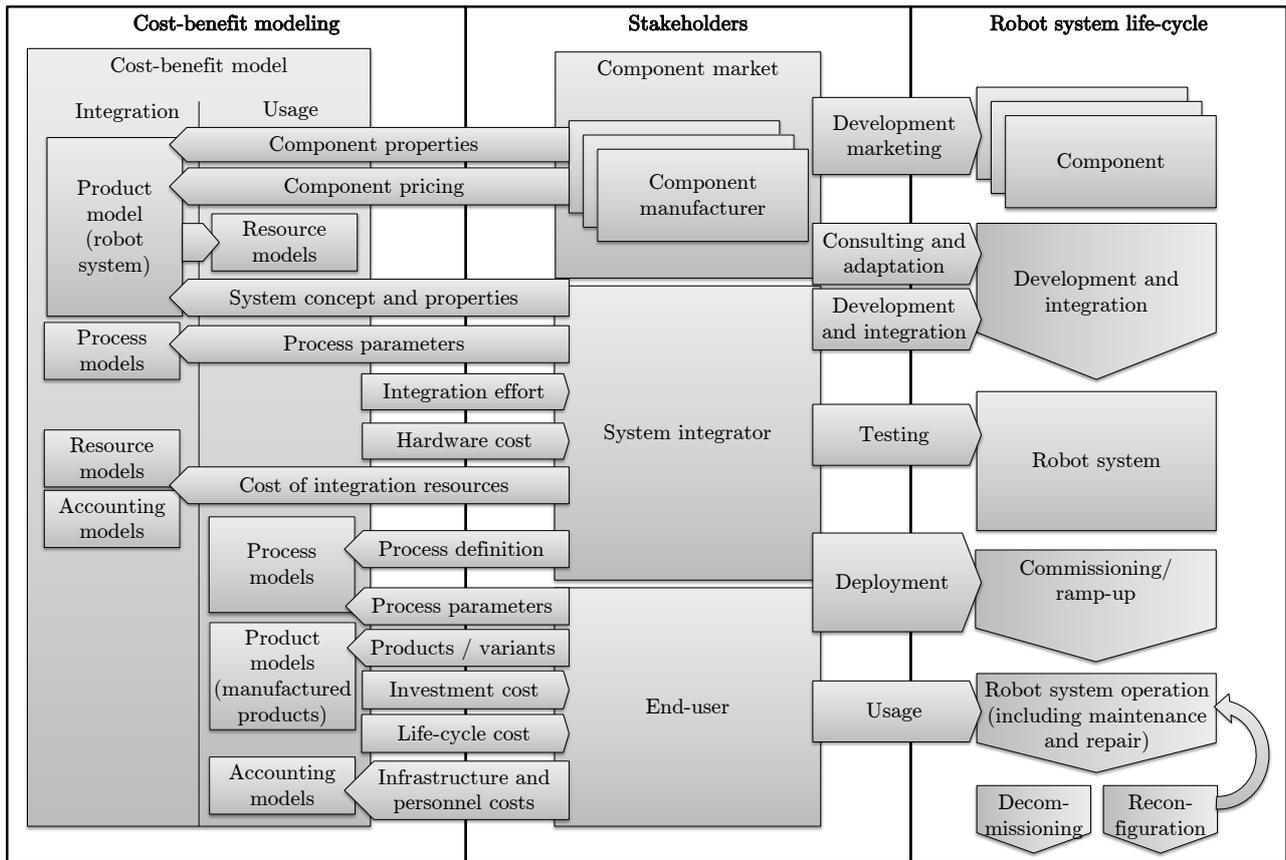


Figure 3.4 Stakeholders in integration and operation of industrial robot systems, their connection to cost modeling in robotics, and life-cycle phases.

As outlined in Section 1, the challenge for cost-benefit assessment of robot systems is that the required information is distributed among the different stakeholders, namely component suppliers, system integrators and end-users. Figure 3.4 shows the life-cycle of the engineering process for robot systems, the involvement of these stakeholders and their connection to different aspects of cost-benefit computation. The relation between the middle and the right column specifies the involvement of the different stakeholders in the life-cycle of the robot system. The component suppliers develop components that are the building blocks of the robot system. Apart from sales, they might offer consulting regarding the use of their products and minor adaptations of their products for specific applications. The system integrator is responsible for the development, integration and testing of the robot system up to the FAT. The deployment of the robot system into the factory of the end-user typically involves both, the end-user and the system integrator.

The regular usage is then managed by the end-user. Not shown in Figure 3.4 is that the system integrator will usually be involved in maintenance, repair and reconfiguration. Also not shown is the fact that the robot manufacturer is directly involved in the service of the robot.

The relation between middle and left column of Figure 3.4 highlights what information in the cost model can be supplied by which stakeholder and how those stakeholders can draw results from the cost-benefit model. The concept of the robot system and hence the bill of material of components for the integration are supplied by the system integrator. Information on component prices and performance are supplied by the component supplier. The properties of the product model during integration, i.e. the robot system, define the properties of the resource models during usage, as the performance in operation depends on the used components and their integration. The system integrator mostly interacts with the integration aspects of the cost model. The system integrator supplies parameters for the models of the integration activities, related resource models, and most of the accounting information during the integration phase. In turn, the system integrator is interested in information on the required efforts for the realization of the system in terms of personnel effort and hardware cost. The end-user supplies information for the process and product models in the usage phase as well as accounting information as it is the stakeholder most involved in the usage phase. In turn, the end-user receives information on the required investment for the robot system and an overview of costs and benefits over the system life-cycle.

Summing up, the involvement of the stakeholders and their ability to provide information and usage requirements for the cost model vary in the different life-cycle phases:

- During development and integration, the main work efforts are carried out by the system integrator that purchases components and integrates them into a robot system. The system integrator defines the system setup and requires information on the cost effects of design decisions in order to justify design choices to the end-user and maximize project profit. Component man-

ufacturers supply information on performance characteristics and pricing of components.

- During commissioning and ramp-up the responsibility for the robot system shifts from system integrator to end-user. Both stakeholders have significant work efforts in this transit and define the related cost items.
- During operation, the main stakeholder is the end-user. For repair, commissioning, and reconfiguration, external partners, in particular the system integrator and robot manufacturer, are involved. The end-user has the knowledge on how the system is used and what products are manufactured. In addition, the end-user has the required information about the internal cost structure in manufacturing.

The aim of the cost analysis from the point of view of the end-user is to obtain information whether the use of the system is cost-effective and how its operation can be optimized from an economical point of view. The system integrator wants to assess whether building the robot system for the end-user at a specific price is an economically sound project that promises a profit justifying the involved risk. Furthermore, the system integrator is interested in demonstrating to the end-user the benefits of his particular robot system concepts over the entire life-cycle. The model presented here reflects these different point of view and the specific involvement of the stakeholders in single life-cycle phases by structuring model elements and data accordingly.

3.3.3 Cost and flexibility of robot systems

Besides secondary objectives such as strategic advantages, improvements in quality or workplace ergonomics, the most important reason for automation are cost savings through reduction of manual labor per produced unit. Manual workplaces are typically characterized by a low initial investment as little or no machinery is required and the workplaces are not closely tailored to a specific application. As human workers can be employed very flexibly for changing tasks, this leads to relatively low fixed cost and high adaptability of the work environment. However,

the manual work also creates high variable costs per piece compared to automated processes.

Compared to manual processes, automating a process typically requires a significant investment. Furthermore, requirements for space and maintenance increase the fixed yearly cost. This initial investment is expected to pay-off through the reduction of personnel cost. Compared to the cost of manual labor the costs for energy and other operating supplies of most automation solutions are very low. Therefore, the variable costs per product unit of automated processes are typically much lower than the variable cost per piece of manual processes. Hence, automating a process typically means replacing variable cost¹ by fixed cost in order to achieve a cost advantage for a certain production volume (Figure 3.5).

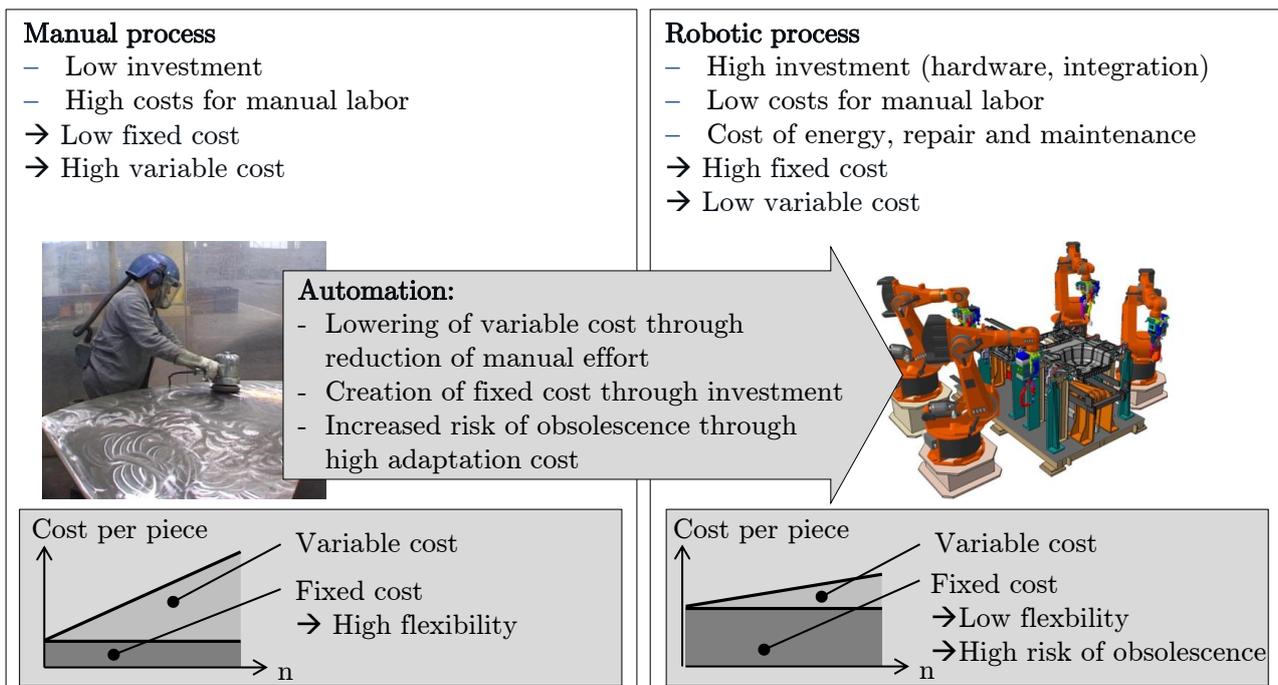


Figure 3.5 Comparison of cost aspects of manual and automated production.

However, automation systems are typically tailored for a specific range of products or often even for a single product. The adaptation to new products requires

¹Personnel cost refers to personnel cost per produced unit. Even through personnel cost on the company scale are often fixed due to job protection the personnel cost per produced unit is considered as variable cost as it is assumed that workers can be employed for other tasks. This assumption appears valid as retraining of human workers for standard production tasks causes relatively low effort when compared to automation.

notable effort resulting in a reduced flexibility when compared to manual work. This leads to a significant risk of obsolescence of automation systems resulting in an expectation of short payback intervals of typically two to three years in most industries. Hence, the aspect of risk resulting from the uncertainty of the future in terms of market demands and future products is very important. In the light of this fact, it is even more astonishing that an explicit quantification of risk is not carried out in many capital budgeting decisions regarding automation.

3.3.4 Interaction of system integrator and end-user

The realization process of the robot system includes the first two life-cycle phases Development and Integration and Commissioning and Ramp-up. The realization for industrial robot systems is complex and involves a wide range of development methods and software tools (Figure 3.6). This includes standard office tools such as text processing and spreadsheets during the entire project. Tools for design and simulation of the robot system such as as offline programming (OLP) and computer aided design tools (CAx, CAD) are used in development. Integrated development environments (IDE) are used for creating custom software functions. Vendor specific tools for configuration and online programming are used during robot system integration. Most tasks in these phases are carried out by the system integrator.

The analysis of the application requirements results in the requirement specification as an input for the system integrator. Often, a rough economical feasibility check is carried out in this phase to check whether cost-effective automation appears possible for the analyzed production task. In the following, a rough concept is generated by the system integrator to obtain the functional specification for the project. This often includes simulations and illustrations of the final machine. Depending on the complexity of the robot system, single functions are analyzed in a separate feasibility study. After the validation of all functions, the robot system is designed in detail resulting in the functional specifications for all components and modules. Depending on customer requirements, industrial sector, and size

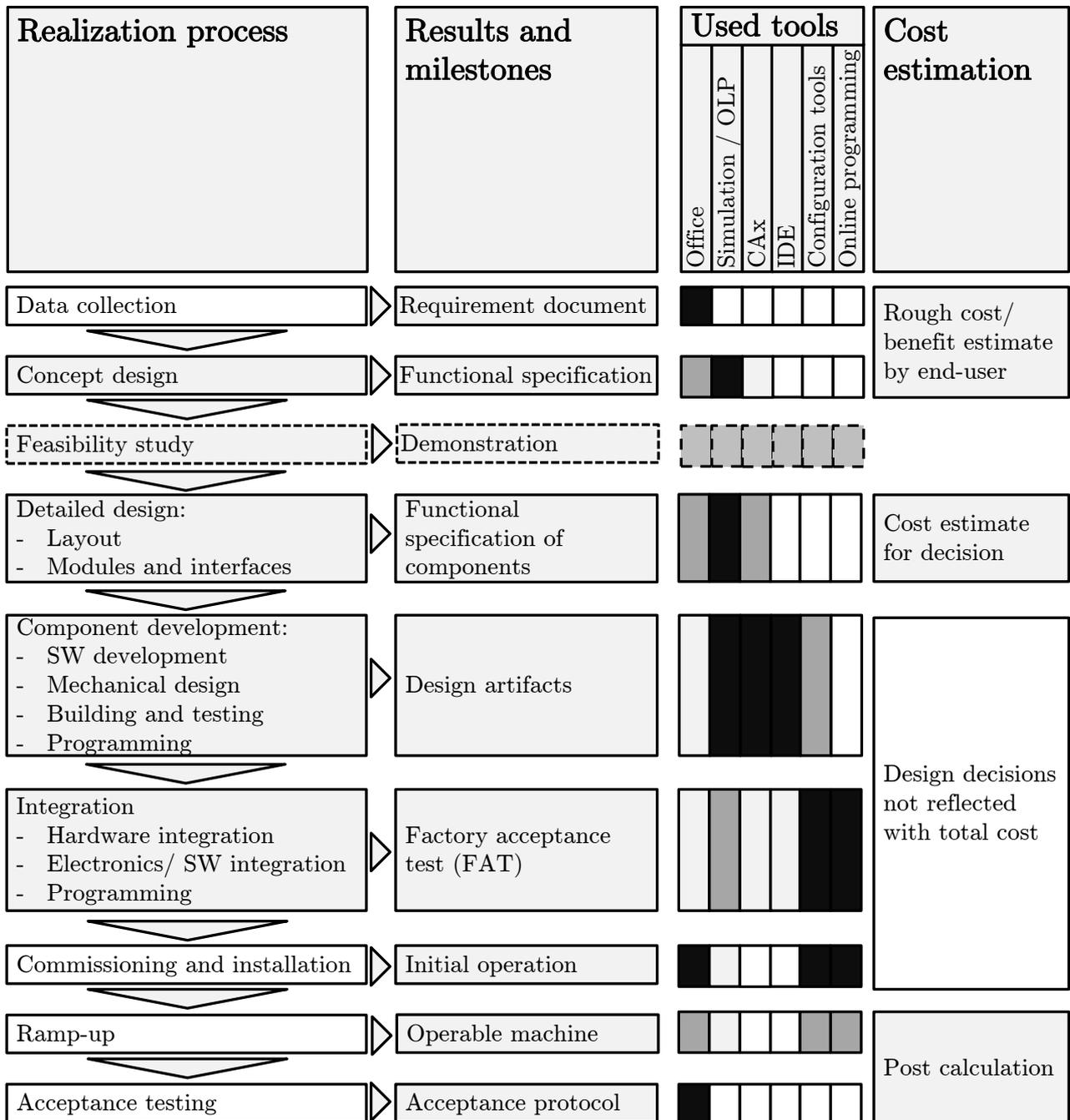


Figure 3.6 Key stages of the realization process, their results and the used engineering tools in these stages. Optional phases are drawn with dotted line.

of the robot system, the process up to this point is sometimes part of the realization project, i.e. after completion of a contract between system integrator and end-user, and sometimes part of project initiation before a realization contract has been signed. Sometimes the work is performed within a small pre-project preceding the actual realization project.

After the detailed concept, the single components of the robot system are developed or purchased by the system integrator. In this phase, the system integrator tries to realize the system according to the planned cost structure. Currently, cost effects on the entire robot system and resulting life-cycle cost implications are typically not assessed during the development of the single components of the robot system as long as the components stay inside their specifications. In the development phase, the system integrator uses common engineering and software development tools in order to create development artifacts. Examples are CAx tools, for example for mechanical and electrical design, offline programming (OLP) software for simulation and an integrated development environment (IDE) for software development. Subsequently, the single components are integrated into a robot system. Here, the configuration and programming of the single components gain more and more importance. After integration and acceptance by the end-user, the robot system is commissioned, including documentation of the system, and installed at the end-users facilities. After the initial operation, the production is ramped-up and acceptance testing is carried out. At this stage, the system integrator and the end-user might perform a post calculation to learn for further projects. However, in industrial practice, this post calculation is often not carried out.

3.3.5 Alignment of system integrator and end-user interests in the realization of robot systems

System integrator and end-user have different aims in the assessment of cost and benefit. The system integrator estimates the realization efforts for the robot system and determines the project price for which he can offer the system while

being able to generate a profit. As he wants to convince the end-user to accept his offer, the system integrator is motivated to offer the robot system at low cost. Therefore, the system integrator will usually try to:

- build the system from components with which he has previous experience,
- exclude some integration activities and components from the offer in the hope to sell these as additional services to the end-user later on,
- propose a solution that can be supplied at low cost rather than produce a solution with optimal performance,
- overstate soft performance characteristics of the proposed solution,
- contractually withhold development artifacts from the end-user in order to bind the end-user to long-term service agreements, and
- transfer realization risks to the end-user, for example through requiring certain work piece properties or contractually excluding certain risks.

On the other hand, the end-user has to evaluate competing, heterogeneous proposals from different system integrators with respect to a wide range of performance indicators relating to technical performance, cost effectiveness, and project risk. These indicators have to be evaluated in the light of uncertain scenarios regarding the future production for finding the best solution over the whole life-cycle. In order to perform this assessment, the end-user requires exactly the information that the system integrator is motivated to withhold or at least to disguise.

The engineering process possesses relatively smooth stage transitions in which the interests of partners exchanging information are well-aligned. The cost assessment process contains a break during the contract negotiations that prevents effective cost-benefit assessment over the entire life-cycle of the robot system. This break occurs because the interest of system integrator and end-user are not aligned and therefore these stakeholders are not motivated to share information. The resulting lack of information makes it impossible for system integrator and end-user to negotiate over the life-cycle cost as the most important key performance indicator (KPI) for the end-user. As a result, the negotiation is based on KPIs

that are only indirectly related to life-cycle cost. Often a strong focus is placed on the purchase price of the robot system. This pushes the system integrator towards suggesting robot systems and project conditions that are not optimally aligned with the interests of the end-user. This misalignment of interests is summarized in Table 3.1.

The root cause for above mentioned misalignment of interests is the lack of decision support that prevents a more elaborate grounding of investment decisions. Solving this problem would allow end-user and system integrator to negotiate on a more advanced level as the end-user would be able to assess the true life-cycle cost of the robot system. This in turn might justify a higher upfront investment as it would be possible to prove that the higher investment will pay off in the long run. Establishing this type of cost-benefit transparency could push the robotics market towards more ambitious solutions and hence increase the pace of innovation in industrial robotics.

3.4 Problem statement and breakdown

In order to formulate the problem, the usage scenarios of cost-benefit assessment are outlined. Based on these scenarios the problem is formulated and broken down into single sub-problems whose solutions form the main contributions of this thesis.

3.4.1 Usage scenarios of cost-benefit assessment and problem formulation

The analysis of the development life-cycle of the robot system reveals that the insufficient flow of information between system integrator and end-user regarding the life-cycle cost of the robot system poses a problem for setting-up well-targeted automation projects in which the interests of system integrator and end-user are well-aligned. This misalignment of interests occurs in the conceptual design phase,

Table 3.1 Conflict of interests of system integrator and end-user.

Aspect	System Integrator	End-user
Purchase cost	Aims at achieving a high profit margin, i.e. minimum realization cost and high purchase price. Motivated to propose solutions that can be realized at low cost/ risk.	Aims at lowest system price while fulfilling the requirements. Has to weight differences in purchase price and performance of different suppliers.
Operating cost	Wants to generate revenues from servicing of the robot system. Interested in end-user lock-in and regular service requirements. Not motivated to reduce the operating cost beyond the requirements of the end-user.	Wants to reduce operating cost as far as possible. Wants to avoid service and lock-in as far as possible. Might view these aspect as secondary during procurement phase.
Life-cycle cost	Only interested in the life-cycle cost of the system as sales argument.	Truly relevant cost figure for the end-user. However, due to the complexity of assessment, the end-user often evaluates proposals primarily based on the sales price.
Development risk	Aims to reduce responsibility for the realization risk, further encouraging unambitious solutions. Tries to hedge the development risk on the end-user.	Wants to outsource the development risk to the system integrator as far as possible. Requires flexibility for changes in requirements at later stages of the project.
Additional work	Wants to exclude additional work and make additional claims in case of changes.	Prefers a price guarantee where the price for additional work is included in the system price.
System performance	Interested in meeting the end-users requirements. Sales people sometimes encouraged to promise unrealistic performance to get contracted.	Aims at obtaining a system with best relation of cost and benefit over entire life-cycle. Struggles to find relevant performance metrics and specify meaningful requirements.
Service	Wants to create service lock-in.	Interested in flexibility with respect to service provider.

the detailed design phase, and the post calculation of the project and influences the cost-effectiveness over the entire life-cycle.

A decision support tool is required to overcome this misalignment during the conceptual design phase. This decision support tool has to guide principal design decisions based on their effects on costs. Currently, in the conceptual design phase, formal cost calculation is not used at all or very simple calculations are made. These calculations usually compare competing solutions with respect to their realization price and some other KPI. However, as already explained, decisions taken in the conceptual design phase are defining the life-cycle cost of the final system. These aspects become more important as usage scenarios of robot systems become more dynamic in terms of manufactured volume, variants, and new products.

The detailed design phase is typically closely entwined with the contract negotiation between system integrator and end-user. The system integrator has to continuously refine its design in order to quote the project to the end-user. Typically, the end-user and the system integrator make cost calculations to evaluate the benefit generated by the project. Furthermore, system integrator and end-user communicate through the performance characteristics of the robot system. A cost-benefit calculation tool should help in this phase to integrate relevant information of the cost calculations of both sides in order to allow a life-cycle assessment of the proposed automation project. This is in the interest of both parties. The end-user would be able to resolve the trade-off between conflicting design parameters, for example investment cost and system performance, based on design information rather than establishing arbitrary thresholds constituting a particular resolution for this trade-off. The system integrator would be capable of demonstrating the benefits of its technology over the entire life-cycle instead of suggesting a system that conforms as well as possible to the arbitrarily chosen resolution of performance trade-offs of the end-user.

During the development phase, a cost-benefit tool can be used as a decision support system to reflect design decisions. As a tool for post calculation, the cost-benefit tool can be used to determine the actually achieved costing performance

of the robot system over the life-cycle. This evaluation can be used to award a bonus to the system integrator to motivate the system integrator to align its interest with the end-user's interest.

As it becomes clear from the above mentioned life-cycle and common costing methods outlined in Chapter 2, the aspect of risk related to single decisions or the automation project is currently not widely addressed. Sometimes a few scenarios regarding commissioning and use of the robot system might be calculated in order to obtain a rough understanding on how the project reacts to changing assumptions. However, this is not sufficient to shape the robot system in the design process and to identify functionality that significantly contributes to or prevents certain risks.

Problem definition: The problem addressed in this thesis is to enable the pervasive use of costing calculations in the entire design process of a robot system for supporting design and investment decisions while making the associated risk of different options transparent and integrating the knowledge of the involved stakeholders seamlessly.

3.4.2 Breakdown of problem

The problem of evaluating the cost-effectiveness of advanced robotic solutions is broken down into different aspects. Firstly, a cost-benefit model reflecting the particular structure and life-cycle of robot systems is required. This model needs to be able to simulate the cost aspects of the robot system over the entire life-cycle. Furthermore, this model needs to be able to incorporate the inherent uncertainty in all cost data and allow assessing the influence of these uncertainties on the cost and benefits of the robot system. Secondly, the cost-benefit model requires extensive amounts of input data that is distributed among different stakeholders. Usually not all required information is available. This missing information needs to be estimated, for example based on past projects with similarities. For this purpose, means for knowledge management and for reasoning to complete missing

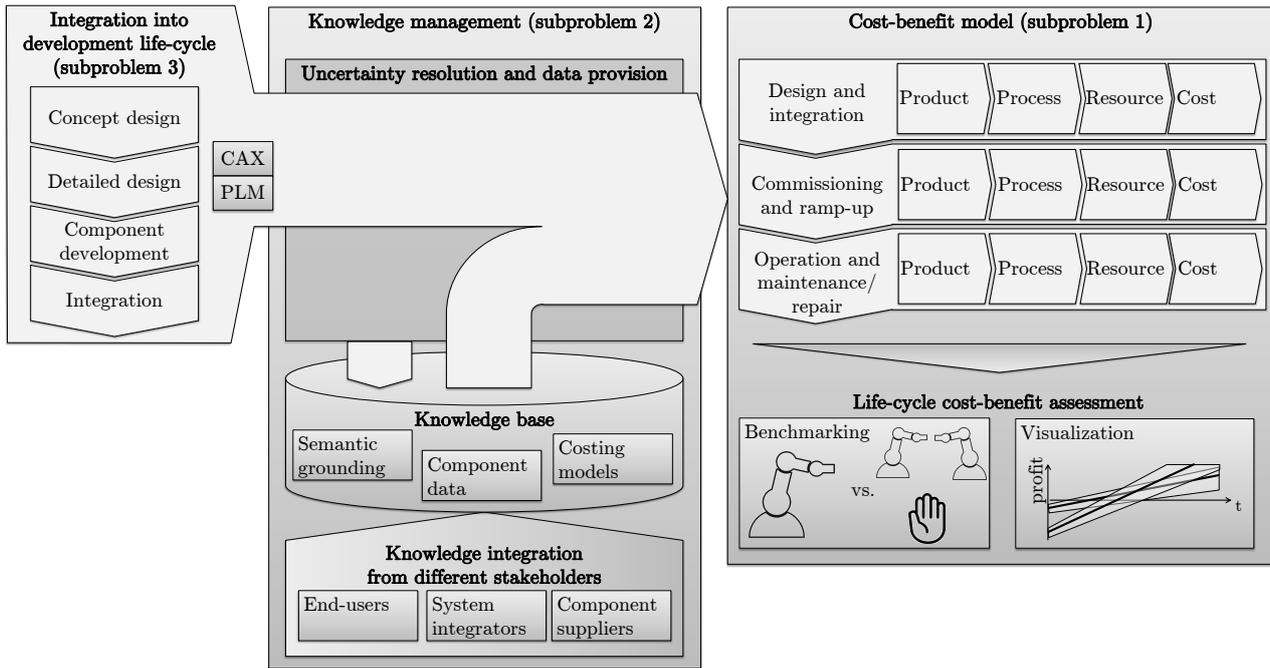


Figure 3.7 Relation of the addressed sub-problems.

data based on comparable automation projects or user estimation are required. Thirdly, the cost-benefit approach needs to be integrated into the development life-cycle to address the above mentioned requirements. These three sub-problems and their relations are shown in Figure 3.7 are explained in the following sections.

Cost-benefit model

The problem of finding a suitable cost-benefit model includes the problem of modeling of all decision-relevant costs and benefits of owning the robot system over the entire life-cycle. The problem on the cost side revolves around finding suitable models for describing the cost-causing activities and computing the resulting cash flows. The problem of quantifying the benefits that result from the use of the system in production is more fundamental and includes the definition how benefits are quantified in principle. Distal-monetary benefits, i.e. benefits which are hard to quantify in terms of cash-flows, such as flexibility with respect to changing products, improved quality or strategic considerations are not explicitly addressed

in this thesis. However, some of these benefits can implicitly be linked to the aspect of uncertainty represented by the model.

Cash flows are measured in currency units and can be directly compared with each other after being discounted to the same time basis. Although measured in currency units, the terms cost and cash flow are not limited to actual transfer of monetary resources. For example, the consumption of resources, e.g. work, results in cost although the resulting transfer of monetary resources might take place at a later point in time, i.e. at the end of the month in case of salaries. This assumption appears useful as the cost-benefit model should exclude accounting details, such as the accounting practice regarding salaries, as far as possible. Therefore a cash flow is defined as the inflow/creation or outflow/consumption of valuable assets from or to a stakeholder directly related to activities performed for modifying or operating the robot system. In the following, positive cash flows refer to an outflow/consumption of valuable assets, e.g. cost in the form of outflow of financial resources or resource consumption. Negative cash flows refer to benefits and returns, i.e. to inflows of valuable assets.

The cost model takes a usage scenario of the robot system as input and yields the cash flows that occur due to the activities for producing and using the robot system. However, these cash flows themselves do not allow an assessment whether or not the investment into the robot system is a good decision or the evaluation of certain features with respect to their user over the system life-cycle. In order to do this a metric for the benefits of using the robot system is required.

The addressed problem in cost-benefit modeling is finding a model description that connects the information on what is produced (product) with the used production technologies (process), the resources that are used in the production process and basic cost data of the producing company to determine the production cost. This objective has to be achieved in all relevant phases of the robot system's life-cycle. During the development, integration, commissioning and ramp-up phase, the product is the robot system itself, the technologies relate to integration tasks and the resources are personnel and production means for producing the robot system. During the usage phase, the robot system is the resource used for pro-

duction of actual customer products and the technologies are the manufacturing processes carried out in the robot cell.

The model needs to reflect the interdependence of the single life-cycle phases. In particular, it needs to describe the dependence of performance characteristics of the robot system in the usage phase on the set-up of the robot system defined in the integration phase. Model descriptions need to be found for products, processes, resources. Furthermore, overheads on these costs need to be modeled for the individual companies having a stake in building and using the robot system. All these models need to be able to represent uncertainties in their properties. Finally, means are required for executing the cost-benefit model and representing the resulting information to decision makers in an intuitive fashion.

Knowledge management

The use of existing cost-benefit models often requires significant effort due to the large amount of input data required for computing the model. For the stakeholders participating in the development, commissioning, and operation of robot systems required data is often not available as for example:

- the costs do not occur in the own company, such as development efforts from the view of the end-user or installation efforts or personnel cost from the view of the system integrator,
- efforts and cost are not known as robot systems are special machinery and directly transferable examples of similar devices do not exist or
- because the relevant cost data is not tracked by the company, e.g. set-up times for machine tools in SMEs by experienced workers.

The core task for knowledge management is to capture information from different stakeholders, namely component suppliers, system integrators, and end-users as well as general market data and to integrate it into a consolidated set of knowledge to be used by the cost model. If input data is missing, for example in case a machine is developed without a specific customer supplying its personnel cost for

operation, estimates and an assessment of the quality of the estimate, i.e. the level of uncertainty associated with it, are required.

To that end, methods for knowledge management are required that solve three main tasks:

1. Data input: Capture and categorize data from existing automation projects, e.g. by surveying existing projects.
2. Knowledge representation: Mechanisms for data representation that allow categorization and aggregation of information from past automation projects.
3. Knowledge usage: Means that allow the user to parameterize the cost-benefit model for a specific application to be evaluated by using existing data, while the system questions the data representation to complete missing information including an estimate of the uncertainty related to this information.

Critical attention has to be paid to the fact that the cost-benefit assessment will be carried out in early phases of the robot system's development with limited knowledge about the single components.

Integration of costing into the development life-cycle of the robot system

The cost-benefit assessment system developed here can be used in different phases of the development life-cycle to support decisions. As the development proceeds, more information on the system setup becomes available and should be reflected in the cost model to improve its accuracy. Therefore, a tight integration of the costing tools into the development life-cycle and seamless information transfer between development tools, mostly PLM tools and CAx tools, is required. This will allow to add additional information from CAx tools to the cost-benefit model and feed-back information generated during the cost-benefit assessment process to the main models used in CAx design. The requirements on the cost-benefit assessment tool and its single components are summarized in the Appendix 1.

Apart from the technical challenges of integrating the cost-benefit model in the development life-cycle, also the motivation of potential users to supply data and utilize the cost-benefit assessment tool have to be discussed. The business models of system integrators and component suppliers depend partially on customer lock-in. This leads to the problem that the cost-benefit assessment tool needs to be designed in a way such that these stakeholders are willing to supply data to the cost-benefit assessment system, because they see a benefit for their business in it.

4 Conception of the cost-benefit model

In this chapter, the overall concept of the cost-benefit model for robot systems is established. At first, the methodic approach is defined. Based on this, the chapter explicates the structure of the cost-benefit model for the entire life-cycle of robot systems. It defines the applied methods and mathematics using the state of the art and building on the analysis of the problem. The aspect of handling uncertainty plays a central role in this conception. Furthermore, the approach for handling knowledge from different stakeholders, namely component suppliers, system integrator, and end-user is defined.

4.1 Overall costing system concept

As outlined in the state of the art (see Chapter 2), different options for treating cost-relevant aspects of robot systems exist. This section discusses these different alternatives and evaluates them in the light of the analysis presented in Chapter 3. Based on this, the preferred approach to cost-benefit assessment is chosen.

4.1.1 Approaches to cost computation of robot systems

Five different approaches to cost-benefit assessment are considered. The process-oriented approach estimates cost based on models for the different work tasks during implementation and usage of robot systems. The expert-based approach

computes cost estimates through a combination of expert estimates. Analogy-based approaches attempt to estimate costs from comparable solutions that have already been realized. Approaches using artificial intelligence analyze cost data from past projects to establish an expert system without the need for explicit model building for the development and usage process. The hardware overhead approach estimates overall cost through fixed overheads on the investment of system components. These approaches are investigated in more detail in the following.

Process-oriented approach The process-oriented approach is centered around a model of the work processes executed in the different phases of the robot system's life-cycle. Cost models for all processes allow an estimation of the cost generated by the work process. Using these cost models, cost aspects along the life-cycle of the robot system can be simulated. Hence, the process-oriented approach tries to model all details of cost creation. The downside of this approach is that a high number of specific models for cost creation are required. On the other hand, this approach allows to reflect a high level of knowledge in the cost-benefit model that is available instantly in case an estimate is required.

Expert estimation Experts are often able to provide estimates of cost for single integration phases or subcomponents of the robot system. These expert estimates can either be provided in terms of a scale, for example robot price depending on payload range or programming effort depending on number of weld seams, or as absolute values for a specific task. In this context, two possible embodiments of the expert system exist. In an expert network, the estimation task is broken down into domain specific estimation tasks that are then supplied to an expert network to obtain estimates for these aspects. The estimates are then combined to obtain an overall estimate. Another possible embodiment is to store typical expert estimates in a knowledge base system for retrieval in case they are needed. Expert estimation allows to obtain a large amount of information without the need for explicit modeling. However, general questions regarding biased estimates by experts persist. Fur-

thermore, the approach of an expert network prohibits to obtain immediate cost estimates and creates significant effort for each estimate.

Analogy-based approaches Analogy-based approaches rely on case-based reasoning and try to estimate cost from past projects that have analogies to the analyzed application. Such analogies are for example the properties and quantities of the manufactured product, the applied manufacturing technologies as well as the type and quantity of components of the robot system. Analogy-based approaches allow to use information from past projects effectively. However, the comparison often requires experts familiar with the past projects under scrutiny. Additional questions prevail with respect to confidentiality of the data of past projects and the fact that cost calculations are not available in a standardized scheme. A corresponding, anonymized database is required to realize this approach. Furthermore, market changes might invalidate the information from past projects.

Data analytics and deep learning A lot of current research in robotics focuses on utilizing approaches from machine learning and data analytics in robotics. Using approaches such as deep learning could be used to extract structures from cost data and project information of past automation projects. In particular, the ability to establish high-level abstractions from data could prove useful for structuring cost information from past projects. Again, as with analogy-based approaches, a problem might be the availability of structured cost information with semantic annotation. Furthermore, the amount of information that can be gathered in significant detail is most likely not sufficient for machine learning approaches.

Hardware overhead model A high percentage of the cost of a typical robot system results from hardware cost, for example for the robot itself, peripherals, and controllers. Hardware produces additional effort during integration, set-up, initial operation, and maintenance. The hardware overhead model approach estimates typical overhead costs that these components cause as a percentage of the sales price. The underlying assumptions are that components cause these costs and that these costs are related to the component

price. In fact, the effort for some activities is correlated with the component price. For example, the maintenance of higher priced components is typically also more expensive or high performance products require more careful integration and therefore cause higher integration cost. However, this approach can only make very general estimates as it cannot trace actual causalities leading to cost generation. It is expected that this approach is only suitable for modeling a few key cost items, since most activities required for integration do not depend on the purchase price of the involved equipment but rather on the number and complexity of integration tasks and on how the component is used. The required efforts for these tasks vary widely for different types of components and also depend on the exact usage and functions of the components in the robot system.

4.1.2 Assessment and selection of conceptual approach

Figure 4.1 compares the considered approaches with respect to key findings from the analysis in Chapter 3.4. The considered criteria for the comparison are:

- Accuracy and detail: expected accuracy of estimation and level of detail of the assessment of robot system and application
- Uncertainties: possibilities of handling uncertainty in the model
- Data: supply of data, availability of required information and how it can be obtained
- Speed: speed and effort for obtaining estimates for a specific automation project
- Stakeholders: ability to reflect views of different stakeholders and integrate their data
- Integration: integration with the design process and the tool used in the design process

- Risk: risk associated with the approach for obtaining useful results for the cost estimation for robot systems (risk regarding applicability and feasibility of the chosen approach)

The expert assessment of the different approaches with respect to the introduced criteria shown in Figure 4.1 is justified by the specific properties of the approach stated with the respective evaluation.

	Process-oriented	Expert estimation	Analogy-based	Data analytics	Hardware overhead
Accuracy and detail	+ High level of detail, explicit modeling	- Limited detail, biased estimates	○ Limited detail, lack of detail in analogy	++ High level of detail if enough data available	-- Very general approach with lack of detail
Uncertainties	++ Explicit modeling and propagation	+ Reflection in expert estimates	- No data source for uncertainty	++ Statistical methods explicitly treat uncertainty	○ Possible, but only very general
Data	○ Large amount, fair availability	+ Supply by experts, limited expert availability	+ Based on past projects, if existing	-- Large data amounts usually not available	++ Very simple data collection of hardware cost
Speed	++ Computation of models yields immediate results	-- Request of expert estimates takes time	- Manual intervention requires time	+ Fast evaluation after data analysis	++ Very fast computation of results
Stakeholders	++ Model allows inclusion of different views	○ Experts can take different views	- Views highly specific for particular project	- Views highly specific for particular project	+ Possible to specify where overhead occurs
Integration	++ Processing of design files	-- Creation of expert digests from design info	○ Analogy criteria need to be extracted	+ Extraction of relevant metrics from data	++ Simple computation from design data
Risk	○ Relevant costing models need to be found	+ Proven method, risk of biased estimates	+ Proven method, but limited detail	-- Applicability unclear	- High risk of lack of accuracy

Figure 4.1 Comparison of identified approaches for cost modeling in robotics.

As can be observed from Figure 4.1, the process-oriented approach is the most promising with respect to the specific requirements for the costing system. Hence, the process-oriented approach is selected as the predominant approach for the development of the cost-benefit model. In particular, the ability to model cost creation mechanisms in great detail and the ability for implicit consideration of uncertainties in these models are beneficial here. Furthermore, this type of model requires only the execution of algebraic functions for simulating a specific system set-up or usage scenario. Therefore, the model allows supporting concurrent decisions during the system design process. Finally, the process-oriented approach is easy to integrate into the overall design process of the robot system, as it at-

taches activities to design artifacts contained in the development data. The key question for applying the process-oriented approach is the detailed structure of the cost-benefit model and finding activity models that allow describing single costing aspects of the system development and usage. This combination of the process-oriented approach to costing with the PPR-approach commonly found in plant engineering (see Section 2.4.1) is one of the key contributions of this thesis and detailed in the following section. For modeling single cost aspects other approaches such as expert estimation, analogy-based approaches and hardware overheads might be used, even though the process-oriented approach is chosen here as the central approach for the cost-benefit model.

4.2 Uncertainty-aware cost-benefit model

This section describes in detail the structure of the developed cost-benefit model and the theory behind the different components. As outlined above, the model uses a process-oriented costing approach and combines this approach with the PPR-approach from plant engineering.

4.2.1 Types of cost

Work activities imply effort and cost through the continuous usage of resources or consumption of material and energy. Activity-driven costs are defined as costs that occur through performing work activities, which have a defined start-time and end-time. The start of the activity can be associated with an event triggering the activity. Examples of activity-driven cost are the cost associated with personnel effort for the integration of a component into the robot system or cost for the usage of energy due to the execution of a manufacturing task.

Other costs are caused directly through discrete events. These so called event-driven costs are directly and immediately caused by specific events without the usage of resources of the stakeholder, i.e. without a continuous activity. When

examined closely, event-driven costs usually capture activities that are carried out by another stakeholder or are beyond the scope of the model. An example of event-driven cost is the cost for payment of system equipment. Activity-driven and event-driven cost are treated equivalent in the cost model, i.e. a discrete cost-item is modeled through an activity with zero time duration.

Some cost items are not related to a specific activity, but result from the usage of basic resources, such as capital or space, over time. These costs are referred to as time-driven costs. Time-driven costs are cost whose amount depends on the time passing while being in the possession of the robot system. Examples of time-driven costs are the cost of space occupied by the robot system in production or the cost of capital for financing the robot system.

4.2.2 Structure of the cost-benefit model

One key aspect of the process-oriented cost model is its model structure. The model structure has to reflect relations of work processes, events that cause the expenses and finally their cost. Furthermore, it needs to support the re-use of knowledge. The model for activity-driven and event-driven cost is structured with respect to the work processes resulting from the production requirements of the product. The time-driven cost aspects are treated in a separate structure outlined later-on in Section 4.2.2.

It is proposed to build the cost-benefit model structure for activity-driven and event-driven costs based on the PPR-approach (see Section 2.4.1). It is further proposed to extend this approach to model the generation of expenses in the different life-cycle phases according to life-cycle costing (LCC) methods (Section 2.1.2). Finally, it is proposed to model the creation of expenses on product, process, and resource level by using techniques from activity-based costing (ABC, Section 2.1.1). The resulting cost-benefit model combines PPR as typical approach for structuring plant information, with an assessment of the entire robot system life-cycle, and an implementation of the process-oriented costing approach through so called activity models borrowing from ABC.

This cost-benefit model (see Figure 4.2) takes the product-centric view, i.e. the product is taken as entry point for the analysis of cost. The product model specifies the manufacturing needs in terms of features to be created and the manufacturing processes to be carried out on the product. The model of the process, including models of the involved technologies, refines this information with details about the required involvement of resources. The resource models translate the information about resource involvement into efforts related to the consumption of primary resources. In order to relate this to cost caused by the production activity, it is proposed to introduce a fourth level to the PPR-approach. This fourth level will be dedicated to modeling of cost creation. It will be called the cost-pool. Here, the efforts on the resource level are transformed into actual cost. This extended PPR-approach is called PPRC-approach in the sequel.

The models in different life-cycle phases are interconnected. The model of the robot system is the product model in the realization phases while it becomes the resource model in the usage phase. Therefore, the product model during the realization phases defines the performance criteria of the resource model during nominal production. This is the case because the physical properties of the robot system determine the throughput, robustness, and quality that are later achievable in production. A role-based approach is used to capture this interdependence (see Section 4.2.4).

For further clarity regarding the combination of the PPRC-approach and activity-based costing, the models, their inputs and their outputs shown in Figure 4.2 and Figure 4.3 are outlined for the GMAW example. The manufacturing requirements are defined by the product, which requires the creation of a specific feature, e.g. producing a weld seam with 3 meters length with geometrical properties specified by the desired end-state of the product. This feature information on the product side is the input for process related activities. The production volume is translated into an output on process level using a model of the manufacturing technology. Furthermore, constraints arising from the use of specific resources, e.g. maximum speed of robot or maximum power rating of welding source, are taken into account in this process. Staying with the welding example, the volume of weld seams to

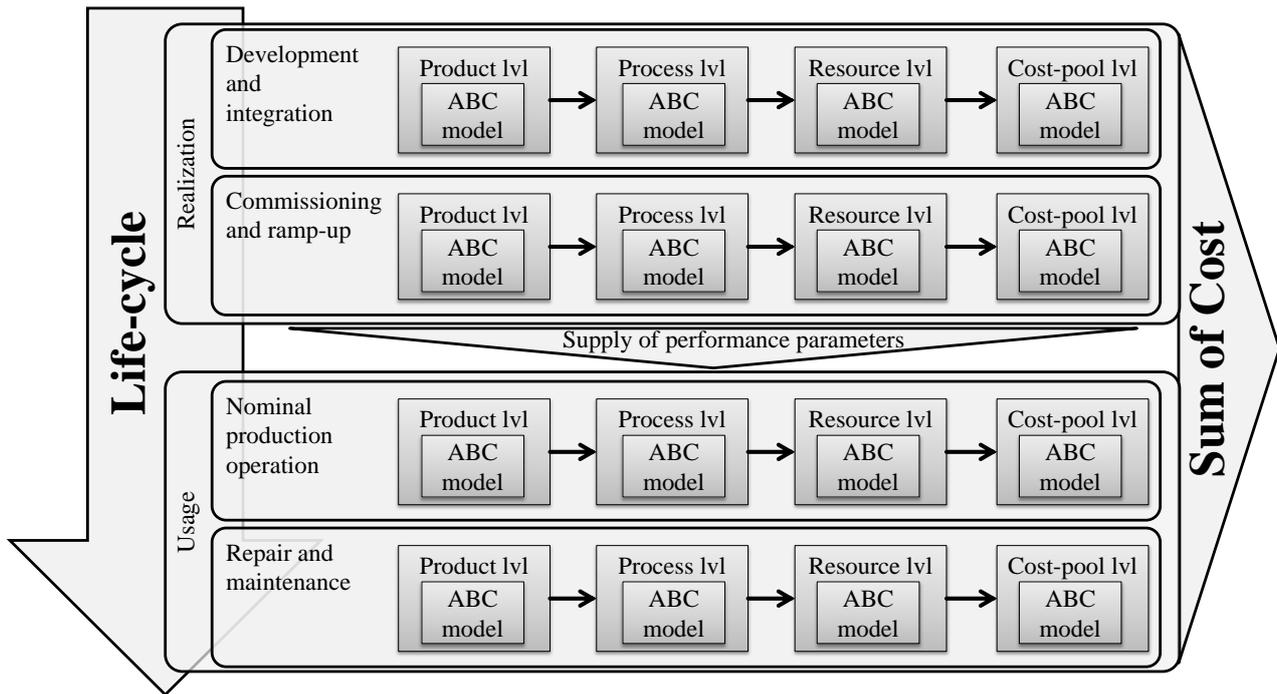


Figure 4.2 Schematic structure of the life-cycle model.

be manufactured is transformed into the required time for welding by using a model of the welding process. This output on the process level is the input on the resource level. For the welding example, the resource level translates the time required for welding into the overall time for production of the weld seams, the required energy, and the amount of manual labor. The outputs of the resource model are inputs for the cost-pool level. Here, the production costs are determined from accounting data. The cost-pool differentiates between product specific costs, for example commodities and components, which are treated as event-driven cost and cost that are related to activities of the manufacturing resources, for instance cost of energy or work effort.

In the following, the structure of the cost-benefit model for activity-driven and event-driven costs in the realization phase (Development and Integration, Commissioning and Ramp-up) and the usage phase (Nominal production operation, Repair and Maintenance) is explained in detail. In each of these life-cycle phases, the model is set up according to the general schematic for a single life-cycle phase shown in Figure 4.3. The production schedule in this figure either refers to the

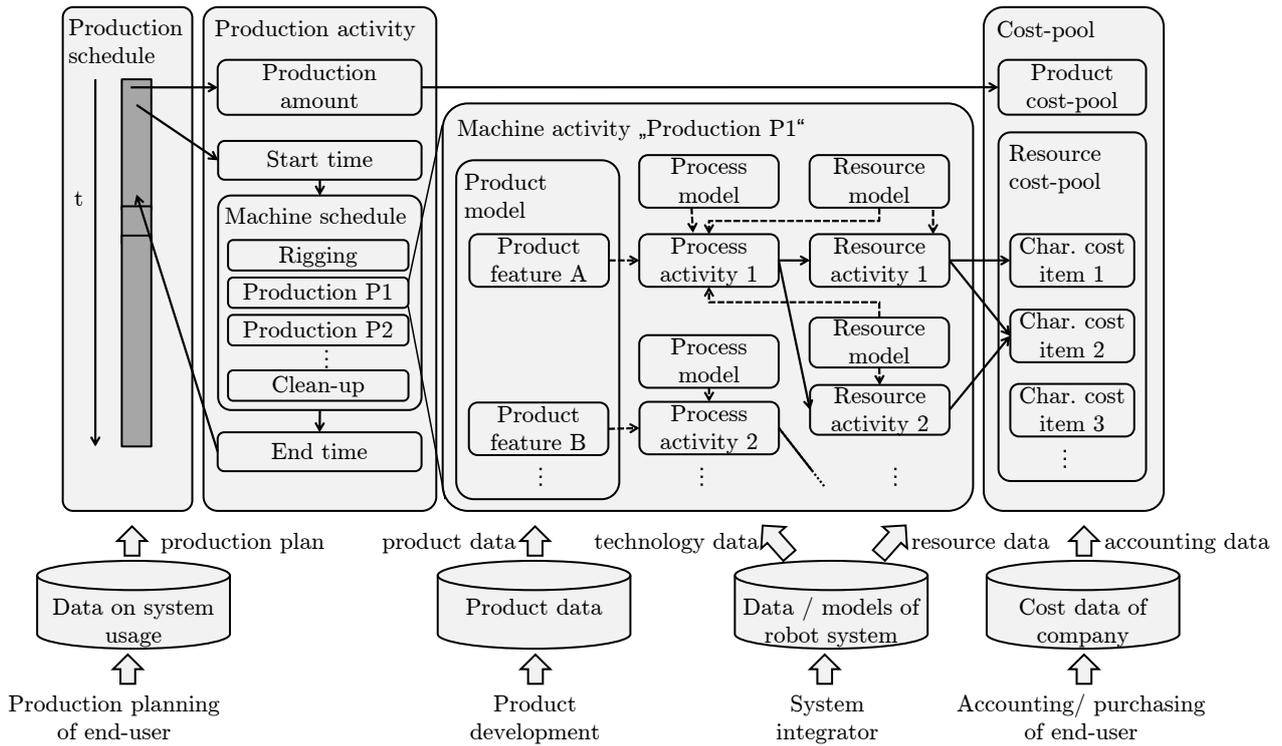


Figure 4.3 Structure of the used cost-benefit model and data sources for production-centric costs.

products produced on the robot system (during nominal production) or to the activities required for realizing the robot system (during development, integration, commissioning, ramp-up, reconfiguration, repair and maintenance).

Life-cycle phases centered around the robot system as product

During the life-cycle phases development and integration, commissioning and ramp-up, repair, maintenance, and reconfiguration, required activities are mostly defined by the components of the robot system. Costs are caused by component purchases and work efforts for integration, engineering and problem fixing. These costs are related to building the robot system or for keeping it operable and are therefore called machine-centric costs. Table 4.1 gives examples for machine-centric cost items that are relevant for robot systems.

A high percentage of machine-centric costs occur due to work efforts resulting

Table 4.1 Machine-centric cost for robot systems.

Category	Examples	Life-cycle phase	Underlying driver
Infrastructure adaptation costs before machine installation	Adaptation of building, relocation of existing machinery, adaptation of material flow, adaptation of company organization	Commissioning and ramp-up	event-driven, activity-driven
Equipment cost for purchasing of components from third party suppliers	Robot, sensors, fencing, tools and grippers, controller, computers, safety equipment, custom machined parts, etc.	Development and integration (integrator), order placement and ramp-up / commissioning (end-user) ¹	event-driven
Engineering cost for planning, implementation and problem shooting	Planning, management and design of robot system.	development and integration (integrator), order placement and ramp-up/ commissioning (end-user)	activity-driven ² , event-driven
Integration cost for making single components to work together as a system	Labor effort for set-up, interfacing, programming, problem solving	development and integration, commissioning and ramp-up	activity-driven ² , event-driven

¹ For the system integrator view, the costs occur when purchasing the robot system, for the end-user when paying invoices of the system integrator.

² Items might appear as event-driven or activity-driven to the end-user depending on whether the efforts are included in a fixed price contract or not.

from activities for designing and building the robot system (Dietz et al., 2013b). These activities are related to the components of the robot system. For example, an industrial robot requires mechanical and electrical installation, programming, as well as further activities before it can be used for production. Hence, for machine-centric activities, the bill of material of the robot system constitutes the product model. The activities required for integration of these components are described by process models. In this phase, the used resources are mostly personnel resources required for carrying out the integration tasks.

The knowledge on required integration and set-up activities for components is

stored in a knowledge base and can be retrieved through queries (see Section 4.3 for details). The activities are either related to the component itself (for example the robot of a specific type and manufacturer), its class (for instance an industrial robot) or its role in the robot system (for example a handling robot). The different activities might have pre-condition activities that need to be executed beforehand. For example, mechanical robot installation is required before online optimization of the robot program.

Furthermore, activities are assigned a priority that defines how crucial they are for achieving operating ability of the entire system. For example, the installation of the robot has a higher priority than documentation of its configuration¹. Through the evaluation of preconditions and priority, a schedule of integration activities can be synthesized from the bill of material of the robot system (see also Section 4.2.6 for details on this scheduling problem). In this context, the bill of material is central to the product model for machine-centric life-cycle phases. The bill of material can be obtained by export from the engineering tool used for plant engineering. The underlying models for the integration activities are process models in the terminology of the cost model. Examples for process models are provided in Chapter 5.

Some costs in Table 4.1 might appear as event-driven or activity-driven to the end-user while they are activity-driven for the system integrator. Again, this makes apparent the different view on cost of end-user and system integrator. Development and integration activities like engineering and integration cause pure activity-driven cost for the system integrator. For the end-user, these costs might appear as both, activity-driven or event-driven costs. In case the activities are included in a fixed price offer, they appear as event-driven costs to the end-user. The occurrence of the events triggering these costs depends on the payment modalities in the contract between system integrator and end-user. In case these are additional services that are offered by the system integrator in the form of a service agreement, these costs occur as activity-driven costs to the end-user.

¹Documentation here excludes the risk assessment of the robot system which should be carried out prior to installation.

Modeling of production activities

During nominal production, the activities are mostly related to creating required product features or to supporting this feature creation. Production-centric costs are costs that are caused by activities directly related to the production of goods utilizing the analyzed robot system as a means of production. In the production phase, the process models represent actual production processes.

Activities causing production-centric costs are usually highly repetitive and carried out many times a day. They are governed by the production schedule which defines the actual plan of production tasks that are to be carried out by means of the robot system. In this phase, the production schedule is therefore the key element used for scheduling cost-causing activities in the cost-benefit model.

The high repetitiveness in the production phase makes it necessary to distinguish between the product in terms of a product type or product design and actual instances of the product. The product or product type refers to an information model of a specific product that collects all information that is required to manufacture the product. The product instance or also called physical product refers to one particular physical product entity that is manufactured and conforms to a specific product type. Each product type is associated with a set of activities that need to be carried out in order to manufacture the demanded product features. These activities have to be carried out for each physical instance of the product to be manufactured.

In most manufacturing scenarios, several instances of the same product type are manufactured directly consecutively in a lot in order to minimize preparation efforts. Each lot requires specific preparation and post-processing efforts that are also attached to the product model, but only invoked in the model if the previously manufactured physical product is of another product type as the current one. Maintenance and repair activities interrupt the production use of the robot system. This is implemented by assigning a high priority to these activities in comparison to the regular production activities. Chapter 5 provides examples for activity models.

Including the views of different stakeholders

As already outlined, the cost-benefit model aims at integrating knowledge from different stakeholders. Also, the output of the cost-benefit model should reflect the views of these stakeholders. For this purpose, cash flows generated in the model are classified in three different types (see Figure 4.4) that are explained in the following.

Type 1: Cash flows that constitute payments by the system integrator to a third party, such as the purchase of equipment for building the robot system, or efforts of the system integrator, for example the effort caused by personnel of the system integrator.

Type 2: Cash flows that are payments of the end-user to the system integrator, for example the payment for the robot system or payments for service contracts or additional services.

Type 3: Cash flows that are direct payments of the end-user to third parties or efforts that are purely on the end-user side, for instance the purchase of raw material for production, the effort caused by own material.

This distinction allows to generate specific views for the different stakeholders on costs and benefits. For the view of the system integrator, only cash flows of type 1 and type 2 are relevant. For end-users only cash flows of type 2 and type 3 are included. End-users that perform system integration themselves will only see type 1 and type 3 cash flows in case they want to assess the project as a whole.

Therefore, this classification of cash flows allows to display the life-cycle cost of the machine from different perspectives while utilizing the same cost-benefit model. Due to this classification, cash flows irrelevant to the customer view can be removed from the model in order to hide internal business details before handling cost data to customers. The view of component suppliers is reflected by being able to take the view of their customers by using the cost model. Component manufacturers are therefore able to run market analyses based on their products' performance characteristics and hence being able to position their products on

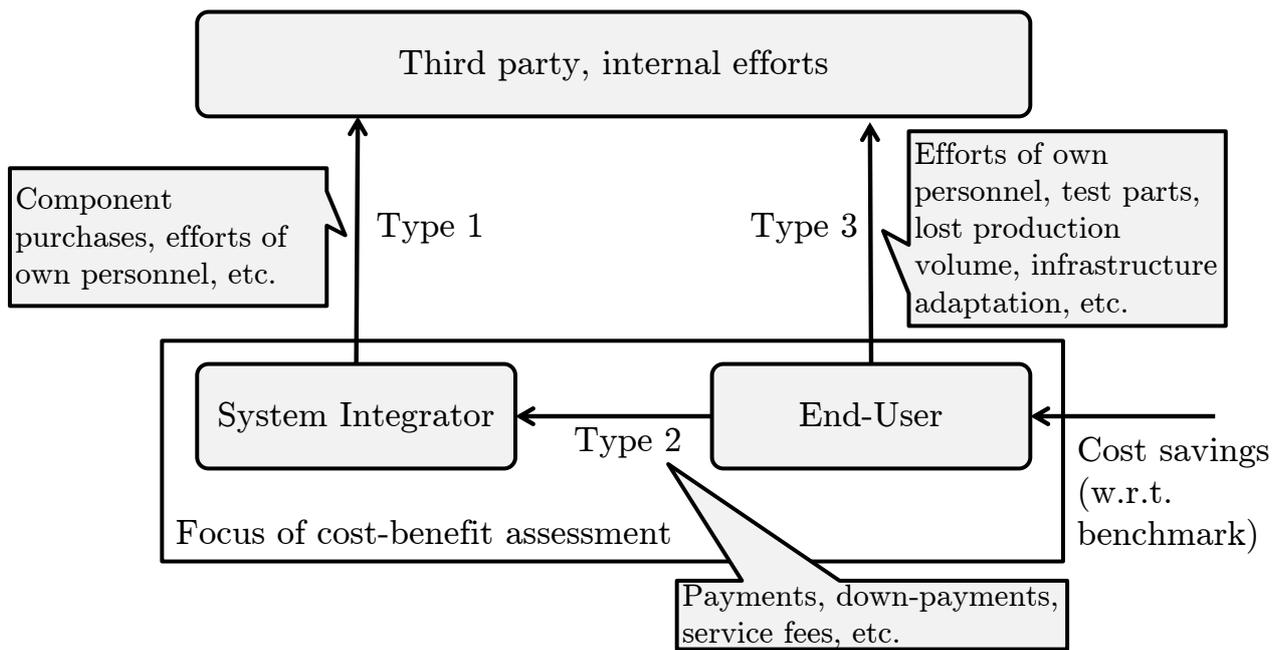


Figure 4.4 Classification of cash flows for system integrator and end-user view.

the markets correctly. The Evaluation in Chapter 7 provides examples for the views of different stakeholders.

Quantification of benefits

The cost-benefit model allows to determine the cash flows in terms of cost occurring over the entire life-cycle. As each activity carried out around the robot system causes cost, the accumulated cost figure monotonically increases as time advances. In contrast to this cost, the benefit is resulting from the value creation through the manufacturing process. To determine the cost effectiveness of the robot system, the value creation through production has to be put into perspective with the costs. There are two fundamental ways of doing this.

Firstly, it would be possible to attempt modeling the value creation of the performed production activity in terms of sales revenues and profits of the manufactured products. This figure could be directly associated with the cost per unit to determine the net present value (NPV) of the robot system. However, typically robot systems only execute a small portion of the overall manufacturing process.

It is not always easily possible to quantify the benefit of producing or assisting with the production of a certain feature of the product in currency units. While the benefit from producing the final product can typically be quantified based on accounting information, this is not always possible for all single operations required for its creation without a certain level of arbitrariness in distributing the overall product value to its features. Furthermore, such a model would require extensive inclusion of product marketing aspects as the value of the product and its features depends on achievable market prices. It would furthermore muddle strategic aspects and aspects of price formation with the capital budgeting decision. This would de-focus the model from its actual purpose and make it hard to adapt to different companies as this information is highly specific. That makes this first approach unsuitable for most robot systems.

Secondly, the benefit of the robot system can be evaluated in the perspective of alternative solutions for manufacturing that act as a benchmark. This approach is common in capital expenditure budgeting. It allows to compare different strategic investment options with each other. The selection of the benchmark scenario is the result of the strategic process of the end-user company. The benchmark can be any means of production for the same production schedule as for the system under scrutiny. For automation decisions in SME production, typically a manual, often already existing production scenario is used as a benchmark. For the benefit quantification, the cost-benefit model is built and computed for the actual robot system and compared to the cost of the benchmark solution. The cash flows resulting from the use of this benchmark system are then taken as zero line and cash flows resulting from the system under scrutiny are compared to this benchmark. This comparative procedure allows avoiding the necessity to quantify the benefits resulting from the value creation of the product feature by comparing two options of realizing the demanded production schedule. The downside of this approach is that the performance in terms of costs and benefits of the robot system in different product scenarios can only be evaluated in a scenario-based approach. Nevertheless, this approach appears useful and is chosen as the aim here is to support decisions between options on how to set-up production and not whether or not to produce and market a certain product.

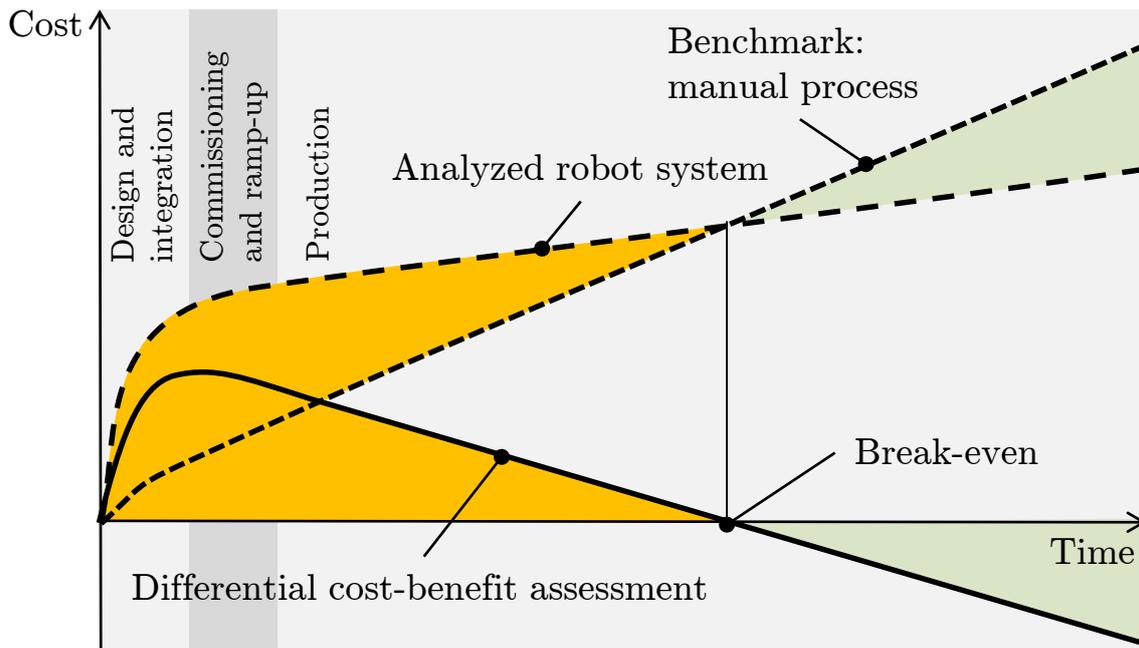


Figure 4.5 Differential cost-benefit assessment through comparison of an analyzed robot system with a benchmark.

Figure 4.5 schematically shows an example for the differential assessment of the benefit. In this figure, a robot system is compared to manual production as benchmark. In this example investment cost of the manual solution are lower, which is noticeable by the lower rise of the cost curve in early project phases. However, running cost per unit is higher due to the higher manual work effort. When subtracting the cost of the automated solution, an amortization curve is obtained that also indicates the break-even point for the investment.

The treatment of the benchmark case depends on the project scenario. In case the new robot system is to replace an existing machine, a comparison to the existing machine as a benchmark is advisable. When comparing to the benchmark scenario, the investment cost of the benchmark might be included. In this case, the benchmark is treated as an alternative investment option. Alternatively, the investment cost of the benchmark might be neglected. In this case, the means of production of the benchmark scenario already exist, have been completely depreciated and cannot be used otherwise.

Time-driven costs

As already outlined time-driven costs are treated separately from activity-driven costs. Time-driven costs are purely machine-centric as they refer to long-term cost resulting from owning and operating the robot system. In the following, the most common time-driven cost items are explained. Cost of capital refers to the interest rate paid for the capital invested in the robot system. The depreciation of the investment in the robot system is relevant in case the end-user desires a cost-benefit computation according to accounting rules. The cost of space occupied by the robot system might play an important role as space is scarce in many production environments. The cost of leasing of the robot system is particularly relevant if a sell and lease-back approach is applied. Further time-driven cost might result from equipment required for the operation of the robot system and the cost for infrastructure required by the robot system, for example recurring cost for supply infrastructure for specialized media. Table 4.2 provides an overview of time-driven costs for robot systems and further explanation. Time-driven cost are modeled directly on the cost-pool level and lead to continuous cash flows (see Section 4.2.5).

4.2.3 Modeling of activities

Activity models are the fundamental building blocks of the cost model as already shown in the schematic in Figure 4.3. Activities occur on process, resource, and cost-pool level and are defined on these levels as follows. A process activity is the act of creating a desired product feature or generating desired product properties by employing a manufacturing technology. On the resource-level, the activity definition is focused on the utilization of the means of production caused by the process. A resource activity is an act of using a production resource (i.e. machine or personnel). The definition of resource activities corresponds to typical activity definitions that are common in costing as it relates to cost-causing manufacturing resources carrying out work. The cost-pool captures the cost caused by consumption of commodities and the involvement of indirect departments. A cost-pool

Table 4.2 Time-driven cost for robot systems.

Category	Examples	Life-cycle phase
Running infrastructure costs: all cost that result from providing the required infrastructure for machine operation	Fixed cost of building, administration, availability of technicians	Operation
Cost of capital: cost resulting from capital being bound in the automation equipment	Interest on lend money, opportunity costs	Operation (end-user), integration, commissioning and ramp-up (depending on payment conditions)
Cost of space occupied by the robot system	Cost for rent per m ² , calculatory cost for space, opportunity cost	Ramp-up, operation
Cost for supply of specialized fluids and media	Cost for pressurized air infrastructure, power supply infrastructure	Operation
Depreciation of equipment as an alternative to cost of capital of investment	Reduction of book value of machine	Operation
Cost of leasing of the robot system ¹ or equipment required for the operation of the robot system	Cost for robot system, cost for measurement equipment	Integration, operation

¹ In particular relevant if a sell and lease-back approach is applied for the robot system.

activity is the act of using commodities and employing indirect departments in the company in order to carry out production.

It is critical for the cost-benefit model (again, see Figure 4.3), how the activities are represented on the different levels by activity models. Activity models \mathcal{M} allow a transformation of inputs (cost drivers, \mathcal{C}) into outputs (activity levels, \mathcal{A}) as shown in Figure 4.6. Typically, activity models are represented as algebraic equations or by using look-up tables that work on interval numbers. The models are parameterized depending on the specifics of the automation project and the employed resources.

Cost drivers \mathcal{C} are measures that cause effort on a particular level of the cost-

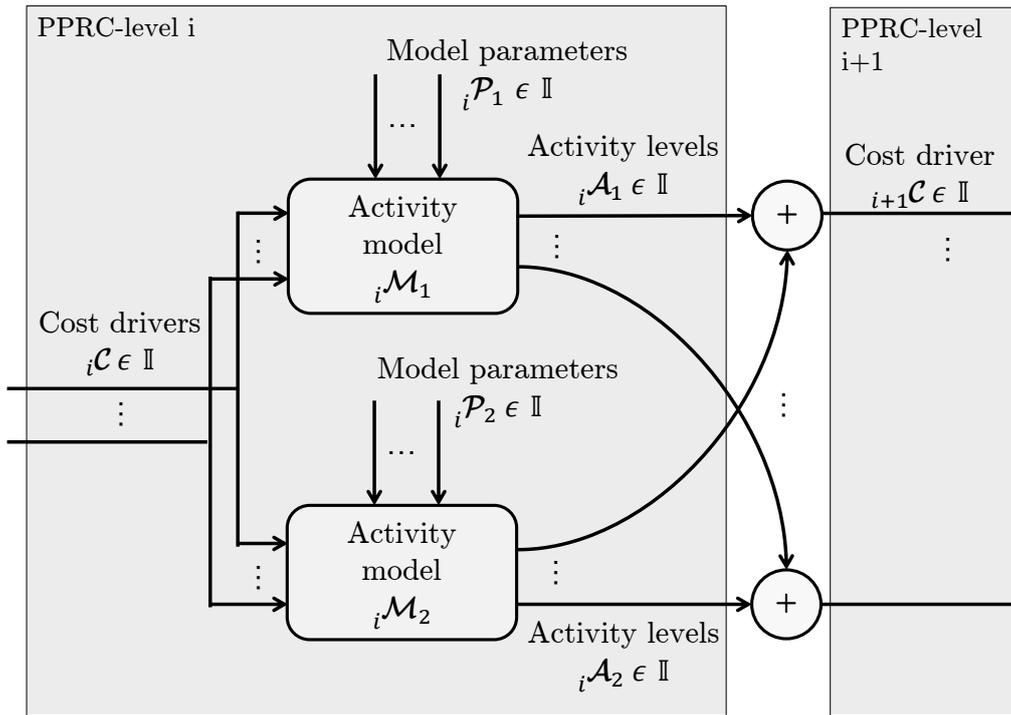


Figure 4.6 Schematic of activity models and their inputs and outputs.

benefit model and hence are the inputs to the activity models on each PPRC-level. The term cost driver in the context of this thesis is not only applied to drivers that directly cause costs but also as driving quantities of activities that cause efforts that are not immediately related to cost, i.e. on process and resource level. These efforts finally lead to costs when traced back through further layers of the PPRC-model. A more accurate term for cost drivers in this context in this context would be activity driver or effort driver. However, as the term cost driver is well established in literature this term is used.

An activity level \mathcal{A} is defined as a measure for the amount of an activity that is actually carried out to achieve a specified production purpose. Therefore, activity levels are the outputs of activity models like product, process, resource or cost-pool models. This output can either be a measure specific to the activity or related to direct cost which is then processed directly in the cost-pool, for example for purchase cost of raw material and components.

An activity model is described by:

$${}_i\vec{\mathcal{A}}_k = {}_i\mathcal{M}_k \left({}_i\vec{\mathcal{C}}, {}_i\vec{\mathcal{P}}_k \right), \text{ where} \quad (4.1)$$

i is a prescript indicating the level in the PPRC-hierarchy from product level ($i = 1$) to cost-pool level ($i = 4$). The subscript k indices the activity models on level i . The activity model ${}_i\mathcal{M}_k$ is a function ${}_i\mathcal{M}_k : \mathbb{I}^m \rightarrow \mathbb{I}^n$, where m is the number of input cost drivers and n is the number of activity levels. ${}_i\vec{\mathcal{C}} \in \mathbb{I}^m$ is the vector of cost drivers and ${}_i\vec{\mathcal{A}}_k \in \mathbb{I}^n$ is the vector of activity levels. ${}_i\vec{\mathcal{P}}_k \in \mathbb{I}^r$ is a set of parameters of the activity model ${}_i\mathcal{M}_k$, while r is the number of required parameters of the model. All input and output quantities of the activity model are interval numbers allowing to take into account uncertainties.

As shown in Figure 4.6, cost drivers ${}_{i+1}\mathcal{C}$ of a subsequent level are obtained by summing-up all activity levels ${}_i\mathcal{A}$ of same type t from the preceding level

$${}_{i+1}\mathcal{C}_t = \sum_n {}_i\mathcal{A}_t. \quad (4.2)$$

The following sections introduce specifics of the activity models on the different levels on the PPRC-hierarchy.

Product models

The product models are structurally different from activity models on other PPRC-levels as they do not possess inputs (cost drivers). They supply information on the desired target state of the product, i.e. they describe what needs to be manufactured. This means the product model contains activity levels that can be directly translated to the cost drivers for the process level. These activity levels are typically directly associated with desired product features. The product model is therefore an information model and not a mathematical model. On the product level, two types of product feature descriptions are distinguished.

Product parameter: Parameters that describe a product feature and are related to the effort of producing this feature. Product parameters can directly be translated into activity levels on the product side and hence to cost drivers on the process level. The underlying assumption is that the features of the product are produced sequentially as it is typically the case in robotic production, hence avoiding problems with feature-based costing for simultaneously manufactured features as outlined by Wierda, 1991.

Product properties: Product properties describe the product and its manufacturing needs by assigning an attribute to a specific property of the product. Product properties are used to determine parameters of the underlying process model. An example is the interface of a sensor integrated into the robot system. In this example, the product property is the interface type. The cost model determines the effort for integration with this interface depending on the interface type. In this example the integration effort corresponds to a parameter on the process level, while the interface type is the product property used to determine this parameter.

Product models and their features are process dependent. Table 4.3 provides examples for product parameters and properties for typical processes to give a basic understanding of the approach.

Additionally to the description of product features that act as drivers of production activities, product models carry information about the required purchase efforts for components or raw material. This cost is directly fed to the cost-pool. It would be possible to model the purchase activities using the suggested PPRC-approach. This would bring the cost model closer to total-cost of ownership (TCO) computations where for example the cost of vendor relations is considered. However, the cost of the relation to the supplier is neglected as this work is focused on cost directly induced by the robot system.

Table 4.3 Product model parameters and properties for typical processes.

Process	Description	Parameters and properties
Arc welding	MIG, MAG or TIG line-weld seams used to join sheet metal of the product	<ul style="list-style-type: none"> ○ seam cross-section, mm² ○ seam length, m
Spot welding	Spot weld for sheet metal joining	<ul style="list-style-type: none"> ○ Sheet thickness, mm ○ spot size, mm ○ material
Painting	Painting and similar processes that involve the even application of a liquid to a surface	<ul style="list-style-type: none"> ○ surface to be covered, m² ○ layer thickness, mm
Pick and place	Transport of a part by a robot from position \vec{l}_A to position \vec{l}_B	<ul style="list-style-type: none"> ○ pick position \vec{l}_A, m ○ place position \vec{l}_B, m ○ part weight, kg
Deburring	Deburring of parts with linear burr, e.g. due to interstice of casting mold	<ul style="list-style-type: none"> ○ edge length, m ○ burr cross section, m² ○ material
Machining	Machining processes to mill work pieces out of a solid block by removing material	<ul style="list-style-type: none"> ○ machining volume, mm³ ○ material
Glueing	Application of adhesive beading to parts	<ul style="list-style-type: none"> ○ adhesive volume per length, mm³/m ○ beading length, m
Drilling	Drilling of holes into the work piece using a spindle and drilling tool	<ul style="list-style-type: none"> ○ hole diameter, mm ○ hole depth, mm ○ material
Riveting	Joining of pre-drilled parts using pop rivets	<ul style="list-style-type: none"> ○ hole diameter, mm ○ rivet length, mm ○ rivet diameter, mm

Process models

Process models assign a process-specific level of activity to a product feature. Typical process models translate the production requirements of the product into constraints that are imposed by the process such as feed rates or execution times. In many cases, the process activity is described through the activation time of the process t_A which measures how long it takes to carry out the process.

Many cost items are scalar and a linear relation between process-level cost driver, ${}_2\mathcal{C}$, and process activity level, ${}_2\mathcal{A}$, can be assumed resulting the model description

$${}_2\mathcal{A}_k = {}_2\mathcal{P}_k {}_2\mathcal{C}_v, \quad (4.3)$$

where the parameter ${}_2\mathcal{P}_k$ of the linear relation is called the characteristic activity amount. The index v is the index of the cost driver in the cost driver vector ${}_2\vec{\mathcal{C}}$ that is relevant for the model.

More complex implementations of process models using other functions, solvers or simulations are also possible in the developed costing framework (see also Chapter 6). For example, the inclusion of a process simulation model of a PLM-tool to determine process constraints is possible. In this case, the simulation is typically used to determine parts of the process parameters ${}_2\vec{\mathcal{P}}$.

Process models can trigger dependent processes whose execution is either a pre- or post-condition. For example, the welding of a seam requires to approach the weld joint before welding and to retract from the weld joint after welding. The installation of a sensor requires the design, manufacturing, and assembly of a cantilever to carry the sensor. These conditions are stored in the knowledge base and invoked during instantiation of the model. How this connection of knowledge base and cost model is used in order to complete missing information in the cost model is explained in Section 4.3. Chapter 5 provides examples for process models.

Resource models

Resource models provide information on the utilization of the means of production in the manufacturing process. The input cost drivers relate the process constraints, such as required execution times and maximum process speeds. The resource model transforms this information into an activity level describing the resource utilization, i.e. the time the resource is blocked by the process, and the use of commodities caused by the resource utilization, for example the use of electric energy.

Resource models are different from process models as they relate to actual physical instances of resources available on the shop floor. While process models can be instantiated in unlimited number as they only relate to work efforts required to manufacture the product, the cardinality of resource models is fixed. Therefore, the resource models are also decisive for the scheduling of tasks in the production system.

All resource models must have the resource occupancy time t_{occ} as an output activity level in order to provide information on the resource allocation by the manufacturing process. The occupancy time describes how long this resource will be occupied by a running process and is not available for other processes. This output is needed to schedule work tasks and compute the time frame of the machine operation. Resources that have several instances are either modeled separately or have a resource cardinality property allowing the model to accept several tasks simultaneously. This reflects that, either the identical, interchangeable resource is available several times or that the resource is capable of executing more than one task at once.

Cost-pool models

Typically, the cost drivers on cost-pool level refer to the usage of primary resources or personnel resources. Cost-pool models transform these into actual cost by applying overheads and keys of payment from accounting. The output of the

cost-pool model is always a cash flow, i.e. the occurring cost in currency units and the time or time interval of occurrence. Cost-pool models capture for example commodity prices and overheads.

Two types of cash flows, discrete and continuous cash flows, are distinguished. A discrete cash flow is a cash flow that occurs at a discrete point in time in which the valuable asset is consumed or changes the owner. A typical example of a discrete cash flow is the cash flow resulting from the purchase of material or components. The exchange of money and material can be modeled as taking place at a specific point in time². A continuous cash flow is a cash flow that occurs over a certain duration, typically assuming a constant cash flow rate.

Resource-related cash flows are modeled as continuous cash flow and the duration of the cash flow is assumed to be the occupation duration of the related resource. This is not entirely accurate as often the actual cost does not occur during the usage of the resource. Typically, continuous cash flows could be decomposed into one or several discrete cash flows referring to the actual transfer of assets. For example, personnel efforts are charged at the end of one month or even later in case of collected overtime. However, this simplified approach decouples the model from accounting details specific to the company which is not directly related to the technical details of the robot system. Furthermore, it charges cost, when the respective effort occurs focusing the view of the model towards a robot system itself.

It would be possible to model complex cost creation behavior in accounting, such as usage depending prices, e.g. for electricity, or payment targets for paying bills. However, the models here are typically simple overhead models of the form

$${}_4C = (1 + {}_4o/100) {}_4C_v, \quad (4.4)$$

where ${}_4C$ is the resulting cost of the discrete cash flow, or the cost rate of a

²Of course the purchase of material or components might involve complex payment and ownership modalities. However, these are neglected here for evaluating the cost-effectiveness of robot systems.

continuous cash flow, ${}_4o$ is the overhead rate in percent and ${}_4C_v$ is the relevant cost driver of the cost-pool model.

4.2.4 Configuration of the cost model

One of the key problems for cost-benefit assessment of robot systems is the limited availability of input data for the cost model. This lack of information relates to the structure of the required model and its parameters. The artifacts generated during system design, in particular the data generated by the used PLM tools, contain information on the general set-up of the system and its components. To facilitate the use of cost-benefit assessment by drawing from this information, the model is tightly integrated with engineering data formats used during plant design. By reading data from these engineering models and cross-relating them to a knowledge base with costing information (see Section 4.3), the cost-benefit assessment tool is capable of determining a large portion of the required input data and its uncertainties with a minimal level of user input.

AutomationML as input format for cost model configuration

The aim of the cost-benefit model is to determine the cost and benefits resulting from a specific design and usage scenario of the robot system. In order to realize the required tight integration with the development process of the automation solution, the vendor neutral data exchange standard AutomationML is used for input. Data regarding to system components, products and processes is extracted from the AutomationML file as described in the following.

System components: the single components of the robot system specified by type and vendor and their role in the robot system.

Products: the products produced with the robot system and relevant cost drivers (e.g. the length and dimensions of weld seams in a work piece).

Processes: The executed manufacturing processes required for the product features contained in the product information.

Appropriate roles for semantic grounding of the above mentioned information were defined (see Appendix 2). These roles can be exported into a role library for use in AutomationML. The role information allows the extraction of required information from the knowledge base even if an exact match of the identified component is not available or if required information for this match is missing. The resolution of uncertainties and extraction of missing information in the AutomationML description of the robot system is described in Section 4.3.

Process for building the cost model

In the following, the set-up of the cost model for a specific robot system design from AutomationML data is described. Figure 4.7 shows how the different sources of information are combined during the instantiation of the cost-benefit model. Figure 4.8 shows the schematic of the instantiation process of the cost-benefit model.

The AutomationML description of the work cell from the plant engineering tool chain is parsed and transformed into an RDF description for easy access to information. Section 4.3.1 provides details on this transformation. In a first step, the machine-centric parts of the cost-benefit model is instantiated. For this purpose, the components of the robot system are analyzed with respect to required information and their role. Detailed information on the components is retrieved by matching them to entries in the knowledge base (see Section 4.3.3 for details on this matching process). A check of system completeness is performed by checking that all required roles of the robot system are present in the bill of material. This check is performed with respect to role templates of robot systems in general or application-specific templates and is required to ensure that the description of the robot system is complete. In case required roles are not reflected by the contained components, the empty slots are filled with generic components from the knowledge base. For example, if no safety equipment is contained in the design, the

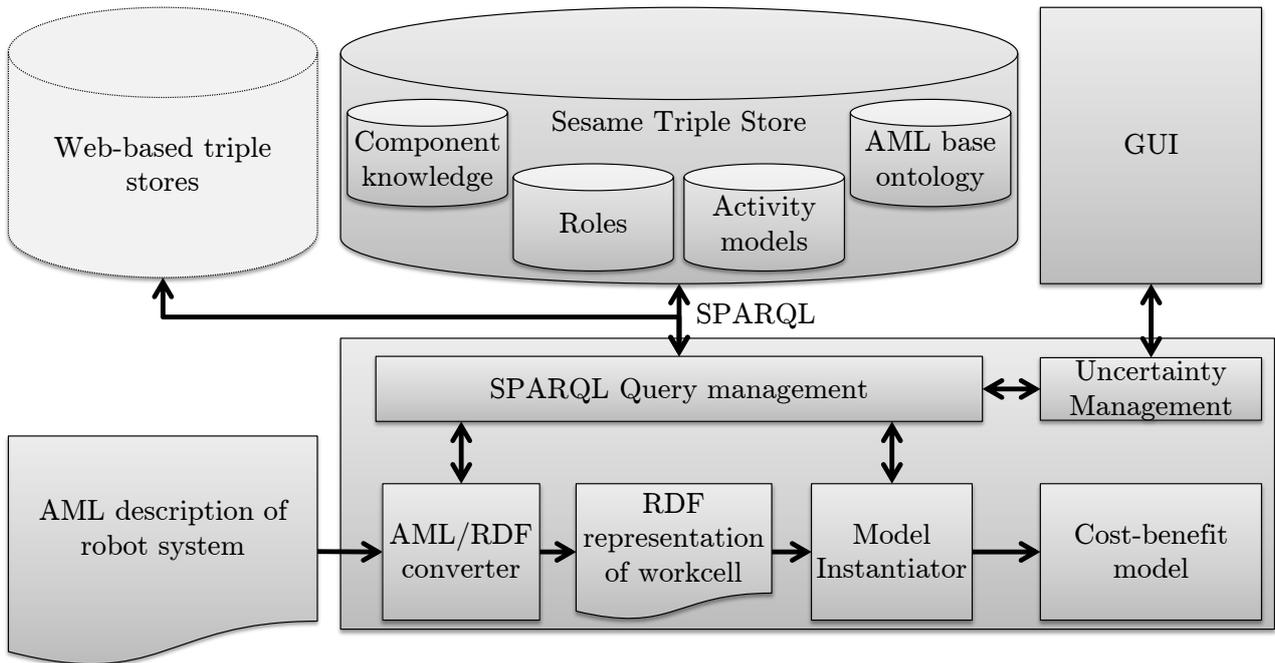


Figure 4.7 Combination of different sources of information during instantiation of the cost-benefit model.

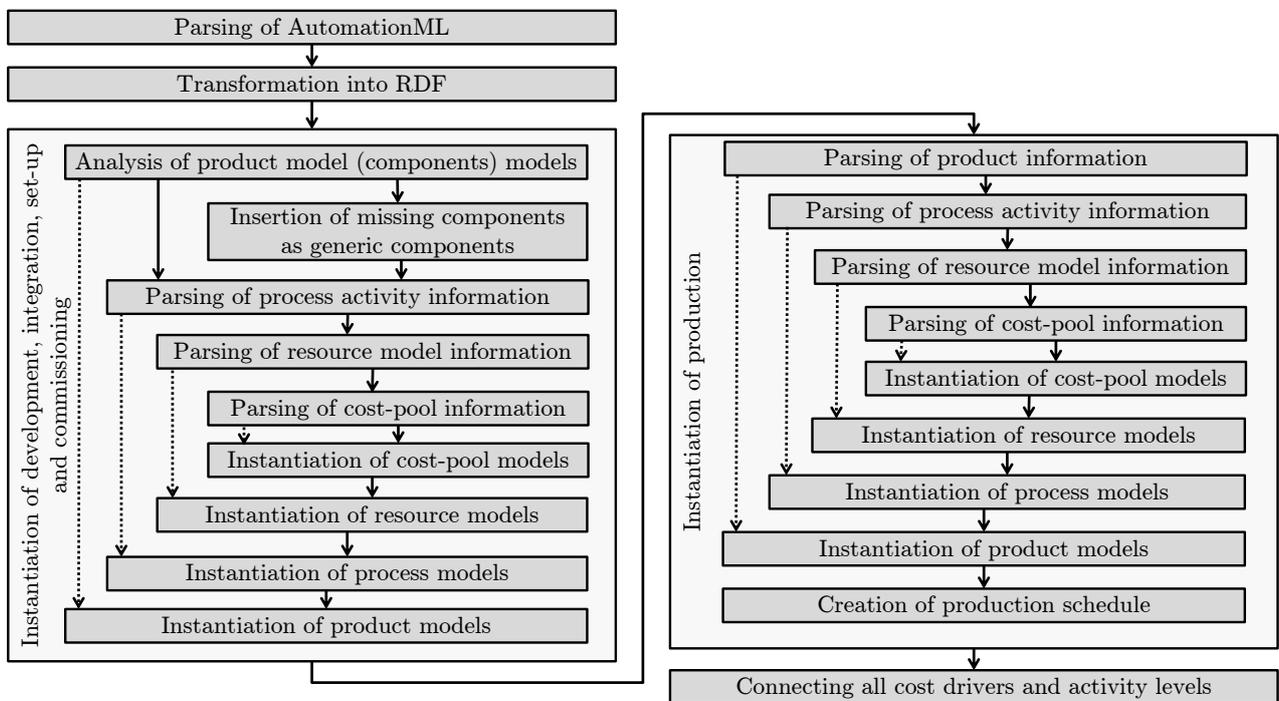


Figure 4.8 Schematic of the process for model configuration.

slot is filled with a generic safety fence. With this completed bill of material, the instantiation of the cost model for the design phase is carried out.

Subsequently, the activities that are required for the integration, commissioning, and ramp-up of the contained components are retrieved from the knowledge base (see left part in Figure 4.8). Detailed information on these activities such as the models suitable for the description, related parameters, and the link to required resources are extracted from the knowledge base. This activity information also points to the required resources and allows retrieving information on them. The representation of the resource in the knowledge base points to the cost-pool items attracted by the employed resources and through this, the cost-pool models and parameters are found. Subsequently, the models are instantiated in reverse order of information retrieval, i.e. working from the cost-pool level towards the product level. The instantiation in reverse order is related to the implementation of the model (see Chapter 6) through a delegate mechanism that allows models on PPRC-level i to automatically invoke the models on level $i + 1$.

After the machine-centric model has been built, the production-centric model is generated (see right part in Figure 4.8). The general process for this is comparable to the machine-centric model. However, here the actual products manufactured by the robot system form the entry point of the instantiation process. After the instantiation of the product model, a production schedule is generated. The parameter information on the resource level for the production-centric model is drawn from the machine-centric product model as explained in the following section.

Parameterization of production-centric resource models

The production-centric resource models and machine-centric product models possess a strong bond, as the performance of the machine during production is determined by its composition and components. The models of the components of automation systems, i.e. the product models during the design phase, reference the performance parameters of these components in the knowledge base. This

information is used to parameterize the resource model in the usage phase, because these models relate to the same components. In this way, the cost model of the robot systems' usage is built and parameterized based on the robot systems' design information.

Hence, the model descriptions and parameters of the production-centric resource model are derived from the machine-centric product models, by performing a match between the required production resources and available components in the robot system. This match is carried out based on roles, as the production activities only relate to required resource roles and not to concrete components. All components are assigned possible roles. Possible roles refer to roles that these components may take in a production task. An example is an industrial robot which can be used as a handling device or a positioning device. Also, the required roles of manufacturing processes are defined. To carry out the manufacturing process, at least one component of the robot system is required to take this role. This allows a matching of roles required by the process and the roles of components in the robot system. This match is performed between the resource level in the nominal production (i.e. the restrictions on processing performance implied by the production machinery, for example the handling speed) and the product level in the system integration phase (i.e. the physical properties of the robot system itself, e.g. maximum robot speed). In case the matching is ambiguous or no component is found taking a role required by the process, the plant designer is asked to resolve the conflict by resolving ambiguity or specifying which component will take a required role. Through this, the resource models for the production life-cycle phase can be instantiated from the product model used during system design and development (i.e. the model of the robot system).

4.2.5 Handling of uncertainty in the cost-benefit model

The handling of uncertainties has particular relevance for robot systems as they are typically built as special machinery and exact performance characteristics or cost data are not known exactly during the design phase. This lack of information

leads to deficiencies in decision making on the selected system concept and the selected components. Firstly, the decision for or against an investment taken by the end-user only considers a nominal scenario or few alternative scenarios. Today, the key parameters on which the cost-effectiveness of the automation system depends are usually not systematically identified and taken into account during the investment decision. For example, an automation system with a higher price might offer a better compromise as its cost-effectiveness might be less sensitive with respect to uncertain parameters of the cost model. In turn, this would result in a cost-benefit performance that is more resilient to changes in the market and deviations from the planned production scenario. Furthermore, design decisions, for example including or omitting particular features, have to be made in the face of the described opacity and lack of information regarding costs and benefits. A cost-benefit model should help to quickly assess what costs a design decision causes not only in the form of purchase prices of the components, but also in terms of efforts in system development, installation, and maintenance effort and the related level of uncertainty of these predictions.

Types of uncertainties in cost modeling

Uncertainty in cost modeling has several forms, stems from various sources and impacts the cost-benefit assessment on various levels. Therefore, to address uncertainties in the cost-benefit model, different types of uncertainty have to be distinguished. Uncertainties are classified with respect to different dimensions, in particular the following three.

Object: The uncertainty is related to the system as a whole or to single system components.

Type: The uncertainty is epistemic, i.e. due to lack of knowledge, or aleatory, i.e. due to underlying randomness.

Scope: The uncertainty concerns data, identity, or the model.

In the following, a classification of uncertainty for cost-benefit assessment of robot systems is proposed. This classification is based on existing classifications as outlined in Section 2.2 and adapted to the cost assessment process of robot systems.

Composition uncertainties During the design process, an initial design of the robot system is continually refined. In particular, this process involves the definition and specification of the single components. Therefore, in early design phases, the exact composition is not yet defined. A composition uncertainty is the uncertainty regarding the single components, i.e. hardware and software, of which the robot system is composed. Composition uncertainties are related to the system as a whole, epistemic in nature, as the information has not yet been created and related to identity.

Association uncertainties Composition uncertainties relate to not knowing the single components of which the robot system is set-up. As soon as the composition uncertainties are eliminated, the single components as building blocks of the robot system are known. However, this does not mean that the exact identity of these components is known. Uncertainties regarding the identity of the devices are called association uncertainties. They become apparent when additional data regarding the components has to be retrieved. Association uncertainties can be epistemic and aleatory in nature. Aleatory association uncertainties occur when a system component has been defined in general terms, but the exact type of the component in terms of brand and model has not yet been defined. Epistemic association uncertainties occur when the exact model of a component has been defined, but no matching data source to retrieve further information can be identified unambiguously, for example because of misspelling of product codes and names. Association uncertainties are related to single components, related to identity and aleatory or epistemic in nature.

Parameter value uncertainties The cost-benefit model requires parameters in order to be computed. Parameter value uncertainties occur when no value for a required parameter is known. They are epistemic in nature as they refer to a lack of knowledge. Parameter value uncertainties are attached to single components and are related to data.

Parameter variability uncertainties It might be impossible to exactly determine the value of required parameters of the model due to an inherent uncertainty. Parameter variability uncertainties reflect the inherent uncertainty of parameter values of the model. They are aleatory in nature and refer to data. However, they might also have epistemic components stemming from faulty parameter estimates. Parameter variability uncertainties occur for example for component prices which might differ depending on numerous factors, e.g. depending on market situation, customer, purchase volume, and negotiation ability of the customer and salesperson. Parameter value and variability uncertainties are closely related as the estimation of an unknown parameter (parameter value uncertainty) typically leads to different estimates resulting in uncertainty regarding the exact parameter value (parameter variability uncertainties).

Time uncertainties Time plays a special role in the cost-benefit model as it governs the sequence of events and activities and is irreversible. Time uncertainties reflect the uncertainty with respect to the exact time of an event. Time uncertainties are closely related to parameter variability uncertainties as the time is also a parameter of the used cost models. Time governs the causal relationship of events in the event-driven part of the model and hence also time uncertainties influence the causality of activities in the model and therefore its consistency. Take for example the causal relation between two events. This means that one event, for example the start of production of product B, takes place in time after another event, for instance the finalization of product A. This causality in combination with time uncertainties of the start and end points of events would lead to an exponentially increasing number of alternative scenarios of the cost model

as shown in Figure 4.9. The figure shows how the chain of causality of activities with time uncertainty leads to an increasing number of alternating scenarios if more causally connected activities are added. Hence, for keeping also large models responsive when interacting with the user, a separate treatment of time uncertainties is required.

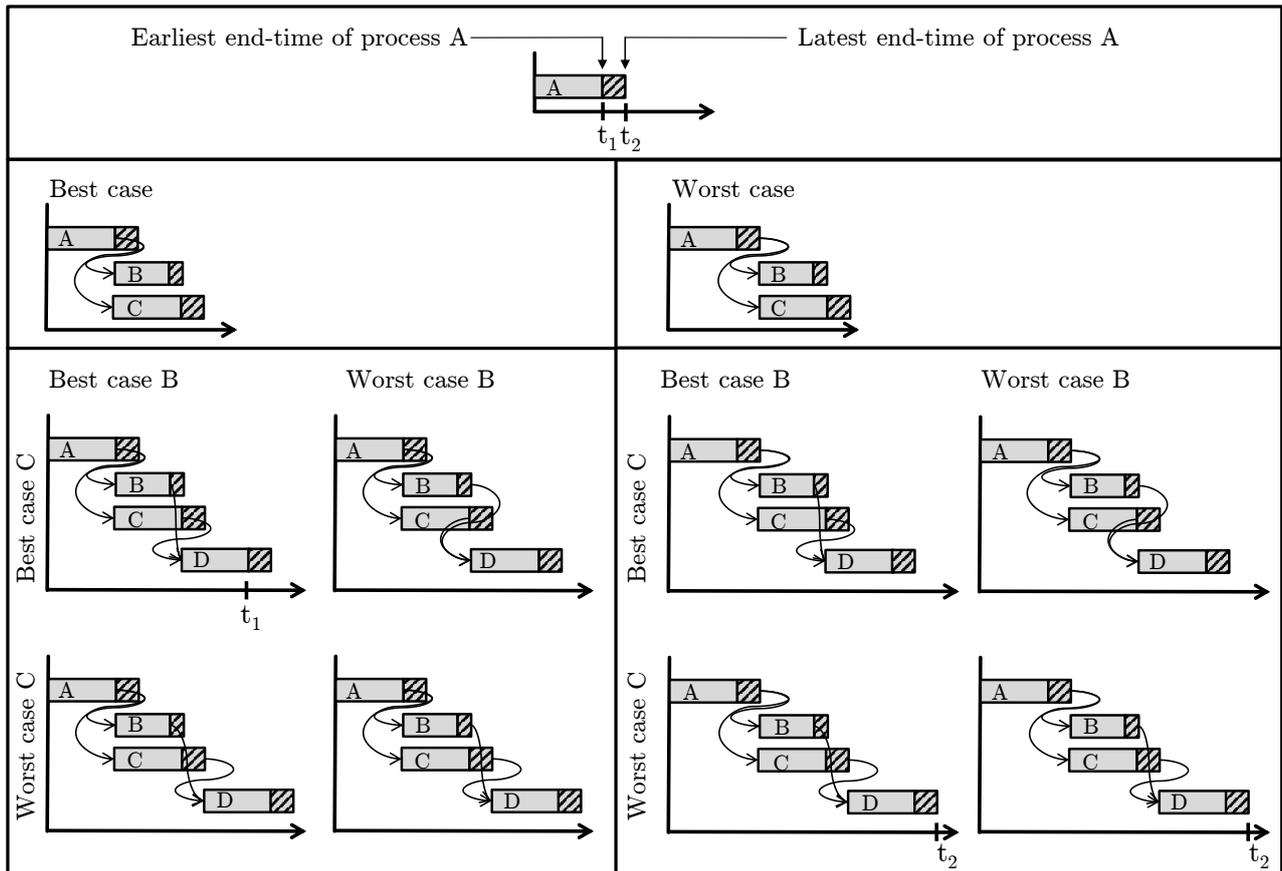


Figure 4.9 Alternative scenarios of an event-triggered system of four processes with causality conditions (curved arrows).

Model uncertainties Models are an idealized, simplified description of reality. Due to this fact, models neglect some aspects of real systems. Model uncertainty relates to the uncertainty introduced by the deviation of the model from the real world properties of the modeled process. Model uncertainties occur in every modeling process. The effect of model uncertainty on the model results is not further analyzed in the scope of this thesis. The activity models reflect uncertainty in terms of parameter variability uncertainties, but do not explicitly address the aspect of model uncertainty.

Uncertainty in the cost model

All above mentioned types of uncertainties have to be addressed in the cost-benefit assessment. However, many of the uncertainties such as uncertainties relating to identity or parameter value uncertainties are only relevant during the instantiation phase of the model where the required information to parameterize the model needs to be found. Once this uncertainty is resolved, an uncertain estimate of the related property is available. Hence, the elimination of identity and parameter uncertainties is subject to the instantiation process of the cost-benefit model. In the cost-benefit model itself, all uncertainties are reflected as parameter variability uncertainties. In order to resolve all uncertainties to parameter variability uncertainties during the instantiation of the cost model, a knowledge base storing typical parameters is used. The resolution process for the different types of uncertainties is described in detail in Section 4.3.3.

The result of the resolution process is an interval number for every uncertain model parameter. This interval number is interpreted as an upper and lower bound for the quantity under scrutiny. This upper and lower bound is typically obtained through the interval spanned by all possible estimates of the underlying parameter. For example, all found prices $[C_1, \dots, C_n]$ where $C_i \in \mathbb{R}^n$ for a component span an interval $C_I = [\inf(C_i), \sup(C_i)] \in \mathbb{I}$. This resolution process allows to reflect all uncertainties as interval numbers and to use the computational rules of interval arithmetics (see Section 2.2.3) for uncertainty propagation in the model. The result of the model are uncertain cash flows computed on the cost-pool level. These cash flows are then discounted to obtain the NPV of all cash flows to a common time basis as outlined in the following section.

Discounting of uncertain cash flows

Cash flows reflect the time value of money as they contain a temporal component specifying when the exchange of value takes place. In order to compare cash flows they need to be discounted to the same time basis, i.e. the time value of the cash flows at the same point in time t needs to be determined. All cash flows considered

in this thesis are interval numbers and therefore called uncertain cash flows. The specifics of discounting uncertain cash flows are introduced in the following. The cash flows are discounted based on yearly interest rate q as decimal number. For discrete cash flows that occur at a single point in time, the discounting is straight forward and follows the standard present value equation

$$C(t_2) = C(t_1)(1 + q)^{\frac{t_2-t_1}{c_{dpy}}}, \quad (4.5)$$

where $C(t_i)$ is the value of a cash flow at a specific time t_i in days, q is the yearly interest rate as interval number where supremum and infimum are given as decimal number, and c_{dpy} is the number of days per year, as interest occurs yearly. The interval operations used in equation (4.5) are implemented as outlined in the state of the art (see Section 2.2.3).

The discounting of uncertain continuous cash flows requires a differentiated treatment. Consider a continuous cash flow with rate $c(t)$ occurring between time t_1 and $t_1 + \Delta t_1$. The differential cash flow can be expressed as:

$$dC(t) = c(t)dt \quad (4.6)$$

where dt is the time differential. Using equation (4.5) to discount to time t_2 yields

$$dC(t_2) = dC(t)(1 + q)^{\frac{t_2-t}{c_{dpy}}} = c(t)dt(1 + q)^{\frac{t_2-t}{c_{dpy}}}. \quad (4.7)$$

This cash flow is integrated over t from t_1 to $t_1 + \Delta t_1$ to obtain the value of the cash flow at time t_2

$$C(t_2) = \int_{t_1}^{t_1+\Delta t_1} c(t)(1 + q)^{\frac{t_2-t}{c_{dpy}}} dt. \quad (4.8)$$

In most cases, a time-constant cash flow rate $c(t) = c = \text{const.}$ can be assumed in which case the integral can be solved to

$$C(t_2) = c c_{dpy} \frac{(1 + q)^{\frac{t_2-t_1}{c_{dpy}}}}{\ln(1 + q)} \left(1 - (1 + q)^{\frac{-\Delta t_1}{c_{dpy}}} \right). \quad (4.9)$$

As outlined in the state of the art, great care needs to be taken in case of multiple occurrences of variables. In case of equation (4.9), not considering the multiple occurrences of the interest rate q leads to a discounted cash flow with large overestimation. In the following, it is shown that the discounted continuous cash flow is monotonic with respect to q under certain assumptions that are applicable here. This allows to compute equation (4.9) separately for infimum and supremum of q while all possible results are still guaranteed to lie within the resulting interval.

It is assumed that the interest rate q and the time interval Δt_1 are always positive and that $q < 1$. In the cost-benefit model, all cash flows are discounted to the start of the project. Therefore it can be assumed without loss of generality, that the discounting time t_2 lies outside of the cash flow, i.e. $t_2 > t_1 + \Delta t_1$ or $t_2 < t_1$. In order to prove that cash flow $C(t_2)$ in equation (4.9) is monotonic with respect to q , it has to be shown that its first derivative with respect to q does not change sign. This first derivative is computed as

$$\dot{C}(t) = c c_{\text{dpy}} \underbrace{\frac{1}{\ln(1+q)}}_{A:=} \underbrace{(1+q)^{\frac{t_2-t_1}{c_{\text{dpy}}}-1}}_{B:=} \dots \left(\underbrace{\frac{t_2-t_1}{c_{\text{dpy}}} - \frac{1 - (1+q)^{-\frac{\Delta t_1}{c_{\text{dpy}}}}}{\ln(1+q)} - \frac{t_2-t_1 - \Delta t_1}{c_{\text{dpy}}} (1+q)^{-\frac{\Delta t_1}{c_{\text{dpy}}}}}_{C:=} \right). \quad (4.10)$$

It can be seen from equation (4.10) that $A \geq 0 \forall q \geq 0$, $A > 0 \forall q > 0$ and $B > 0 \forall q \geq 0$. Hence the product AB is zero only for $q = 0$ and has no other root. For the term C , a more detailed analysis is required in order to determine whether the first derivative in equation (4.10) can become zero. For the sub-terms in C , it holds true that

$$\begin{aligned} 0 < (1+q)^{-\frac{\Delta t_1}{c_{\text{dpy}}}} &\leq 1 && \text{as } \Delta t_1 > 0 \\ 0 &\leq \ln(1+q) && \text{as } 0 \leq q \leq 1. \end{aligned} \quad (4.11)$$

The term C can be restructured to

$$C = \underbrace{\frac{t_2 - t_1}{c_{\text{dpy}}}(1 - (1 + q)^{-\frac{\Delta t_1}{c_{\text{dpy}}}})}_{C_1:=} - \underbrace{\frac{1 - (1 + q)^{-\frac{\Delta t_1}{c_{\text{dpy}}}}}{\ln(1 + q)}}_{C_2:=} + \underbrace{\frac{\Delta t_1}{c_{\text{dpy}}}(1 + q)^{-\frac{\Delta t_1}{c_{\text{dpy}}}}}_{C_3:=}. \quad (4.12)$$

Using equation (4.11), it can directly be seen that the term C_1 is always negative if cash flows are discounted to the past, i.e. $t_2 < t_1$ (see also below). Hence, to prove that the term C is always negative it has to be shown that also the terms $-C_2 + C_3$ is always negative. This term can be reformulated to

$$C_3 - C_2 = \frac{\overbrace{(1 + q)^{-\frac{\Delta t_1}{c_{\text{dpy}}}} (\Delta t_1 \ln(1 + q) + c_{\text{dpy}}) - c_{\text{dpy}}}_{D:=}}{c_{\text{dpy}} \ln(1 + q)}. \quad (4.13)$$

The denominator of the fraction in equation (4.13) is always positive. This can be easily observed using equation (4.11). For the nominator it can be shown that the derivative of the first summand D is

$$\frac{d}{dq} D = -\frac{\Delta t_1^2}{k} (1 + q)^{-\frac{\Delta t_1}{c_{\text{dpy}}}-1} < 0. \quad (4.14)$$

Therefore, the term D in the nominator of equation (4.13) is strictly monotonic falling. As $D(q = 0) = c_{\text{dpy}}$ the term $C_3 - C_2$ is always negative for $q > 0$. Hence, term C in equation (4.10) has no zero crossing. Therefore, the function for the computation of the NPV in equation (4.9) is monotonic.

Due to this monotonicity, the discounted cash flow can be obtained, by computing the equation for the infimum and supremum of the interest rate separately, i.e.

$$C_{\text{inf}}(t_2) = \frac{c c_{\text{dpy}}}{\ln(1 + \text{inf}(q))} \left(\frac{1}{(1 + \text{inf}(q))^{\frac{t_1 - t_2}{c_{\text{dpy}}}}} - \frac{1}{(1 + \text{inf}(q))^{\frac{t_1 + \Delta t_1 - t_2}{c_{\text{dpy}}}}} \right) \quad (4.15)$$

$$C_{\text{sup}}(t_2) = \frac{c c_{\text{dpy}}}{\ln(1 + \text{sup}(q))} \left(\frac{1}{(1 + \text{sup}(q))^{\frac{t_1 - t_2}{c_{\text{dpy}}}}} - \frac{1}{(1 + \text{sup}(q))^{\frac{t_1 + \Delta t_1 - t_2}{c_{\text{dpy}}}}} \right). \quad (4.16)$$

Then the two results are combined to the uncertain cash flow

$$C(t_2) = [\min (C_{\text{inf}}(t_2), C_{\text{sup}}(t_2)), \max (C_{\text{inf}}(t_2), C_{\text{sup}}(t_2))] \quad . \quad (4.17)$$

This leads to an improved estimate for the discounted cash flow compared to direct interval computation of equation (4.9) and avoids overestimation.

Note that for all cash flows, the process of discounting combines the uncertainty of the cash flow at its time of occurrence with the uncertainty induced by the (uncertain) interest rate. Hence, discounting a cash flow $C(t_1)$ to time t_2 and subsequently back in time to t_1 leads to an overestimation of the uncertainty, as the uncertainty of the interest rate is taken into account twice: when discounting forward and when discounting backwards in time. This can be understood by observing Figure 4.10. The figure shows the NPV of a cash flow over time. As can be seen the uncertainty in the interest rate increases the level of uncertainty and hence the width of the uncertainty interval while discounting to the past and to the future. This leads to an overestimation of uncertainty if an uncertain cash flow is discounted forth and back in time. Therefore, the cost model should make sure that the discounting of cash flows only occurs into the future or into the past. For comparing cash flows in the cost-benefit model, cash flows are always discounted into the past. For example, if two cash flows $C_1(t_1)$ and $C_2(t_2)$ are compared, the operation will be carried out at $t = \min(t_1, t_2)$.

Using the equations above, it is possible to discount uncertain cash flows in the cost model while avoiding overestimation of uncertainties. This allows to introduce dynamic aspects to the cost model while treating uncertainties. The equations above are therefore of major importance for building a uncertainty-aware, dynamic cost-benefit model.

4.2.6 Scheduling and computation of the cost model

In order to produce cost estimates, the cost-benefit model needs to be computed for the entire life-cycle of the robot system. This means that a schedule of the

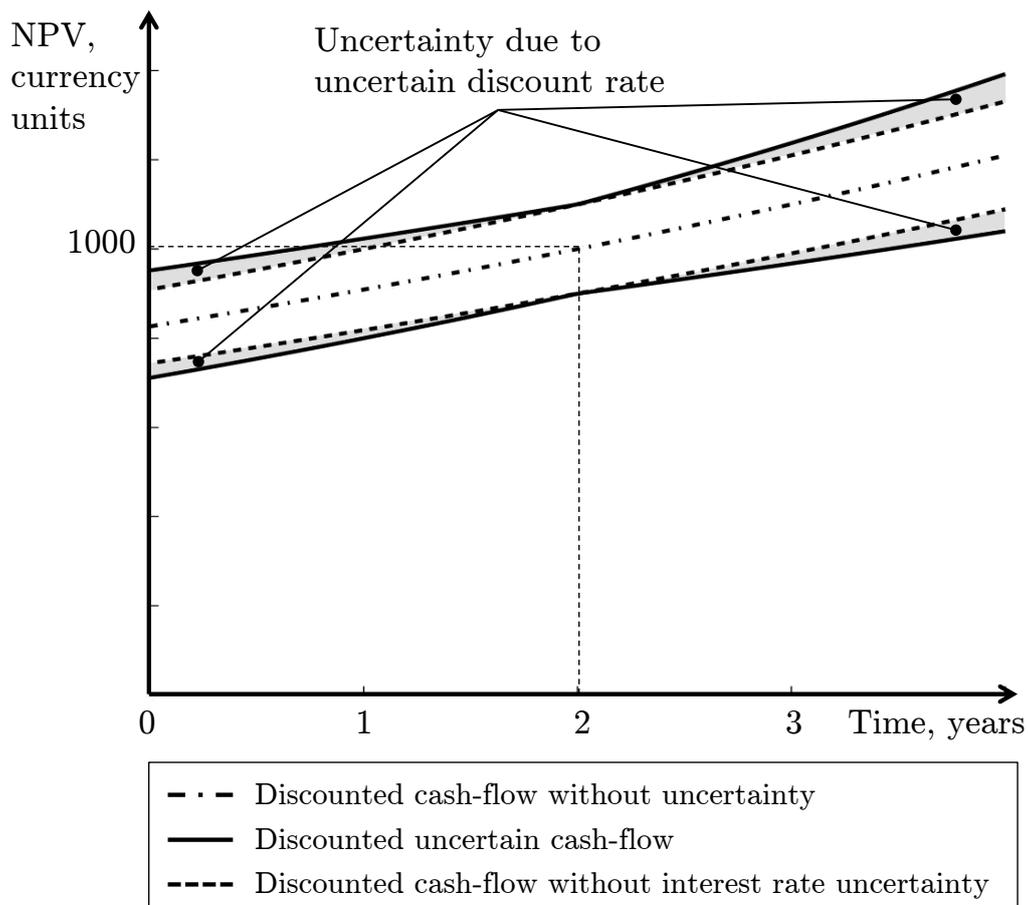


Figure 4.10 Example of the NPV of a discrete cash flow of 1000 currency units occurring at the beginning of the second year.

required activities needs to be found that fulfills causal constraints between the activities and constraints imposed by the availability of resources. Based on this schedule, the activity models must be computed in the order product, process, resource, and cost-pool to calculate the cash flows occurring in a certain time interval.

It would be desirable to find an activity schedule that leads to a cost-optimal execution of all tasks. A cost-optimal production schedule would establish an upper bound on the cost-effectiveness of a certain solution. This challenge is closely related to production scheduling in the production life-cycle phase and optimal project planning in the design, integration, commissioning, and ramp-up phase. Neither in the former nor in the latter domain, the problem is satisfactorily solved in closed-form for planning problems of significant complexity involving

real world constraints. The extension to interval numbers applied here makes it even harder to solve the challenge of finding a globally optimal solution.

An alternative approach to scheduling is modeling the decision-making process typically taking place in project management and production planning. Here, like in real manufacturing, a decision heuristic is established that steers the activity scheduling based on the availability of resources once the state of the automation system changes, for example when a resource becomes available.

It is desirable to keep the computational overhead low and allow for quick responsiveness of the cost-benefit assessment system. Therefore, the manufacturing simulation approach involving a decision heuristic is chosen for the cost model. In order to perform the scheduling, activities are assigned a specific priority depending on their criticality for the project. This criticality describes the importance of the activity for completing the project. Activities that are likely to be on the critical path of the project are assigned a higher priority. For example, the installation of robot hardware has a higher criticality than the interfacing of a sensor as the former would obstruct further work on the hardware of the robot system. In the following, the used approaches for scheduling in the different life-cycle phases are explained in detail.

Scheduling during development, integration, commissioning, and ramp-up

During development, integration, commissioning, and ramp-up, the activities are mostly focused on bringing the robot system into operation. These activities are scheduled by trying to execute activities whose preconditions are fulfilled in the order of their priority while observing constraints imposed by the interdependencies of activities.

Let \mathfrak{A} denote the set of all activities for development and integration containing all integration activities \mathbf{a}_i , i.e. $\mathbf{a}_i \in \mathfrak{A}$. At every point in time t in the life-cycle of the robot system, \mathfrak{A} can be decomposed into the set of executed integration activities \mathfrak{A}_{ex} and not yet executed activities $\mathfrak{A}_{\text{nex}}$. \mathfrak{A}_{ex} is the subset of activities in \mathfrak{A} that

have been completed, while all other activities, i.e. not started or still running activities are contained in $\mathfrak{A}_{\text{nex}}$. All activities \mathfrak{a}_i are linked to a set of precondition activities $\mathfrak{A}_{\text{prec}}^{\mathfrak{a}_i}$. $\mathfrak{A}_{\text{prec}}^{\mathfrak{a}_i}$ being the set of activities that need to be executed before activity \mathfrak{a}_i can be started, i.e. $\mathfrak{A}_{\text{prec}}^{\mathfrak{a}_i} \subset \mathfrak{A}_{\text{ex}}$ is the precondition for starting activity \mathfrak{a}_i . Furthermore, all activities are assigned a priority $\mathfrak{p}^{\mathfrak{a}_i}$ ranging from 1 (low priority) to 3 (high priority).

To determine the execution order of the activities in \mathfrak{A} , the set candidate activities $\mathfrak{A}_{\text{cand}}$ is extracted. $\mathfrak{A}_{\text{cand}}$ is the set of all activities that can be started as their precondition constraints are fulfilled, i.e. $\mathfrak{A}_{\text{cand}} = \{\mathfrak{a}_i \in \mathfrak{A}_{\text{nex}} \mid \mathfrak{A}_{\text{prec}}^{\mathfrak{a}_i} \subset \mathfrak{A}_{\text{ex}}\}$. The candidate activities are sorted into an ordered set \mathfrak{A}_{S} with decreasing priority

$$\mathfrak{A}_{\text{S}} = \begin{pmatrix} \mathfrak{a}_{\text{S}1} \\ \mathfrak{a}_{\text{S}2} \\ \dots \\ \mathfrak{a}_{\text{S}i} \\ \dots \\ \mathfrak{a}_{\text{S}n} \end{pmatrix} = \{\mathfrak{a}_{\text{S}i} \in \mathfrak{A}_{\text{cand}} \mid \mathfrak{p}(\mathfrak{a}_{\text{S}i}) \geq \mathfrak{p}(\mathfrak{a}_{\text{S}i+j}), j > 0\} \quad (4.18)$$

The ordered set of executable activities \mathfrak{A}_{S} is handed over to the scheduler that tries to execute the activities starting from the first element $\mathfrak{a}_{\text{S}1}$, i.e. the element with the highest priority. For each activity in \mathfrak{A}_{S} , it is determined whether all required resources are available. If this is the case, the activity is executed thus blocking the required resources. After this, the scheduler moves on to the next item $\mathfrak{a}_{\text{S}i+1}$ until all elements in the list have been attempted to execute. This procedure ensures that the resources are used as far as possible in parallel and important activities are handled with high priority.

Scheduling during nominal production

During the nominal production, a robot system follows a defined production schedule only interrupted by maintenance and repair. Let p_i be a product manufactured by the robot system and P_{prod} the set of all manufactured products. As

common, the production task is specified by defining the lot size $s_L^{p_i}$ for each product $p_i \in P_{\text{prod}}$. Furthermore, the relative production volume r^{p_i} of product p_i is specified by the user, i.e. how many lots of this product are produced relative to the number of lots of other products. From this information, the scheduler derives a production schedule as follows.

Let r^{LCM} be the common denominator of the relative production volumes r^{p_i} . Then the scheduler generates one production cycle by iterating the lot index $i_L \in [1, \dots, r^{\text{LCM}}]$. The lot production activity $\mathbf{a}_{L^{p_i}}$ of a specific product p_i is inserted into the production schedule if $(i_L \bmod \frac{r^{\text{LCM}}}{r^{p_i}}) = 0$. The lots are always inserted with the predecessor lot as a precondition, i.e. $\mathbf{a}_{L^{i-1}} \in \mathfrak{A}_{\text{prec}}^{\mathbf{a}_{L^i}}$. The lot production activity then triggers the single contained production activities and delegate their execution to the contained activities, i.e. the production of single product instances. This means a lot production activity $\mathbf{a}_{L^{p_i}}$ of product p_i is an ordered set that looks like follows:

$$\mathbf{a}_{L^{p_i}} = \left(\begin{array}{l} \mathbf{a}_{p_{i1}}, \text{ where } \mathfrak{A}_{\text{prec}}^{\mathbf{a}_{p_{i1}}} = [] \\ \mathbf{a}_{p_{i2}}, \text{ where } \mathfrak{A}_{\text{prec}}^{\mathbf{a}_{p_{i2}}} = [\mathbf{a}_{p_{i1}}] \\ \dots \\ \mathbf{a}_{p_{in}}, \text{ where } \mathfrak{A}_{\text{prec}}^{\mathbf{a}_{p_{in}}} = [\mathbf{a}_{p_{in-1}}] \end{array} \right). \quad (4.19)$$

The details of this delegation mechanism are explained in Chapter 6.

4.3 Knowledge management of cost data

In order to set-up the cost model for a specific robot system, knowledge is required for determining the required activities and subsequently selecting, instantiating, connecting, and parameterizing the appropriate activity models for these activities. The required knowledge consists of different types of information from different domains and is typically supplied by different stakeholders in different formats. Examples are given in the following (see also Figure 4.11).

Knowledge on the specific set-up of the robot system that is only applicable to the specific robot cell, i.e. which components are included and how they are connected. The AutomationML file from plant engineering specifies this information.

General information on the used components that are applicable to all installations, for example component prices and performance characteristics for specific components of which robot systems are built. This information can be extracted from purchase offers and data sheets of these components.

Information on the engineering and integration process of the robot system that relates to know-how, efforts, and costs that are specific for the integrator of the robot system. Sometimes, this information can be retrieved from project post calculations or the ERP-system of the system integrator. Sometimes, this information is only implicitly available.

Cost information relevant for the use of the robot system which is specific for the end-user, for example manufacturing volumes and production specific cost items, such as hourly cost of the manufacturing personnel.

Figure 4.11 gives examples of different knowledge categories and specifies whether they are specific for individual robot systems, i.e. only applicable to the robot system for which they have been collected, or transferable to other installations. Furthermore, it is indicated which involved stakeholder can supply the required knowledge or if the knowledge is general for the entire domain of industrial robotics.

This section introduces the handling of knowledge in the cost-benefit model. A knowledge base forms the foundation for the knowledge handling. The different components of this knowledge base are introduced in detail below. First, the section explores the handling and processing of data that is specific for the analyzed robot system. It then introduces an ontology that provides semantic grounding to the cost benefit model and allows completing missing information through reasoning. Finally, it explains the handling of uncertainties, i.e. how a lack of knowledge is addressed.

	General	Component supplier	System integrator	End-user
Knowledge that is specific to individual robot systems			System setup (product model during system design)	Product models (production)
				Production schedule
				Benchmarking costs
Knowledge that is transferrable between different robot systems	Semantic grounding	Component models	Integration activity models	Personnel cost in production
	Specification of data models	Service cost and technician cost	Personnel cost in integration	Capital budgeting approach
			System performance	Purchasing overhead

Figure 4.11 Different domains of information required for the cost model.

4.3.1 Machine-specific knowledge

Input information for the cost-benefit model that is specific for the analyzed robot system is stored in the AutomationML file format. This allows easy integration with design tools for the engineering of the robot system, such as PLM software for cell design and simulation. The AutomationML file contains the available information of the set-up of the robot cell, that has been specified by the system integrator during the design process up to the point of the cost-benefit analysis. The instance hierarchy of the AutomationML file defines components that are contained in the robot system. The roles of these components in the AutomationML file provide semantic meaning. They allow to relate components in the AutomationML cell description to components in the RoboCost ontology. The roles are part of the RoboCost ontology introduced in the following section and can be exported to AutomationML to be used in plant engineering. Furthermore, the AutomationML file contains information on the manufactured products in terms of required features to be manufactured by the robot system. This information is supplied by the end-user but typically transferred to the system integrator during the design phase.

To ease interpretation and integration with other data sources, the AutomationML file is parsed and transformed into RDF format before being processed by the cost-benefit assessment tool. The RDF description allows easier retrieval of data and matching of the system set-up to component properties stored in the knowledge base. This comprises in particular the retrieval of missing data as described in Section 4.3.3. Therefore, even though AutomationML is used as exchange format for integration with engineering tools, the knowledge processing itself is done in RDF. In this sense, RDF is a solution to perform advanced information processing based on AutomationML.

In order to transform the AutomationML description into RDF, the approach proposed by Runde et al., 2009 is employed. The AutomationML file is mapped to an AutomationML base ontology which acts as semantic grounding for the RDF version of the AutomationML file. The transformation is carried out in several steps. First, the internal interface class library of the AutomationML file is parsed as this library does not reference other elements in the AutomationML file. Second, the role class library is parsed. Here, references to interfaces are resolved during the parsing process. The third step is the parsing of the system unit class library. Also here, references to the interface class library are carried over to RDF in the parsing process. Internal links are parsed and stored, but not yet resolved as it cannot be guaranteed that the referenced element has already been parsed. Subsequently, the instance hierarchy, which contains the description of the actual robot system, is processed. All parsing algorithms work recursively and move on to the contained elements of each element before returning. Concluding the parsing process, internal links of the AutomationML file are resolved. The parsing is implemented hard-coded and not in a declarative way. Hence, a change in the AutomationML description would also require an adaptation of the source code used for transforming the AutomationML description into RDF.

The RDF description of the robot system is used to instantiate the cost-benefit model. The AutomationML and RDF descriptions as well as the resulting cost-benefit model are specific to the analyzed robot system. In the process of creating the cost-benefit model, information from the knowledge base that is not specific

to one particular robot system is integrated. This information is stored in a cost ontology.

4.3.2 Cost ontology for machine independent knowledge

The core problem of cost-benefit analysis of robot systems lies in the distribution of knowledge on system cost among the different stakeholders. Manual integration of all this knowledge into the machine-specific design documentation is tedious and usually not carried out. In order to automate this process, a formalized, machine readable description of machine independent knowledge is required. This knowledge should then be available to all system designers to perform cost analyses. For this purpose, a knowledge base capturing machine independent knowledge (see Figure 4.11 for machine-independent knowledge aspects) is set-up.

The knowledge base contains information which is applicable to different domains of industrial robotics as is reflected in the components of the knowledge base outlined in the following.

Semantic grounding of cost-relevant terms: The knowledge base contains the so called RoboCost ontology, that provides semantic grounding for the cost model and its components. The RoboCost ontology also includes the role library used in the AutomationML file.

Component database: The knowledge base contains information on robotic components, their properties, and activities required during the system set-up and operation, e.g. price, roles, delivery time, technical specification.

Models for integration activities: Activity models that describe cost-relevant aspects of integration activities together with meta-information on how to use these models are contained in the knowledge base.

Data model of AutomationML: The knowledge base further contains a description of the data model of AutomationML allowing the transformation of the AutomationML to RDF as described above.

The RoboCost ontology provides the semantic grounding of the entire cost-benefit model and allows reasoning during the set-up phase of the cost model. Further, the semantic grounding allows combining knowledge from different sources. It is the core component for creating the cost model for a specific robot system from the associated AutomationML description and resolving uncertainties and reasoning on missing knowledge in this process.

The RoboCost ontology is a domain ontology that defines concepts and their interrelation for industrial robot systems. It contains the high-level classes outlined in the following. Figure 4.12 shows how these top-level classes of the RoboCost ontology are interrelated.

Automation company: Companies that sell or purchase components, robot systems or services in the automation market.

Automation resource: Personnel or hardware devices that are used as resources in order to build or operate robot systems.

Automation role: Definition of roles that can be taken by automation resources. While the resource itself describes what type of personnel or hardware is used, the role describes how it is used in a particular robot system. An exported version of the role library is also used in the AutomationML description of the system as described above.

Automation task: Activities related to building, maintaining, or operating robot systems.

Models: Activity models that describe cost-relevant aspects of automation tasks.

Model IO: Input and output quantities of models and their relation to each other.

Model parameters and properties: Values for parameters and properties of models and automation resources³.

³Please note that this definition does not correspond to the definition of parameters and properties in RDF.

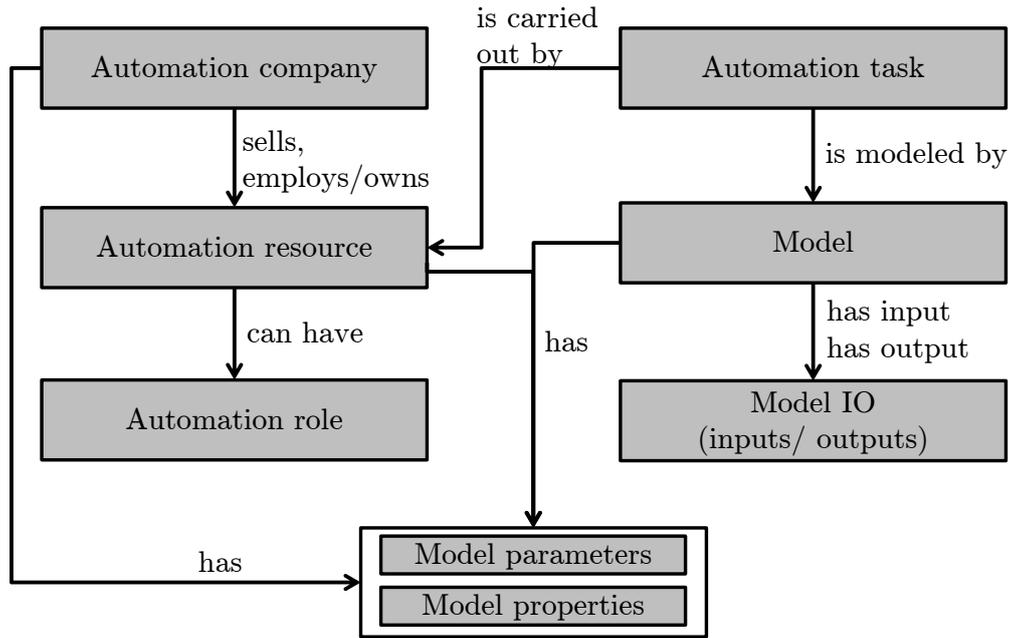


Figure 4.12 Relation of top-level classes of the RoboCost ontology.

All entities in the RoboCost ontology are referenced through URIs in order to unambiguously refer to concepts in the ontology. Furthermore, an ontology element can contain different human readable names for the handling of association uncertainties as explained in detail in Section 4.3.3. This allows mapping different real world descriptions that might refer to the same component to an unambiguous URI by performing fuzzy reasoning from the real world name to the different name properties associated with the knowledge base. Through this mechanism it is possible to match real world objects to a unique URI without the need for strict naming conventions.

The top-level classes of the RoboCost ontology and the entities that belong to these classes are described in detail in the following.

Automation company

For automation companies, the following sub-classes depending on the role in the market exist.

Component manufacturer: manufacturer or distributor of automation components such as sensors, drives, etc.

Robot manufacturer: manufacturer of industrial robots. Due to the central importance of robots, a separate class is introduced, even though robot manufacturers are strictly speaking also component manufacturers.

Software vendor: producer or distributor of software for use in or with robot systems, e.g. offline programming systems.

System integrator: company that integrates automation components into a machine.

Industrial service provider: company that provides services around industrial robot systems, e.g. maintenance.

End-user: company that uses robot systems in their production.

One company may belong to several of these sub-classes. Each company is classified as being a small, a medium, or a large enterprise. Furthermore, specific accounting properties for the cash pool can be stored with the company data, in particular overhead costs for personnel and purchasing as well as interest rates for capital.

Automation resource and component database

The top-level class automation resource contains information on the means (i.e. resources) required for the execution of integration and manufacturing activities. All automation resources have at least one role that they support. Figure 4.13 shows the sub-class structure for automation resources. The top-level distinction is between human resources (automation personnel), technical resources used for production or integration, but not being part of the robot system (technical resource) and technical components integrated into the robot system (automation system component). The human resources always have a connection to the company by which they are employed. Automation system component is the class

to which all technical components (for example robots, sensors, etc.) of automation systems belong. This class is split up into sub-classes to specify the types of components in more detail. The first distinction is between software and hardware components. This distinction is useful as the pricing model for hardware (purchase-based) and software (license-based) is usually different.

The members of the automation component class hierarchy constitute a component database that is used for obtaining information for building the cost-benefit model (see Section 4.3.3 for more detail). The automation component class hierarchy defines required parameters and properties of the members of these classes to which the inserted components need to comply. For example, all hardware components require a price, a weight parameter and an IP-code property.

The automation personnel sub-class lists the human resources used in the life-cycle of the robot system. This might refer to actual human resources in terms of individual employees, raising data protection issues, generic human resource types employed at a specific company, e.g. electrician of system integrator A, or generic human resources in general, e.g. electrician.

The used resources in the design process (technical resource) are numerous, for example CAD-station, lifting device, laser scanner for system referencing. The sub-class has no dedicated structure, but typical resources are stored there directly as instances.

Automation role

The roles in the RoboCost ontology comprise the roles which are contained in the developed role library for AutomationML (see previous section). When compared to the AutomationML role library, the ontology defines additional roles related to companies and personnel.

It is important to distinguish between the entity (automation resource) and its role (automation role). These two concepts are closely connected as the type of the automation component often defines its role. Nevertheless, they are not the same

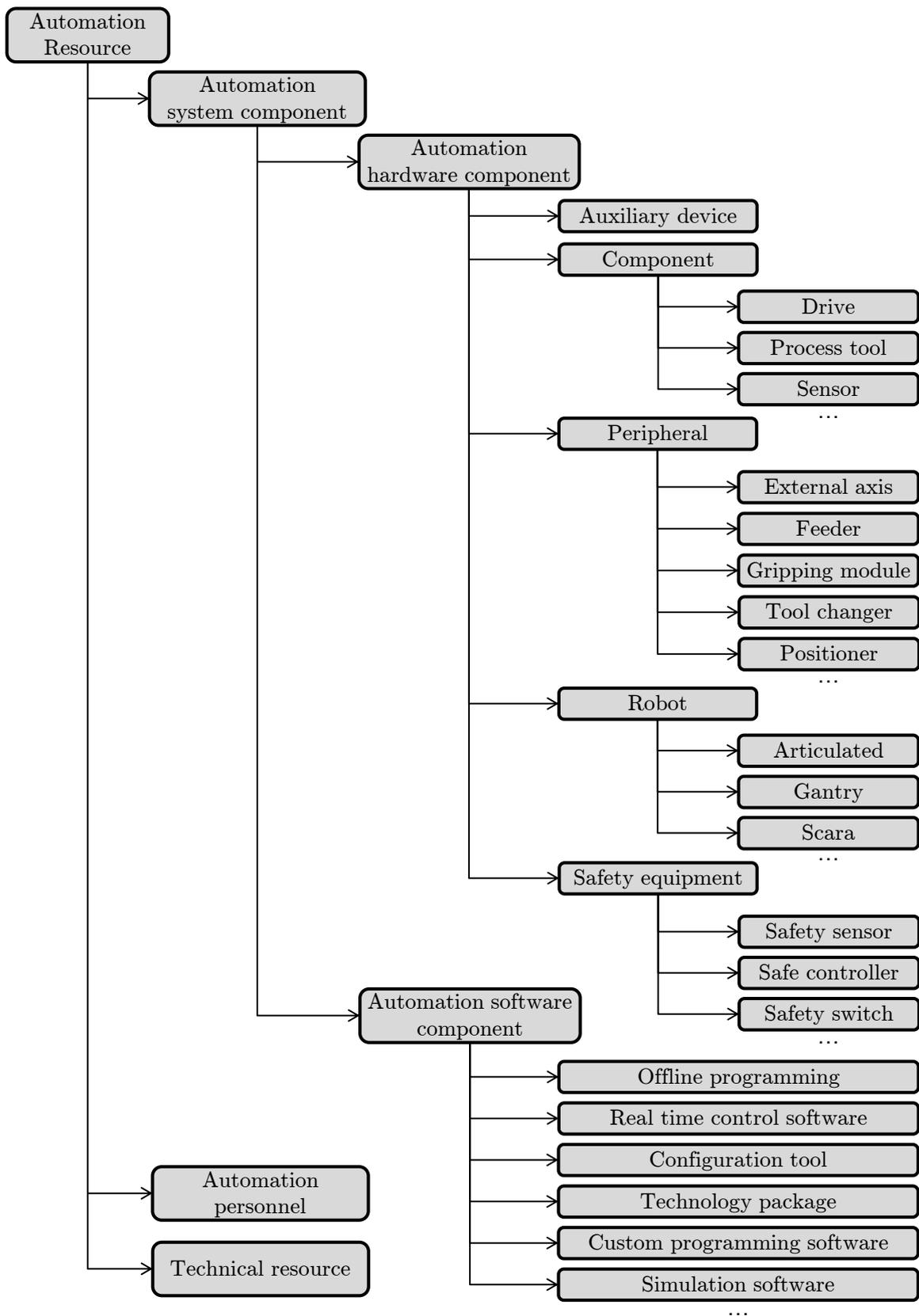


Figure 4.13 Resource classes of the RoboCost ontology.

as one component may play different roles in different manufacturing systems. For example, a triangulation sensor might be used for work piece localization, quality control, online seam tracking, and potentially more roles. While some parameters and properties of a component depend on the component, others relate to the role of the component in the automation system. Examples for the former are the purchase price and technical properties of a component. An example for the latter is the integration effort of a sensor, because this effort depends on whether the sensor data is used in real-time or offline. It is important to distinguish role and component in order to be able to capture these nuances in the cost model.

Automation task

The class hierarchy for automation tasks is the key entry point for the activity-based cost model. Here, cost-causing activities are described.

Automation components, their roles, product features, or other tasks can link to automation tasks and hereby specify that their inclusion in a robot system requires the execution of the linked automation tasks. Figure 4.14 shows the modeling of this interdependence. Using this information, the cost-benefit model for integration and operation of the robot system is built by instantiating the required activities.

The automation task description comprises the following information:

- The role of the resource carrying out the activity.
- The company that is responsible for the task, that has to carry the costs that are generated in the cost-pool due to the activity.
- The model that can be used to describe the cost-relevant aspects of the task (see below).
- Optionally, the task can list precondition activities that need to be carried out before the task can be executed.

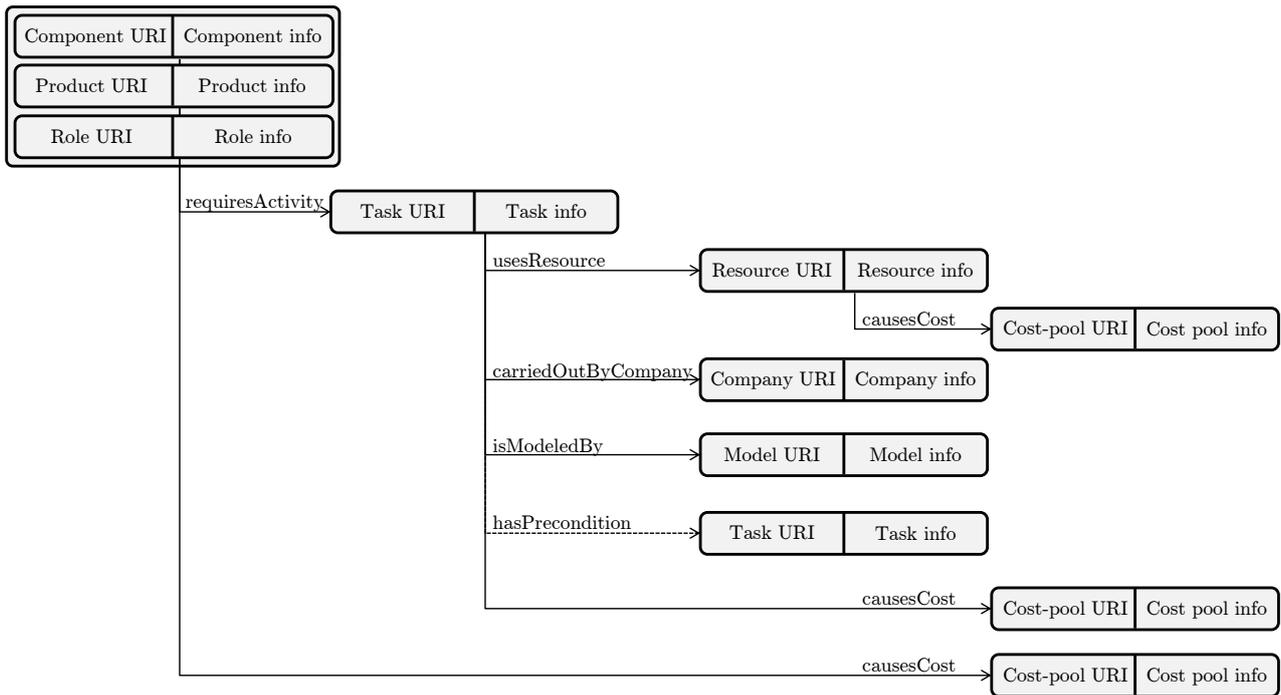


Figure 4.14 Information related to members of the automation task class.

During the set-up of the cost model, the required tasks are identified through product features, components of the robot systems, component roles, or due to other tasks that require preconditions. Then, the activity model related to the task is instantiated and parameterized through the information of resource and company.

Models

The members of the model class link to a mathematical description of the cost model and to a semantic description of inputs (cost drivers) and outputs (activity levels). This allows to connect the models to other activity models contained in the cost-benefit model. Figure 4.15 shows the minimal information attached to each model instance in the knowledge base. A data property contains information where the mathematical model can be found. Object properties link to the input types and output types of the model.

The mathematical description of the model implements a non-differentiating func-

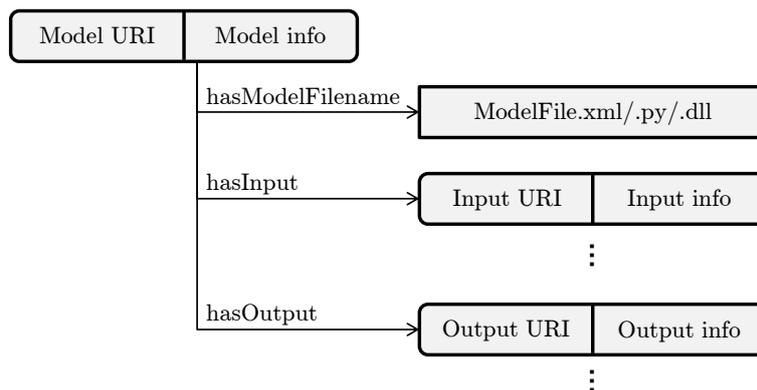


Figure 4.15 Information related to members of the model class.

tion that transforms inputs into outputs according to equation (4.1) on page 105. This relation is typically an algebraic function, a look-up table, or a combination of both. During instantiation of the cost-benefit model, this description is parsed and an executable form of the model is generated as shown in Figure 4.16. This form needs to be able to handle interval computations.

In order to store the activity models in a database and load them dynamically into the cost-benefit model, a serialization of the models is required. Three different ways for specifying model functions in the knowledge base have been devised. The first way is a specialized XML-dialect. The second option to specify model functions is the implementation through Python scripts. As a third option managed code DLL libraries can be imported. The different ways of including cost models are discussed in detail in the realization chapter (see Section 6.1.1). The model functions are serialized in a file which is referenced in the knowledge base with additional model information (see Figure 4.15).

Model IO

As outlined, the models contain information on inputs and outputs that are used for computation. The members of the class Model IO are the vocabulary used for inputs and outputs and allows the matching of inputs and outputs from different PPRC-levels of the cost model. This aspect is important in order to connect the single activity models to each other to form an integrated cost model for

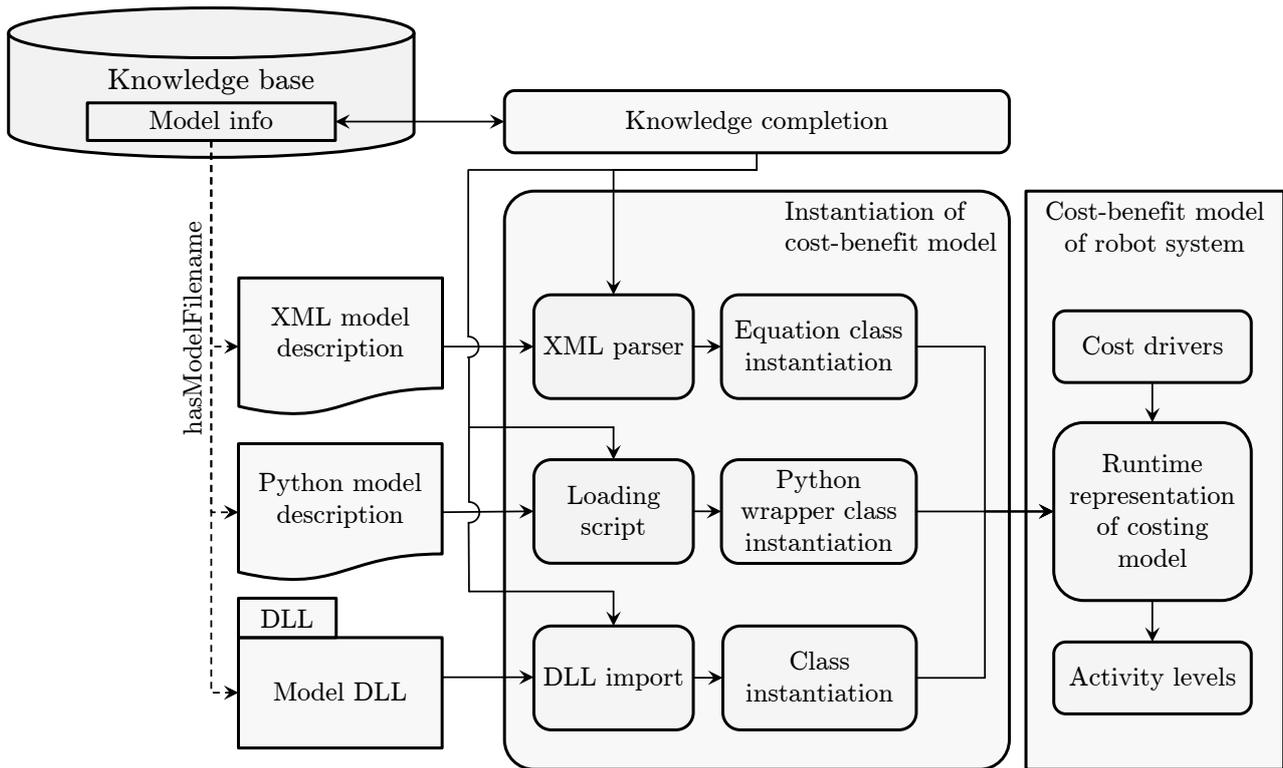


Figure 4.16 Instantiation of activity model from serialized form.

the robot system. The members of this class hence provide semantic grounding to the information that is exchanged between the different PPRC-levels in the cost-benefit model.

Model parameters and properties

The specific characteristics of models, components and other entities of the RoboCost ontology need to be described in detail for building a cost-benefit model. For this purpose, the concept of parameters and properties is introduced in the ontology.

Parameters define actual numerical values referring to performance characteristics or economical information of other entities in the ontology. Model parameters are stored as members of a separate top-level class structure in the RoboCost ontology. Parameters have two data properties relating to the infimum and supremum of the

parameter. Furthermore, they are assigned a unit of measurement. Components or other entities link to parameters through the object property *hasParameter*.

Properties refer to characteristics that can take one particular value from a set of possible values. An example is an interface or the IP-protection class (International Protection Marking) of a component. The interface can be selected from a range of possible interfaces, such as for example TS232 and Canbus, while the IP Code can take discrete values such as for example IP65 and IP67. A separate top-level property class hierarchy contains the sub-classes relating to the properties. These classes are populated with the sets of potential values for the respective property. Entities in the ontology can refer to these values through the *hasProperty* object property. Through this structure, the property itself can be retrieved by querying the type of a certain property value.

Domain ontology for AutomationML file format

As already outlined, the information generated by plant engineering tools is used in the cost-benefit model through input from the AutomationML file format. As already explained, this information is transformed into RDF for further processing. For this purpose, an AutomationML domain ontology is developed that defines the relevant concepts of AutomationML and gives semantic grounding to the AutomationML description after translation into RDF. This AutomationML base ontology describes the characteristic concepts of AutomationML, such as for example the instance hierarchy, system unit class library and role class library. These concepts are described in the form of classes that allow to generate an RDF version of a specific robot system from its AutomationML description by creating entities of the XML-nodes in the AutomationML file and assigning them to the specific classes. The AutomationML base ontology is contained in the RoboCost Ontology to provide semantic grounding to the RDF version of the AutomationML file (see Figure 4.7).

4.3.3 Handling uncertainties in knowledge management

The cost-benefit model distinguishes between the machine-specific knowledge for a certain robot system and the machine-independent knowledge concerning the entire domain of industrial robotics. The machine-specific knowledge is contained in the associated AutomationML file, while machine independent knowledge is stored in the knowledge base. The key task for cost-benefit modeling is matching the machine-specific description of the robot system to the machine-independent knowledge in the knowledge base. This matching allows for reasoning on information that is missing in the machine-specific description of the robot system. This is required, as the plant model does not contain the information required to instantiate a full cost model or in early design stages not even the information on the complete system set-up.

The matching process requires that URIs from the knowledge base are set into relation to URIs in the AutomationML data of the robot system. If this is not possible, for example because a relevant component is missing in the knowledge base or the relation cannot be established, other means for estimating the required information are necessary.

Subsequently, it is outlined how the different types of uncertainties (see Section 4.2.5) are modeled and handled in the cost model. It should be noted here that resolution of uncertainties in the developed cost model is always situation-based. Uncertainties are resolved only if an underlying relation, parameter, or property is required for the cost computation.

The matching process is depicted schematically in Figure 4.17. The process starts with a potentially incomplete AutomationML description of the system that has been transformed into RDF. The first step is the set-up of the complete bill of material of the robot system, i.e. the resolution of its composition uncertainty. The components specified in the AutomationML file are broken down into their sub-components until the individual components that can be purchased by components suppliers or need to be manufactured are identified. These components

are then matched to elements in the knowledge base, i.e. the association uncertainties between the elements of the AutomationML information and the knowledge base are resolved. Once the components are known, required activities and related activity models can be retrieved. After this, the parameters and properties required for the activity models are retrieved from the knowledge base. At the beginning of this process, uncertainties are mostly epistemic in nature as the uncertainties refer mostly to relations between the device description and the knowledge base. During the matching process, aleatory uncertainties gain importance as the parameters on the device level are increasingly driven by random processes and chance.

Figure 4.18 provides details on the flow of the matching process. The figure also introduces the main procedures for uncertainty resolution which are explained in detail below. In the context of the figure, the starting point of every process of uncertainty resolution is a parameter value uncertainty in the cost-benefit model. This parameter value uncertainty is then reflected on the knowledge base and might lead to composition, association, and parameter value uncertainties on the side of the knowledge base which are then resolved consecutively as they occur.

In the first step, composition uncertainties are resolved in case the component under scrutiny is a system consisting of separate components. An example is the purchase cost of a robot end-effector which might be composed of several components such as for example gripper modules, standardized profiles, sensors and bus modules. Before the price can be determined, the single sub-modules need to be identified in case they are not specified. Subsequently, components in the knowledge base from which the desired properties can be retrieved need to be found. This relates to the resolution of association uncertainties. Finally, once component matches in the knowledge base have been found, the desired parameter needs to be estimated and the associated parameter value uncertainty needs to be resolved. In the following, the single uncertainty resolution steps are described in detail.

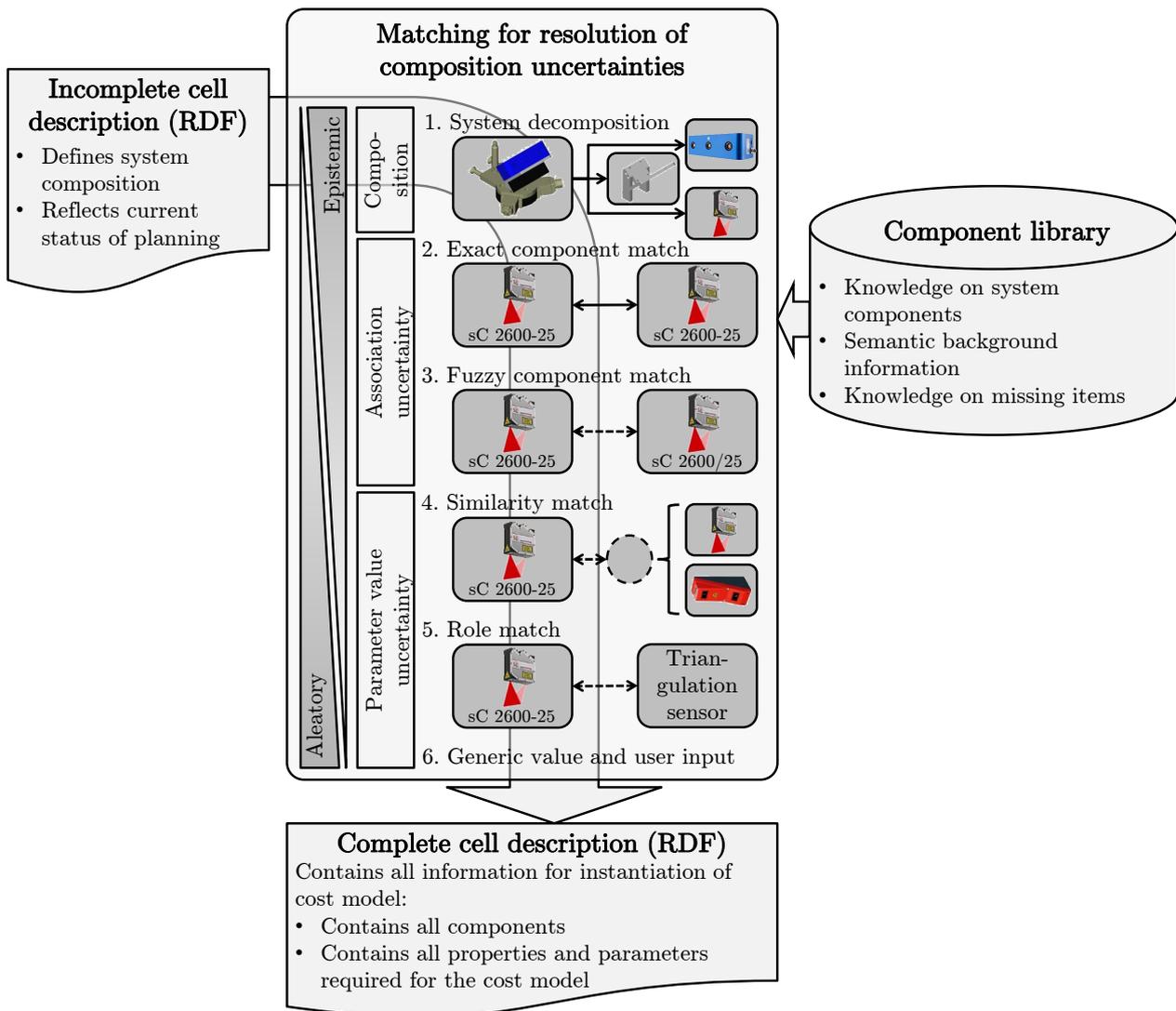


Figure 4.17 Matching process between system composition and knowledge base in order to resolve uncertainties.

Handling of composition uncertainties

Composition uncertainties occur when detailed information on the composition of the robot system or major sub-components is missing. For example, this is the case if major modules of the robot system have been created in the PLM toolchain, but not yet designed in detail. The cost-benefit model has two strategies to handle this type of uncertainty described in the following.

1. Estimation through generic parameters: Generic parameters define component or sub-system properties solely depending on the role of the component or

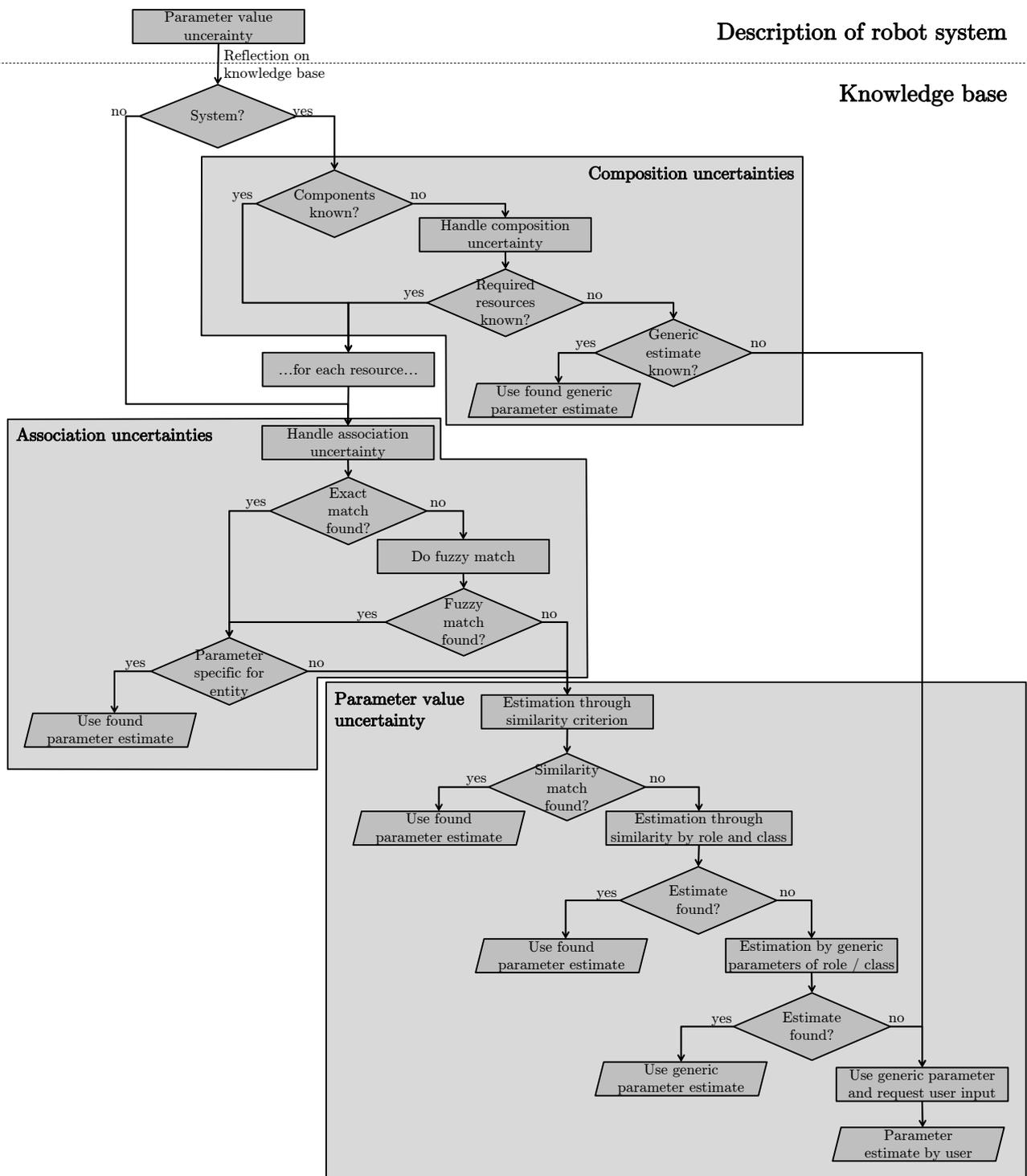


Figure 4.18 Flow chart depicting the handling of uncertainties in the cost model.

sub-system without knowledge on the implementation details or exact component type. In the knowledge base, generic parameters are attached to the component role. If defined they can be used as a very coarse estimate. For example, it is possible to define a price range for a mechanical gripper for grasping single small components without knowing details about the nature of the part or the exact design of the gripper.

2. Definition of required sub-component roles: Instead of defining a lump-sum estimate for the sub-system as a whole in the form of a generic parameter, the system role can list required resource roles for the sub-system. For example, a gripping module for small parts can be decomposed into a mechanical gripping module, jaws, a robot adapter structure, and a control interface. For these components, the required parameters can be determined using the mechanism for resolution of association uncertainties (see next section). This type of estimation is more elaborate than lump-sum estimation, as the parameter retrieval for sub-components takes into account parameters of the overall system, e.g. the weight of the part for estimating the price of the gripper module.

With the first strategy, the composition uncertainty is circumnavigated by estimating the parameter through a lump-sum estimate. With the latter strategy, the composition uncertainty is decomposed into several association uncertainties which subsequently have to be addressed. In general, the cost-benefit model first tries to resolve the composition uncertainty, as in this process more details about the system become known. For example, additional required activities are usually found when decomposing the system to a finer level of detail. Only if the composition uncertainty cannot be resolved, for example because the required role information is missing or because a decomposition in general terms is not possible, the resolution through a generic estimate is attempted.

Handling of association uncertainties

Association uncertainties occur when a device or component in the definition of the robot system cannot be matched to a data entry in the knowledge base. This can occur because the respective entry is missing or because the definitions in knowledge base and system specification mismatch. Association uncertainties are usually related to parameter value uncertainties. This is the case because missing parameter values often reveal association uncertainties and because the resolution of association uncertainties often leads to parameter value uncertainties when a clear one to one relation with the knowledge base cannot be established. To resolve association uncertainties, a matching between the properties of the local entries and the database is performed. The following steps are performed hierarchically in order to handle association uncertainties:

1. Exact match: The cost model retrieves the value of the requested parameter by directly matching with an object in the data base and retrieving the associated parameter of the object. This match either directly addresses the requested entity through its URI or through one of the associated human readable names in terms of the *hasName* data property. This case corresponds an association uncertainty that can unambiguously be resolved by querying the knowledge base.
2. In case no direct match for the component can be found, a fuzzy match regarding the name and vendor property is applied. This means, an appropriate match for the human readable name is searched. In this case, a set of matching components might be found whose properties might need to be combined resulting in a parameter variability uncertainty.
3. If also a fuzzy match cannot be established, the resolution of the association uncertainty through matching with existing data base entries is abandoned and the underlying parameter value uncertainty is resolved. This resolution process is described in more detail in the following section.

The result of the resolution of the association uncertainty is either an unambiguous or ambiguous match between the component under scrutiny and entries of the knowledge base. In case no match is identified, the initial parameter value uncertainty on the side of the cost-benefit model becomes a parameter value uncertainty on the level of the knowledge base as no direct answer can be supplied by the knowledge base. This means the parameter value needs to be estimated from other information stored in the knowledge base as described in the following.

Resolution of parameter value uncertainties

Parameter value uncertainties occur on two levels. Firstly, in the description of the robot system (see Figure 4.18) which triggers the uncertainty resolution process through the knowledge base. Secondly, parameter value uncertainties can occur in the knowledge base, if the knowledge base entry of an entity does not link the desired parameter or if no knowledge base entry could be identified. In the latter case, the cost-benefit model tries to estimate the parameter value through the following steps executed hierarchically until an estimate has been found:

1. Estimation by similarity: The cost-benefit model tries to estimate the parameter value based on similar components by using a similarity criterion defined in the ontology. The similarity criterion is attached to the role of the component and contains knowledge by which quantity a specific parameter is scaled for components of this role. An example for a similarity estimation is the estimation of robot prices based on robot payload. The result of this estimation is a set of estimates of the uncertain parameter forming an interval number containing minimum and maximum values. Hence, the uncertainty of this estimation process is reflected as parameter variability uncertainty in the output and handled as described in the following section. More details on estimation by similarity and the underlying data structures is given below.
2. Similarity estimation by role and class: If no estimate from comparable components based on a similarity criterion can be found, the estimation process falls back to an estimation by components of the same role and class. When

the class criterion is applied, the tree of classes is searched upwards recursively from the direct mother class of the component to the root node and the search is aborted as soon as a result has been obtained. This ensures that the most specific estimate is used.

- a) Estimation by class and role: In a first step, the parameter is estimated from components of the same class and role. Examples for this type of estimation is the price range for articulated welding robots or the price range for triangulation sensors for object detection.
 - b) Estimation by role: If the estimation by class and role fails, the system falls back to estimation by role alone. For example, the price range of welding robots and object detection sensors are estimated. As the class criterion from the previous step is continuously loosened, this corresponds to the final search step related to class and role, i.e. when the class is only restricted to the root note.
 - c) Estimation by class: If also the estimation by role alone fails as no parameter information is contained, only the class of the item is used for estimation. This option is a last fallback that still uses specific information on the actually contained device. This type of estimation is less specific than the match by role as it contains no information of the usage of the respective component and hence also its requirements. Examples for this type of estimation is a general price range for industrial triangulation sensors or a general price range for industrial robots.
3. Generic estimation by role and class: If no estimate based on comparable components can be found, it is attempted to find a parameter estimate from generic properties and parameters linked to the role and class of the item. These are values that are rough estimates, usually with a high level of uncertainty. Again, the estimation by role takes precedence as it is more specific with respect to the application.
 4. Use of generic parameter: In case the relevant parameter cannot be estimated based on the role of the item, generic parameters that conform to the desired

parameter class are used. For this estimate, no relation to component or role exists. The estimates through generic parameters are therefore very coarse and are only used as a suggestion for the next estimation stage, the request of user input.

5. Request of user input: The cost model requests input from the software user, if no estimate or only a generic estimate are found.

The most important step is the estimation by similarity. This process is explained in more depth here. The estimate by similarity starts by querying the role of the desired component in order to retrieve the similarity criterion for devices with this role. Similarity criteria can be defined generally for all parameters of a role or only for specific parameters of the role. In the latter case, the similarity criterion is further linked to the parameter class as shown in Figure 4.19. Parameter specific similarity criteria take precedence over general similarity criteria. Once the applicable similarity criterion is defined, the component knowledge base is queried for components that fulfill the following criteria:

- the component can take the desired role,
- the component can supply a value for the desired parameter,
- the parameter constituting the similarity criterion of the role is defined, and
- the value of the similarity criterion is within the absolute or relative similarity threshold defined in the criterion (default is a relative threshold of 10% which is used if no threshold value is given).

The result is a set of potential parameter values that are handled as a parameter variability uncertainty (see below). The same applies for the estimation through role. Here, estimates are composed of components that can take the desired role and supply a value for the desired parameter. This estimate however is much less specific than estimation by similarity.

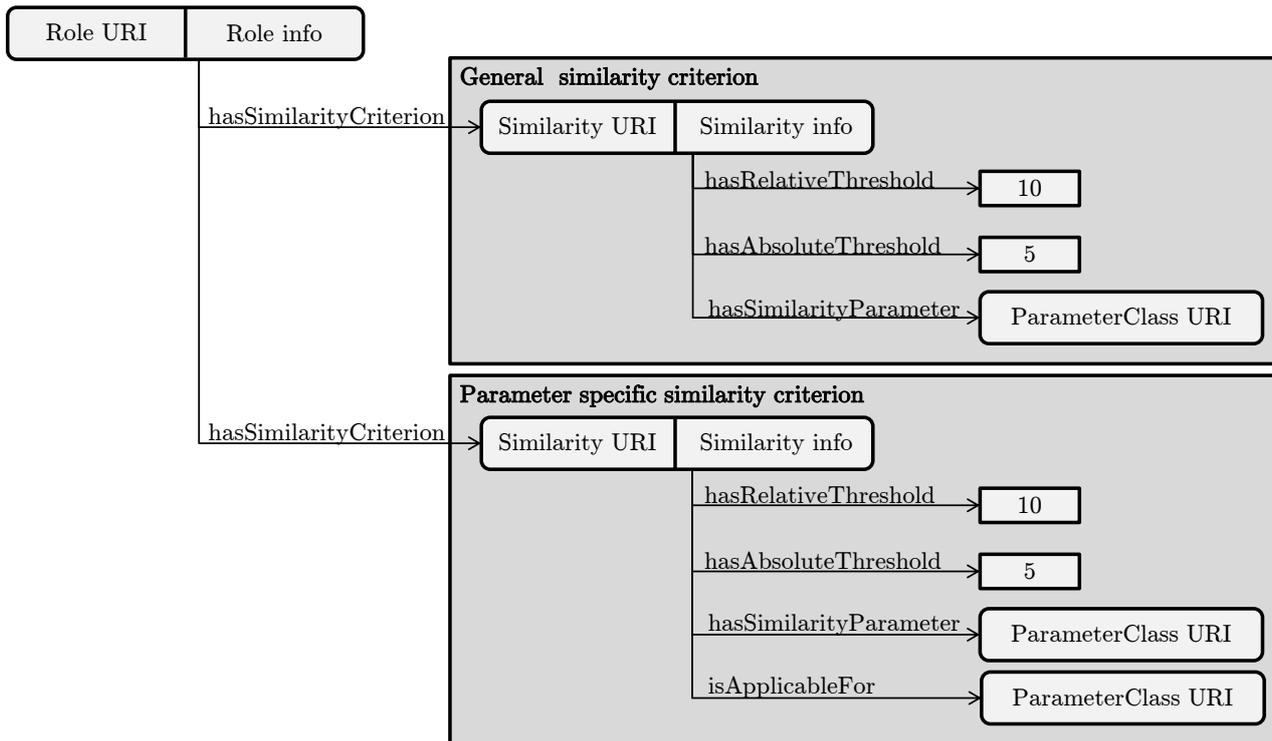


Figure 4.19 Specification of the similarity criterion in the knowledge base.

Resolution of parameter variability uncertainties

Parameter value uncertainties refer to the situation of not knowing the value for a parameter, i.e. completely missing knowledge. Parameter value uncertainties are resolved by estimating the parameter value (see above). This often leads to an uncertainty regarding the exact parameter value. This parameter variability uncertainty can have different sources:

- estimation of upper and lower bound by experts,
- ambiguous match between a real world component and entries in the knowledge base,
- estimation of a parameter value depending on similar components (similarity match),
- conflicting information on a parameter value, usually from different sources, for example two retailers with different prices,

- uncertainty due to inherent randomness of the quantity, for instance volatility of the electricity price due to market situation,
- underlying cause-effect relation that is too complex to monitor, e.g. price formation due to target market and sales volume of customer.

All these types of parameter variability uncertainties are treated in the cost-benefit model through the use of interval numbers. The aim here is to reduce parameter variability uncertainties as far as possible. Due to the underlying nature of the uncertainty and economic constraints regarding the effort for uncertainty reduction, a complete elimination of the uncertainty is usually not possible. Therefore, the remaining level of uncertainty is taken into account in the model. The treatment of parameter variability uncertainties through interval numbers allows propagating the uncertainty through the model and shows the effect on the result.

For the reduction of the inherent level of uncertainty, the model relies on the proper selection and maintenance of information in the used data sources, mostly the knowledge base. The process on how conflicting parameter information is consolidated into a single parameter estimate is described in the following.

The result of the estimation process triggered through parameter value uncertainties are often conflicting parameter values. This situation occurs when the data acquisition process results in a set of n parameter value estimates x_1, \dots, x_n , where $x_i \in \mathbb{I}$, for the parameter \mathcal{P} . These estimates have to be consolidated in a parameter estimate $x_c \in \mathbb{I}$ that can be used in the cost model. The applied procedure is the determination of the interval boundaries containing all available estimates, i.e. the set union $x_c = \cup_{i=1}^n x_i$, computed by $x_c = [\min(\inf(x_i)), \max(\sup(x_i))]$. This procedure guarantees that if the real value of the parameter lies inside one of the bounds of the parameters' estimates also the results will lie within the computed result. This procedure is well-suited for small sets of estimates with a high level of trust in the individual estimates. An assumption that is considered applicable for the type of information that is stored in the knowledge base.

Resolution of time uncertainties

Time uncertainties are caused by the parameter variability uncertainties of the related models leading to uncertain execution times for processes on the resource level. For example a parameter referring to the required work time for an installation task will create a time uncertainty in the resource model of the worker carrying out the installation task that needs to be handled in the model computation. The handling of the time uncertainties for scheduling in the model has been described in Section 4.2.6. Additionally to the temporal sequence of events, time uncertainties are propagated through the cost model into the cost-pool where they typically lead to uncertainties in cost as many cost aspects are time dependent.

4.4 Conclusion of concept

The developed concept allows to integrate knowledge from different sources into a consolidated cost-benefit model of the robot system. The integrated knowledge is coming from the AutomationML description of the robot system, general information from the knowledge base and experts. It uses the common product, process, resource view approach and extends it with the cost-pool view to reflect cost generation aspects. The proposed cost-benefit model uses interval numbers to reflect uncertainty in all cost relevant quantities. This capability is also used in the process of information retrieval from the knowledge base to resolve ambiguities and estimate missing information. The developed approach allows to complete, refine, and integrate information on cost aspects of the robot system. All used knowledge is machine readable. Therefore, a software system can instantiate the cost-benefit model for a robot system and perform computations to assess the robot system over its entire life-cycle.

5 Activity models

In the previous Chapter 4, it has been assumed that it is possible to describe cost-related aspects on each of the levels of the PPRC-approach in an isolated activity model with clearly defined parameters and manageable complexity of computation. These activity models have to be capable of describing the cost creation process in sufficient detail. This chapter introduces examples of activity models to clarify their nature and to substantiate the above mentioned assumption of Chapter 4. Examples of activity models are introduced for the example scenarios outlined in Section 3.1. First, general activity models applicable for general robot systems are introduced. After that, application specific activity models for welding and handling are described.

5.1 General activity models in robotics

The models introduced in this section are not bound to a specific application and can be applied in general to industrial robot systems. Models are based on first principles or literature where possible.

In the description of product and process models, it is distinguished to which life-cycle phase of the robot system the activity models apply. This is necessary as the aspects to be modeled differ between design, integration, commissioning, and set-up on the one hand and regular operation on the other hand. Resource models and cost-pool models are generic and are applied in all life-cycle phases.

Table 5.1 Examples of parameters and properties for components.

Component type	Parameters	Properties
Robot	Payload, repeatability, purchase price, joint lengths, weight	IP-protection class, bus interfaces, manufacturer
Triangulation sensor	x-, z-range, resolution, maximum measurement frequency, weight	IP-protection class, field bus interfaces, manufacturer
Pneumatic gripper	Closing force, jaw displacement, nominal pressure, weight, air consumption per stroke	Manufacturer, gripper type
Electric motor	Maximum torque, stall torque, maximum speed, inertia	IP-protection class, manufacturer
Welding power source	Welding current range, working voltage range, duty cycle percentage, weight	Welding processes, manufacturer, field bus interfaces, IP-protection class

5.1.1 Product models for robot system realization

The product models in the integration phase are a combination of the composition of the robot system in terms of a bill of material and the device descriptions of the contained components including their parameters and properties. This information is typically available from the plant engineering system, in data sheets, or component catalogs. Table 5.1 shows examples of typically available parameters and properties of some component types. As already outlined in Chapter 4, the product model is an information model while the remaining activity models are mathematical models.

As hardware cost typically constitute a significant part of the investment in a robot system, the purchase price of components plays a central role. Other parameters are typically highly specific for the type of the component. The stated parameters might differ depending on the exact specification of the component manufacturer. However, in most cases, it is possible to translate specifications of different component manufacturers into each other.

5.1.2 Process models for robot system realization

Process models for the development and integration phase allow determining nominal required working times for the execution of integration tasks. The development and integration phase encompasses activities such as conceptual development of the robot system, development and integration of different subsystems, and robot system integration as a whole. The life-cycle model shown on the right in Figure 3.4 on page 68 is detailed for the development and integration phase as shown in Figure 5.1. The single phases of this detailed life-cycle model are explained in the following.

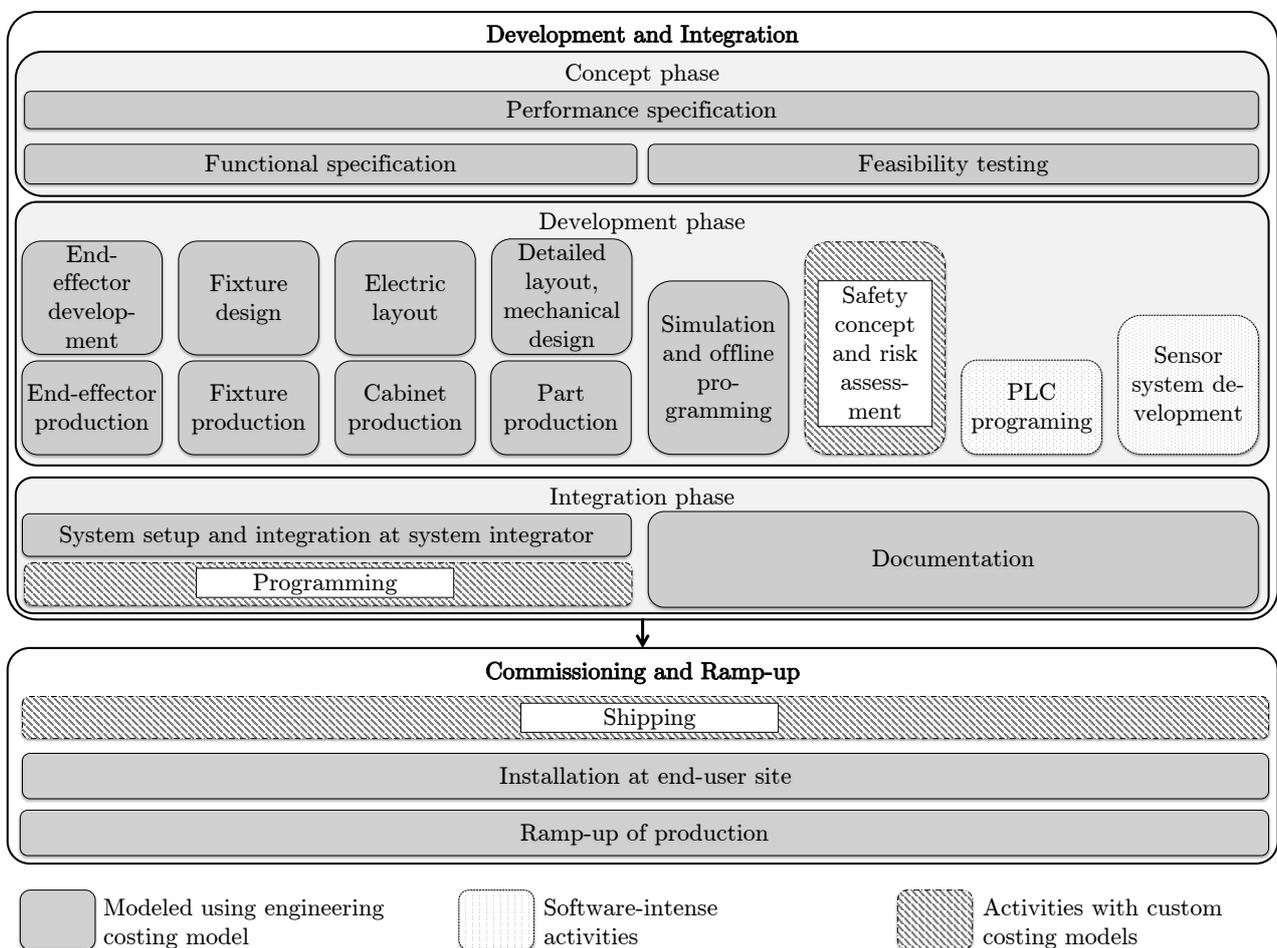


Figure 5.1 Detailed life-cycle during development, integration, commissioning and ramp-up of the robot system.

In the conceptual design phase, the general concept of the robot system is defined. As a first step, the end-user defines the exact requirements in the performance

specification. In doing so, the end-user states the automation problem and defines core performance metrics to be achieved by the robot system. Based on this information, the functional specification is developed by the system integrator. The functional specification defines the active principles used to solve the manufacturing task, a conceptual layout of the robot system, concepts for the robot tools, and a specification of the core functionality. Often, in particular for new system types, feasibility testing is required to validate critical functionality which accompanies the generation of the functional specification. Usually, contract negotiations accompany this phase of system design.

In the development phase, many engineering activities are carried out simultaneously. These activities typically lead to the generation of design artifacts. Based on these artifacts the sub-components of the robot system are created, tested and prepared for integration. This includes general planning tasks such as detailed layouting and simulation of the robot cell. In the development phase, the safety concept is devised and a preliminary risk assessment is carried out. Main tasks in the development phase are in particular the mechanical design of peripherals such as end-effectors and the construction of additional mechanical components such as pedestals and protective fencing. The electric layout is also designed in this phase. The sub-modules of the robot system, for example peripherals and control cabinets and circuits are integrated at the end of the development phase. Also preparatory tasks for initial operation are carried out. Example of such preparatory tasks are offline programming, programming of PLCs and configuration of high level sensor systems, such as image processing solutions.

Finally, the generated components are integrated physically and in software. After physical set-up, the motion programming of the robot is carried out. When offline programming is used, part of the programming has already been done during the development phase. The safety concept is validated in practice. Furthermore, a documentation is written in this phase.

For estimating the efforts occurring due to processes in development, integration, commissioning, and ramp-up, the used approach is derived from literature. For software intense activities, the COCOMO method (see Chapter 2.3.2) is used.

General development tasks are estimated using a modified approach as suggested by Bashir et al., 2001. Some work steps, such as the safety life-cycle, are modeled with custom models tailored to the type of work actually performed during this phase.

Estimation of software development efforts

COCOMO offers a good trade-off between accuracy and available data on the development process as it is well established and only requires input on project type and delivered source instructions n_{DSI} , which is estimated from comparable software projects. This estimate is captured in the knowledge base listing the infimum and supremum of lines of code from past projects depending on the role of the software.

COCOMO allows to estimate the required development effort t_{wo} for organic software projects as

$$t_{\text{wo}} = 2.4 (n_{\text{DSI}} 10^{-3})^{1.05}. \quad (5.1)$$

The COCOMO model is applied to all kinds of software, including custom software for the project, sensor data evaluation, and PLC programming. What cannot be handled with this approach is the reconfiguration of existing software function blocks. How this increasingly common task can be handled in the cost-benefit assessment of robot systems is open to further research. Once a sufficiently large data set is available, the applicability of the COCOMO model and its suggested parameters for software development in automation and PLC programming should be analyzed. However, here the monitoring of software development effort is not further pursued. It is assumed that the COCOMO model offers sufficient accuracy for modeling software development efforts for automation projects.

Estimation of engineering design efforts and development times

Several components of the robot system require detailed mechanical and electrical design. This includes the mechanical and electrical design of the entire work cell as

well as custom-made components such as end-effectors, fixtures, jigs and material feeding solutions (see Figure 5.1 for an overview of design activities).

For estimation of general engineering efforts, the model of Bashir and Thomson (see Chapter 2.3.1) is used. The design effort t_{wo} is expressed as a function of product complexity \mathcal{C} and severity of requirements \mathcal{R} as

$$t_{\text{wo}} = a \mathcal{C}^b \mathcal{R}^c. \quad (5.2)$$

The constants $a \in \mathbb{I}$, $b \in \mathbb{I}$, and $c \in \mathbb{I}$ depend on the development task and the surrounding conditions for development and have to be estimated from past data.

The development activities shown in Figure 5.1 might or might not be required depending on the respective setup of the robot system under scrutiny. Hence, the model of Bashir and Thomson is applied for all development branches shaded in dark grey in Figure 5.1 individually. This allows to estimate the development effort for the single components independent of each other.

In the context of robot systems, it is proposed to interpret the severity of requirements \mathcal{R} as the level of the robot system on the Genefke Scale (see Chapter 2.3.3). The parameter \mathcal{C} for the different activities is chosen from a catalog based on defined criteria for integration activities as shown in Table 5.2, end-effectors as shown in Table 5.3, and fixtures as shown in Table 5.4. These tables show which criteria are applicable for which design function. The value for the criteria is either requested as input from the user or determined through a dedicated analysis of the known description of the robot system. Currently, it is possible to draw the number of robots (\mathcal{C}_5), the number of work stations (\mathcal{C}_6), the size of the end-user (\mathcal{C}_9), additional end-effector axes (\mathcal{C}_{14}), tool changer functionality (\mathcal{C}_{15}), number of fixed parts inserted into a fixture (\mathcal{C}_{20}), additional positioning axes (\mathcal{C}_{23}) and automated feeding (\mathcal{C}_{24}) from design data or data entered in a different context during the cost assessment process. Further information is explicitly retrieved from the user in a questionnaire when required.

The original model of Bashir and Thomson counts the required functions. This im-

explicitly assesses the influence of different functions equally. In order to reflect the differing effort caused by the single functions and criteria listed in tables 5.2, 5.3, and 5.4, the applicable criteria are combined into a consolidated measure for \mathcal{C} using scaling factors F_i :

$$\mathcal{C} = \sum_i (F_i \mathcal{C}_i). \quad (5.3)$$

The applicable factors have been determined from data gathered in expert interviews that were based on past projects. In these interviews, experts were asked to give an assessment of the criteria listed in Appendix 3 and a quantification of the effort of different design activities for their project. The scaling factors were optimized to minimize the error between the prediction of the model of Bashir and Thomson and the expert estimates.

The Genefke Scale as a measure for the severity of requirements \mathcal{R} has a strong impact on the estimated efforts. Generally, robot systems with higher numbers on the Genefke Scale require more effort during design and integration or involve technologies with lower TRL compared to established systems. The model of Bashir and Thompson will predict higher development efforts for models with higher number on the Genefke Scale.

Apart from the effort defining the development cost, also the development time t_A is estimated. This is required to control the resource allocation in the development process on the resource level. As single robot system installations and their sub-components are typically developed by small teams, a constant usage of resources is assumed. Hence, the development time is estimated as

$$t_A = \frac{t_{W_0}}{\sum_i (u_{R_i} t_{R_i})}, \quad (5.4)$$

where t_{R_i} is the daily work time of employee i involved in the development with a relative resource utilization of u_{R_i} .

Table 5.2 Measures related to the functional complexity of the robot system in the concept and realization phase.

Design activity	\mathcal{C}_1 : Clocked production with automatic feeding	\mathcal{C}_2 : Integration with IT-Chain for robot programming	\mathcal{C}_3 : Interfacing with external machines required	\mathcal{C}_4 : Integration with supervisory control system required	\mathcal{C}_5 : Number of robots	\mathcal{C}_6 : Number of work stations for robots	\mathcal{C}_7 : Hard accessibility of parts	\mathcal{C}_8 : Handling of bulky parts or parts that are hard to handle	\mathcal{C}_9 : Large-company end-user	\mathcal{C}_{10} : Process related risk
Performance specification	✓	✓	✓	✓					✓	
Functional spec. and concept design	✓	✓			✓	✓	✓			
Feasibility testing								✓	✓	✓
Layout development					✓	✓	✓	✓		
Simulation	✓				✓	✓	✓			
Electric design			✓	✓	✓	✓				
Setup and integration	✓	✓	✓	✓	✓	✓				
Installation at end-user	✓	✓	✓	✓	✓	✓		✓	✓	
Ramp-up	✓		✓	✓	✓	✓	✓	✓		
Documentation		✓	✓	✓	✓	✓			✓	

Table 5.3 Functions of custom designed end-effectors for determination of functional complexity of the design process.

End-effector	\mathcal{C}_{11} : Accurate (≤ 1 mm) calibration required (e.g. vision sensor)	
	\mathcal{C}_{12} : Advanced sensory capabilities (e.g. force-torque sensing)	
	\mathcal{C}_{13} : Advanced mechanical capabilities (e.g. floating mounting)	
	\mathcal{C}_{14} : Additional axis in end-effector	
	\mathcal{C}_{15} : Tool changer functionality	
	\mathcal{C}_{16} : Supply of solid process media (wire, power)	
Gripper	\mathcal{C}_{17} : Gripper copies shape of work piece	
	\mathcal{C}_{18} : Final work piece shape is determined by gripper	
	\mathcal{C}_{19} : Gripping of easily damageable surface	

Table 5.4 Functions of custom designed fixtures and work piece tables for determination of functional complexity of the design process.

Fixtures	\mathcal{C}_{20} : Number of parts to be fixed
	\mathcal{C}_{21} : Exact positioning of parts by fixture
	\mathcal{C}_{22} : Jigging of parts under process influence
	\mathcal{C}_{23} : Additional axis for positioning (e.g. tilting table)
	\mathcal{C}_{24} : Automatic feeding of parts (e.g. turntable, conveyor)
	\mathcal{C}_{25} : Automatic operation of fixture

System safety design

Robot systems require dedicated activities to ensure safety. This is the so called safety life-cycle of the robot system. The safety life-cycle ensures that the robot system conforms to applicable safety standards and can be operated safely. This process includes the design of protective equipment, for example fences, doors, non-separating opto-electronic safety equipment, such as laser scanners or light curtains, as well as the deployment of safety controllers and underlying logic components. Furthermore, a risk assessment is required in order to document the safety of the robot system and assess any hazards that might result from its usage. Finally, documentation, CE-marking, and declaration of conformity are required. Inspections on the shop floor with occupational health agencies and safety representatives have to be carried out to ensure that the system meets the specified properties and is safe to operate.

The safety life-cycle consists of four main activities, i.e. safety concept, risk assessment, documentation, and inspection, which are described in the following.

Safety concept: The work efforts for the safety concept are included in the layout engineering work step described in Section 5.1.2

Risk assessment: For the risk assessment, it is assumed that n_{WS} safety workshops of duration t_{WS} are required, resulting in a total work effort of $t_{W_0} = n_{WS} t_{WS}$ for each of the participants. These workshops include the plant developer, electrical engineer, process engineer, and safety expert. A good

expert estimate for typical robot cells is $n_{WS} = [2, 3]$ and $t_{WS} = [4, 6]$ h and includes the above mentioned personnel resources.

Documentation: Documentation includes the preparation and documentation of the safety workshops and the preparation of required documents for CE-marking, namely the risk assessment, safety relevant operation instructions and declaration of conformity. A duration $t_{Doc} = [40, 60]$ h, including review, signing, and archiving of the documents is a good estimate for single robot cells. The activity is carried out by a design engineer.

Safety inspection: After the final installation of the robot system at the end-user site, a safety inspection is required involving personnel of the responsible occupational health body, end-user, and system integrator. An expert estimate for the duration of the inspection is $t_{insp} = [2, 3]$ h. Typically 3-5 persons are involved in the safety inspection, namely a design engineer and/or project manager of the supplier, an employee of the occupational health body and the project manager, and/or production engineer of the end-user.

The above mentioned assumptions are not suitable for systems that require dedicated certification, such as components listed in annex V of the machine directive. However, as most robot systems are built from commercial off-the-shelf safety components, this assumption applies to almost all robot systems. Further, the presented estimates only apply to robot systems employing widespread safety principles such as protective fences or non-separating protective equipment. Human-robot collaboration systems relying on other safeguarding mechanisms require different activities in their safety life-cycle and hence other functions for effort estimation.

Acceptance testing

Typically a factory acceptance test (FAT) and a final acceptance test are performed for robot systems. The factory acceptance testing concludes the integration phase and is typically carried out at the system integrator. Subsequently, the system is shipped to the customer site and installed. The final acceptance

Table 5.5 Typical values for constants for the acceptance testing model.

Constant	Unit	Factory acceptance	Final acceptance
$f_{\text{monitoring}}$	%	n/a	[5, 10]
t_{testrun}	h	n/a	[30, 100]
$t_{\text{documentation}}$	h	[1, 2]	[3, 6]
$t_{\text{inspection}}$	h	[1, 2]	[1, 2]

test marks the transition in the life-cycle from the commissioning and ramp-up phase to nominal production. Acceptance testing typically involves an inspection of the system. Additionally, the final acceptance test includes the proof that the system is capable of operating within the defined performance characteristics, in particular availability, quality and production capacity. The effort for acceptance test is therefore modeled through two models as follows. Firstly, the effort for the test run of the system is determined as

$$t_{A_{\text{testrun}}} = f_{\text{monitoring}} t_{\text{testrun}} + t_{\text{documentation}}, \quad (5.5)$$

where $t_{A_{\text{testrun}}}$ is the work effort for the test run, $f_{\text{monitoring}}$ is the time factor for monitoring during the test run, for example what percentage of the test time a worker needs to monitor the test operation of the system, t_{testrun} is the time of the test run, and $t_{\text{documentation}}$ is the time effort for the documentation of the acceptance test.

The test run is typically carried out by technicians of system integrator and end-user. The inspection of the robot system involves a larger group of participants including project leaders and often responsible managers of system integrator and end-user. It is modeled using a constant time effort $t_{\text{inspection}}$. Table 5.5 provides typical values for the introduced constants.

Shipping and transport

Transport costs consist of the cost for loading, overland transport, custom duties, unloading, and the effort for bringing the system to the installation site. Hence, the associated cost-pool model can be written as follows:

$$C_{\text{transport}} = C_{\text{packaging}} + C_{\text{shipping}} + c_{\text{customs}}C_{\text{system}} + C_{\text{customs}} + C_{\text{OnSiteTransport}}, \quad (5.6)$$

where $C_{\text{transport}}$ is the overall shipping cost, $C_{\text{packaging}}$ is the cost for packaging the robot system, loading and arranging transport, C_{shipping} is the transport cost, c_{customs} is the custom duty rate, C_{system} is the sales price of the system, C_{customs} is the customs processing cost, and $C_{\text{OnSiteTransport}}$ is the cost for on site transport, i.e. from unloading to place of installation.

These costs except the packaging efforts $C_{\text{packaging}}$ are modeled as fixed cost items. Shipping cost is modeled depending on the type of transport¹

$$C_{\text{shipping}} = \begin{cases} [800, 1\,500] & \text{€ , nationally} \\ [1\,200, 2\,500] & \text{€ , internationally, overland} \\ [1\,500, 5\,000] & \text{€ , internationally, sea transport} \end{cases} . \quad (5.7)$$

Customs duties can be retrieved from a customs database. Customs cost are not considered further as software for custom duty calculation is available on the market and could be integrated into the system in order to determine the customs rate c_{customs} .

Transport of the robot system to the installation site takes time and requires specialized equipment. The cost for this operation is estimated to $C_{\text{OnSiteTransport}} = [100, 4000] \text{ €}$ and consists of cost for renting specialized equipment, e.g. a forklift for high payload or air cushions for floor transport.

Apart from these fixed cost factors for the robot system, the required work effort for organization of the packaging and creation of shipping papers as well as

¹These figures correspond to typical robot systems which can be loaded on a single truck trailer. This holds true for typical robot systems for the use in SMEs that are addressed in this thesis.

handling of customs processing are modeled as standard effort typically carried out by the project management. As a typical estimate for this task, a work effort of $t_{A\text{shipping}} = [6, 16]$ hours is used.

5.1.3 Product models for robot system usage

Product models for robot system usage are specific to the manufacturing needs of the product. The product models contain general parameters and properties of the product, e.g. weight, material and dimensions. The purchase price of the raw material is included or estimated based on the product properties. Additionally, process specific measures that quantify the processing needs of the product are contained. For example, welded parts contain a specification of length and properties of required weld seams. These parameters are highly specific for the executed processes. Examples of product parameters for specific parts and processes are presented in Section 5.2.

5.1.4 Process models for robot system usage

Process models reflect the production capacity and resource utilization specific aspects of the used manufacturing technologies. Examples for welding, handling, and closely related processes are given in Section 5.2. In the following, only generic process models are listed, that can be used for different production applications.

Online robot programming

The motions of the robots in a robot system need to be programmed to perform the desired manufacturing task. Robot programming means creating a capability of the robot system, i.e. the ability to manufacture a new type of product. Nevertheless, the process is considered to be part of the usage phase as the parameter source for this activity is the work piece itself and its complexity.

Two methods are commonly used. In online programming, which is common in SME scenarios, an operator guides the robot through the task by moving it using buttons or other input modalities on the programming panel of the robot. Therefore, the robot is occupied during the programming task. On the contrary, in offline programming, the robot is programmed in a virtual environment, resembling a CAD-system. Also offline programming usually requires program correction in the work cell due to deviations of real world and virtual model. Currently, new approaches in programming are emerging such as programming by demonstration (Meyer, 2011) or hybrid offline/online programming.

A general expert guess from different projects executed at Fraunhofer IPA is about 1 minute teaching time per path point by an experienced robot programmer (Dietz et al., 2013b). For re-teaching of existing programs, an even lower figure is assumed. The number of program points is estimated based on executed process and relevant parameters of the part, and stored with the product information. For this purpose, the robot programming model is process-specific as, depending on the programmed process, it draws different information from the product model and processes this information differently. The robot programming models for GMAW welding and machine tending are given in sections 5.2.1 and 5.2.2.

Maintenance of robot system components

Maintenance is triggered by the resource model and modeled according to the maintenance model of the robot as described in the next section. Maintenance tasks are described by parameters related to the maintained resource that are listed in Table 5.6.

The maintenance cost model therefore causes direct fixed cost $C_M + C_F$ and cost related to the work efforts caused by maintenance. Yearly maintenance cost typically are about 3-5% of the overall system cost for single shift operation and 6-10% for double shift operation (Lotter et al., 2006). This information is reflected as an interval number in the cost-benefit model.

Table 5.6 Resource parameters for maintenance.

Parameter	Unit	Meaning
C_M	€	Average material cost of maintenance task
C_F	€	Fixed cost for maintenance, e.g. in case of a maintenance contract
t_{Awo}	s	Required working time for maintenance
t_{Aidle}	s	Downtime of the robot system

5.1.5 Resource models

Subsequently, general resource models of robot systems and their key components are described. During integration, the cost relevant resources are mostly personnel carrying out the integration and commissioning. During operation, the robot system and its components are employed as resources in addition to personnel operating and maintaining the robot system.

Robot model

The main cost resulting from the usage of robots stem from their energy consumption and maintenance requirements. Energy consumption of industrial robots heavily depends on the specific robot model (Chemnitz et al., 2011). Anandan, 2016 cites an average power consumption of 7.35 kW for a 100 kg payload robot. Other research focused on improving the energy efficiency of industrial robots and on developing suitable models for the energy consumption (Paryanto et al., 2015). Here, the energy consumption of robots is modeled through two different resource models capturing idling and motion behavior. This allows to reflect phases without motion, where energy is only required for operating the robot controller, and actual motion phases where additional energy is consumed by the drives of the robot.

The idling model captures the consumed electric work W_{Eon} resulting from the

power consumption due to the activation of the robot, for example the energy consumption of the controller or fan, described by \mathcal{P}_{on} .

$$W_{E_{\text{on}}} = \mathcal{P}_{\text{on}} t_{\text{wo}}, \quad (5.8)$$

where t_{wo} is the working time per day, e.g. typically 16 h for two-shift-operation. The model for idle energy consumption runs in the background as time dependent model as it is assumed that the robot system is always running when the production at the end-user site is running, even if no manufacturing tasks are assigned to the robot.

Power consumption due to the motion $W_{E_{\text{motion}}}$ resulting from losses in drives, motors, friction, and intermediate circuit are captured in a separate model. The motion related power consumption is clustered with respect to motion intensity, referring to different operation regimes of the robot:

- Standstill: $\mathcal{P}_{E_{\text{idle}}} = 0 \text{ kW}^2$
- Low speed feed motions: occurring for example during welding or machining with power $\mathcal{P}_{E_{\text{slow}}}$
- High speed positioning motions: occurring during fast transfer in open space, for instance to position a tool or transfer a work piece with power $\mathcal{P}_{E_{\text{fast}}}$

The consumed electric power is computed using the motion specification through distance l_{motion} and speed v_{motion} . Furthermore, the model receives parameters for the specific power consumption of the used robot for slow ($\mathcal{P}_{E_{\text{slow}}}$) and fast ($\mathcal{P}_{E_{\text{fast}}}$) motions and the speed threshold v_{tresh} for switching between these two motion regimes. For fast speed motions, it is further possible to define a motion distance l_{motionfs} in which the robot moves with full speed v_{robot} .

$$W_{E_{\text{motion}}} = \mathcal{P}_{E_{\text{fast}}} \frac{l_{\text{motionfs}}}{v_{\text{robot}}} + \begin{cases} \mathcal{P}_{E_{\text{slow}}} \frac{l_{\text{motion}}}{v_{\text{motion}}} & , \text{ if } v_{\text{motion}} < v_{\text{tresh}} \\ \mathcal{P}_{E_{\text{fast}}} \frac{l_{\text{motion}}}{v_{\text{motion}}} & , \text{ if } v_{\text{motion}} \geq v_{\text{tresh}}. \end{cases} \quad (5.9)$$

²It is assumed here that standstill of the robot does not cause energy consumption. This holds true for longer periods of standstill as the drives of the robot are typically switched off after some standstill time.

The specific values for the parameters $\mathcal{P}_{E_{on}}$, $\mathcal{P}_{E_{slow}}$, and $\mathcal{P}_{E_{fast}}$ are dependent on the robot type. In case no data is available for the specific robot type, a generic estimation based on its payload is applied. The occupation time of the robot t_{occ} is computed as the sum of all motion times of the robot

$$t_{occ} = \frac{l_{motion}}{v_{motion}} + \frac{l_{motionfs}}{v_{robot}}. \quad (5.10)$$

The robot model furthermore tracks the usage of the robot system to predict maintenance activities. These maintenance activities are part of the resource model and therefore also included in the operation life-cycle phase in Figure 5.1. Maintenance can be time-triggered, for example in the case of yearly inspection, or triggered by the actual usage of the robot. For the latter, the accumulated operation time t_{Aacc} is tracked in the model by adding up the operation times of every task i executed by the robot system

$$t_{Aacc_{i+1}} = t_{Aacc_i} + t_{occi}. \quad (5.11)$$

For time triggered maintenance or inspection, the operation time

$$t_{op} = T_i - T_{insp} \quad (5.12)$$

is monitored. T_i refers to the current time, i.e. the time of the execution of the current task, while T_{insp} refers to the time the robot system was last maintained or installed. For both types of maintenance, an event is raised triggering the maintenance if time thresholds are exceeded:

$$\begin{aligned} \text{if } (t_{Aacc} > t_{main,A}) &\rightarrow \text{trigger usage controlled maintenance event} \\ \text{if } (t_{op} > t_{main,inst}) &\rightarrow \text{trigger time controlled maintenance event} \end{aligned} \quad (5.13)$$

The thresholds $t_{main,A}$ and $t_{main,inst}$ are parameters of the robot.

The fulfillment of the condition in equation (5.13) is checked if the resource becomes available after the completion of a manufacturing task. The used model

has already been explained in Section 5.1.4. If a maintenance activity is executed, the time storage in the resource model is reset, i.e. $t_{Aacc} = 0$ for usage triggered maintenance, and $T_{inst} = T$ for time triggered maintenance. Furthermore, the occupation time of the robot is set to the duration of the maintenance task t_{Aidle} to block the resource for usage during the maintenance.

In the same way, the usage monitoring can, in the same way, be used to model system failures and the required repair activities. In this case, an additional resource parameter t_{MTBF} describing the mean time between failures is monitored instead of the maintenance times $t_{main,A}$ and triggers repair activities.

Model of human worker

For workers, the direct personnel cost and number of deployments are the output activity levels of the resource model. The cost of a worker is computed from the personnel cost rate plus an adjustment factor related to the competence of the workers. Furthermore, the number of deployments is computed as

$$\vec{A}_{\text{worker}} = \begin{pmatrix} C_{\text{worker}D} \\ n_{\text{deploy}} \\ t_{\text{occ}} \end{pmatrix} = \begin{pmatrix} c_{\text{worker}} f_t t_A \\ \lceil \frac{t_A}{t_d} \rceil \\ f_t t_A \end{pmatrix}, \quad (5.14)$$

where \vec{A}_{worker} are the activity levels of the worker model consisting of the direct work cost $C_{\text{worker}D}$ and the number of deployments n_{deploy} , c_{worker} is the hourly cost rate of the worker and f_t a time correction factor. t_d is the average working time of the worker per shift, typically the daily work time. The number of deployments for a specific work task is computed by rounding the fraction of actual work time and daily work time to the next higher integer ($\lceil \cdot \rceil$), thus assuming exactly one deployment for each day during which the worker is occupied with the task. t_{occ} is the occupation time that the worker is occupied with the task at hand.

Model of software developer

Again, by using the COCOMO model, the required development time of a project t_{dev} is described by the relation

$$t_{\text{dev}} = 2.5(t_{\text{wo}})^{0.38}, \quad (5.15)$$

where t_{wo} is the required work time for the project according to the process model given in Section 5.1.2. Using this, the required number of developers n_{dev} can be computed from the development time and required effort $n_{\text{dev}} = t_{\text{wo}}/t_{\text{dev}}$.

This information is used to compute the required effort in the resource model using

$$\vec{\mathcal{A}}_{\text{developer}} = \begin{pmatrix} C_{\text{developer}D} \\ n_{\text{deploy}} \\ t_{\text{occ}} \end{pmatrix} = \begin{pmatrix} c_{\text{developer}}f_t t_{\text{dev}}n_{\text{dev}} \\ 0 \\ t_{\text{dev}}f_t \end{pmatrix}, \quad (5.16)$$

where $\vec{\mathcal{A}}_{\text{developer}}$ are the activity levels of the software developer model consisting of the direct work cost $C_{\text{developer}D}$ and the number of deployments n_{deploy} , the hourly cost rate of the developer $c_{\text{developer}}$, and a time correction factor f_t . The number of deployments n_{deploy} is constantly set to zero, as software development is considered an office job without additional costs for deployment

The occupation time t_{occ} for the software developers is the actual development time t_{dev} as computed by equation (5.15) with the time correction factor for developer involvement and efficiency. It is assumed that the number of developers can be scaled in accordance with the optimal number n_{dev} . The underlying assumption is that the required number of developers can be contracted on the market which holds true for general programming qualifications.

5.1.6 Cost-pool models

The used cost-pool models are all overhead models. A fixed percentage of cost is added to the activity volume resulting from the resource models. This relates

mostly to personnel cost overheads and purchase cost overheads. Other more elaborate models including more accounting details or details of price formation are possible but not in the scope here.

Personnel cost-pool model

The total personnel cost C_W is computed by applying the personnel cost overhead $o_{personnel}$ of the company and taking into account an average deployment cost C_{deploy} as follows

$$C_W = (1 + o_{personnel}/100) C_{workerD} + n_{deploy} C_{deploy}. \quad (5.17)$$

The deployment cost depends on the specific realization project, as it captures travel cost estimates that depend on travel times and distance and has to be specified by the user.

Electric energy cost-pool model

Electric energy cost C_E is computed using a simple proportional model

$$C_E = c_E W_E, \quad (5.18)$$

where c_E is the cost of electric energy per kilowatt-hour and W_E is the consumed electric energy.

Material purchase cost-pool model

The purchase cost of material C_M is modeled through a simple overhead model with the overhead rate $o_{purchase}$

$${}_4C_M = (1 + o_{purchase}/100) {}_1C_M, \quad (5.19)$$

where ${}_1C_M$ is the cost of material on the product level and ${}_4C_M$ is the cost of material on the cost-pool level.

5.2 Application specific activity models

In this section, application-specific activity models for the example applications introduced in Section 3.1 are presented. The models are specific for the processes and resources applied in the example scenarios, but transferable to robot systems carrying out similar processes.

5.2.1 Robotic GMAW welding

When estimating the cost for seam welding, three approaches are known as described by the European Welding Foundation (EWF) (EWF, 2007) listed in decreasing order of accuracy:

- Cost per unit: The actual welding times and resource consumption for the product are measured and transformed into cost.
- Cost per length: A feature-based approach attaching cost to the unit length of the weld seams.
- Cost per weight: Computing the cost per mass unit of filler material.

For the cost-benefit model, an approach following the cost per length approach is chosen as this is a feature-based approach compatible with the model setup and as weld seam length can easily be determined from the product's CAD model.

Product model parameters and properties

The product model has to contain all information required for the chosen cost per length approach, namely the length of each weld seam l_{seam} , the weld seam

Table 5.7 Cost relevant parameters contained in the product model for GMAW seam welded parts.

Parameter	Unit	Description
Seam length	m	Length of each individual weld seam
Seam type	-	The type of weld seam describing the topological arrangement of parts
Seam geometry	mm, °	Parameters specifying the relevant seam geometry depending on the seam type

geometry and a reference of the process to be applied, described in a welding procedure specification (WPS). Table 5.7 lists product parameters used for GMAW welded parts.

GMAW welding process models

The production of arc welded parts is a complex process that comprises both, the welding process itself and further auxiliary processes. The most relevant process models are presented below.

GMAW welding The process model for welding is parameterized through the data in the welding procedure specification (WPS). The parameters contained in the WPS are listed in Table 5.8.

The welding speed v_{feed} and the number of passes n_{weld} are required to compute the pure welding duration (arc-on duration) t_A

$$t_A = n_{\text{weld}} \frac{l_{\text{seam}}}{v_{\text{feed}} c_{\text{conv}}}, \quad (5.20)$$

where l_{seam} is the length of the weld seam that can be read from the product model as outlined in the previous section. The constant c_{conv} converts the units of the welding speed into standard SI units. Unit conversions between activity models can be handled automatically by the model using the framework described in Section 6.1.2.

Table 5.8 Typical parameters contained in the welding procedure specification (WPS) of GMAW processes.

Parameter	Unit	Symbol	Description
Wire diameter	mm	d_{wire}	The diameter of the welding wire.
Welding current	A	I_{weld}	The maximum and minimum electric current during welding.
Welding voltage	V	U_{weld}	The maximum and minimum electric voltage during welding.
Shielding gas flow rate	l/m	Q_{g}	The flow rate of the shielding gas during welding.
Wire feed rate	m/min	v_{wire}	The feed rate with which the welding wire is supplied in the process during welding.
Welding speed	cm/min	v_{feed}	The speed with which the welding torch is moved along the weld seam.
Pass of weld seam	-	n_{weld}	Number of seams that are required for a joint.
Welding temperature	°C	T_{weld}	The required temperature of the part for welding. This temperature typically is adjusted by preheating the part.

Approach Motions For each weld pass, the robot has to approach the work piece and retract from it. The process comprises a coarse approach phase with high speed and a slower fine approach phase.

The high speed approach motion is modeled depending on the covered approach motion length l_{motion} , which is estimated using the work piece dimensions measured by the bounding box of the work piece where \check{l}_{bb} is the minimum and \hat{l}_{bb} is the maximum edge length of the bounding box. These dimensions are interpreted as

upper and lower bound of the approach motion as the robot will typically retract to a safe position after the weld pass. Thus the motion length becomes

$$l_{\text{motion}} = [\check{l}_{\text{bb}}, \hat{l}_{\text{bb}}]. \quad (5.21)$$

The slow approach motion is modeled through a constant time interval $t_{\text{Aslow}} \in \mathbb{I}$. This is sensible as the approach motion is typically executed in a defined distance to the seam, e.g. 10 cm, with a defined speed, e.g. 1 cm/s. t_{Aslow} is a parameter of the process. This information is then used in the resource model of the robot which has been presented in Section 5.1.5.

Preheating of part For some welding processes, a preheating of the part is required. In case this process is performed in the work cell by placing a heater at the work piece or through manual preheating, the resulting downtime needs to be considered in the cost estimation. The required preheating energy W_{Th} is computed from specific heat capacity c_{wp} and mass m_{wp} of the work piece, the required preheating temperature T_{weld} from the WPS and the ambient temperature T_{ambient}

$$W_{\text{Th}} = m_{\text{wp}} c_{\text{wp}} (T_{\text{weld}} - T_{\text{ambient}}). \quad (5.22)$$

From this, the time for preheating t_{Aheating} is computed by taking the effective heating power of the used burner torch $\mathcal{P}_{\text{Thheater}}$ as input parameter

$$t_{\text{Aheating}} = W_{\text{Th}} / \mathcal{P}_{\text{Thheater}}. \quad (5.23)$$

The time for preheating t_{Aheating} corresponds to the execution time of the heating process which is supplied to the resource model of the heater and the robot (i.e. for the robot $t_{\text{Aidle}} = t_{\text{Aheating}}$ if preheating is carried out in the work cell). Personnel effort t_{Aworker} can be derived depending on the process execution

$$t_{\text{Aworker}} = \begin{cases} t_{\text{Aheating}} + t_{\text{Apreparation}} & , \text{ for manual preheating} \\ t_{\text{Apreparation}} & , \text{ for heating with fixed heater} \end{cases}. \quad (5.24)$$

Here, $t_{\text{Preparation}}$ is the time required for preparation of the equipment which is modeled as a constant parameter.

Robot programming The robot programming model for seam welding uses the program point based approach introduced in Section 5.1.4. The number of program points is either contained in the product model or is estimated based on the number of weld seams contained in the product model. In the latter case, the number of points on the weld seam is estimated to be 2 for straight seams. For curved segments, it is assumed that program points are distributed evenly along the length with a constant distance $L_S = [30, 50]$ mm. For each seam, additionally four points for close approach and retreat are added. From this estimate, the programming time can be estimated using the 1 minute per point estimate.

Resource models specific for GMAW welding - the welding power source

The only resource model specific to the GMAW process is the model of the welding power source and associated process equipment. The welding power source is the key piece of equipment for the welding process as it transforms electric energy for the welding process. The device energy efficiency η_{weld} of the welding power source is defined as the ration of welding power $\mathcal{P}_{\text{weld}}$ supplied to the process and grid power \mathcal{P}_{G} drawn from the power grid, i.e. $\eta_{\text{weld}} = \mathcal{P}_{\text{weld}}/\mathcal{P}_{\text{G}}$. The welding power is computed using through $\mathcal{P}_{\text{weld}} = U_{\text{weld}}I_{\text{weld}}$, where U_{weld} is the welding voltage and I_{weld} the welding current as indicated in the WPS. Therefore, the energy consumption of the power source W_{EPS} can be computed as

$$W_{\text{EPS}} = \frac{U_S I_S}{\eta_D} t_A. \quad (5.25)$$

State of the art welding power sources reach a device energy efficiency of $\eta_D = [0.8, 0.9]$ (Hälsig, 2015).

The consumed welding wire l_{wire} is computed from the filler volume of the weld seam, which depends on the filler area S_{seam} and the length of the seam l_{seam} and the wire diameter d_{wire}

$$l_{\text{wire}} = \frac{4 l_{\text{seam}} S_{\text{seam}}}{\pi d_{\text{wire}}^2}. \quad (5.26)$$

This equation neglects the evaporation of components of the filler wire which is negligible for solid filler wire. For hollow core filler wire, the formula needs to be adapted to only take into account the part of the filler wire volume that is actually supplied to the weld seam.

The weld seam filler area A_{seam} is computed from the specified geometry of the weld seam of the work piece. Figure 5.2 shows the related parameters for fillet and butt welds. Approximating the weld reinforcement cap by a triangle, this results in

$$S_{\text{seam}} \approx \begin{cases} a^2 + \frac{ah}{\sqrt{2}} & , \text{ for fillet welds,} \\ \frac{Wh}{2} + gt + \frac{(W-g)t}{4} & , \text{ for butt welds,} \end{cases} \quad (5.27)$$

where the geometric features and their symbols are shown in Figure 5.2.

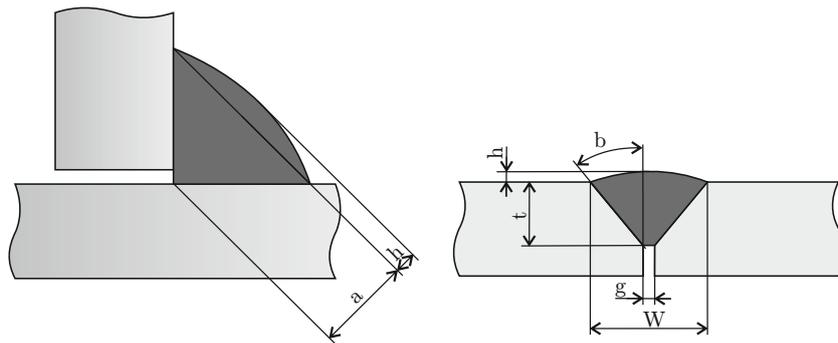


Figure 5.2 Parameters of fillet weld (left) and butt weld (right).

The gas consumption V_g is computed from the gas flow rate Q_g contained in the WPS and the activation time t_A

$$V_g = Q_g t_A. \quad (5.28)$$

Table 5.9 Cost relevant product parameters for handling tasks.

Parameter	Unit	Description
m_P	kg	Weight of the part to be handled
a_P	m/s ²	Maximum acceleration that the work piece is able to endure
t_{cyc}	s	The cycle time with which the part needs to be fed to downstream processes
Grasp type	-	Type of grasp, i.e. 2/3 finger parallel, multi-fingered, planar vacuum, suction cup, fixture

5.2.2 Machine tending robot for work piece handling

In the following, activity models that are specific for the machine tending robot are detailed. The machine tending application was introduced in Section 3.1.2.

Product model parameters and properties

The required product model parameters and properties for handling are listed in Table 5.9. It is noted that the required cycle time is listed as a product parameter. This is the case as the required feeding speed is interpreted as a manufacturing requirement of the product. In an interlinked manufacturing system with fixed pace, the cycle time for the feeding would result from the planning process involving required sales volume leading to the definition of manufacturing content of single robot stations. The product parameters are supplied to the process models for computing the details of the handling process as outlined in the following section.

Process models specific for handling tasks

The main handling process is the transfer of the work piece by handling through the robot. The time for handling is computed under the assumption of constant acceleration as

$$t_{A_{\text{move}}} = \begin{cases} 2\sqrt{\frac{l_{\text{move}}}{a_{\text{move}}}} & , \text{ if } \sqrt{l_{\text{move}}a_{\text{move}}} < v_{\text{move}} \\ \frac{l_{\text{move}}}{v_{\text{move}}} + \frac{v_{\text{move}}}{a_{\text{move}}} & , \text{ else} \end{cases}, \quad (5.29)$$

where v_{move} is the maximum motion speed, being a parameter of the robot (resource), a_{move} is the maximum acceleration chosen as the minimum of the maximum acceleration endurable by the work piece a_{P} and the robot a_{robot} , i.e. $a_{\text{move}} = \min(a_{\text{P}}, a_{\text{robot}})$. The motion distance l_{move} can be estimated from the resource model of the work cell from the position of the material feeder and the target position of the work piece, i.e. $l_{\text{move}} = \|\vec{l}_{\text{dest}} - \vec{l}_{\text{feeder}}\|$.

The same model is used for the reverse motion of the robot. However, for motion without work piece, a_{P} is set to infinity and hence it follows that $a_{\text{move}} = a_{\text{robot}}$.

From this information the input to the robot model can be generated as

$$l_{\text{motion}} = l_{\text{move}}, \quad (5.30)$$

$$v_{\text{motion}} = \frac{l_{\text{motion}}}{t_{A_{\text{move}}}}. \quad (5.31)$$

Additionally to the transfer, the grasping of the work piece contributes significantly to the required handling time. The grasping of the work piece consists of an approach phase, possibly a sensor-based localization phase, the grasping, and a lifting motion that has to be performed with lower speed, depending on the work piece properties. The time required for grasping depends on the type of grasp that is performed.

$$t_{A_{\text{slow}}} = t_{\text{grasp}}, \quad (5.32)$$

where $t_{\text{grasp}} \in \mathbb{I}$ is a fixed time which is a parameter of the grasp type property of the work piece. Furthermore, the model produces the number of grasps n_{grasp}

that are performed as an input to the resource model of the gripper. The release of the work piece is modeled with the same model as grasping.

Robot programming The robot programming model for handling uses the program point based approach introduced in Section 5.1.4. The number of program points is fixed per part. It is assumed that per part the programming of each 3 points for pick and place and two additional transfer points are required. This leads overall to 7 points per part. From this estimate, the programming time can be estimated using the 1 minute per point estimate.

6 Implementation and parameterization

This chapter describes the relevant implementation details of the cost-benefit model and how the different activity models are parameterized using the knowledge base introduced in Chapter 4. The aim is to show how the method developed in previous chapters can be transformed into a usable software system. This software implementation creates the basis for the evaluation of the cost-benefit assessment in Chapter 7.

The implementation architecture of the cost-benefit system is shown in Figure 6.1. The different building blocks forming the architecture relate to either the management of knowledge or the cost-benefit model itself. This chapter follows the structure of the implementation. First, the software implementation of the cost-benefit model is introduced. Subsequently, the implementation of the system for knowledge management and in particular of the knowledge base are explained in detail.

6.1 Implementation of cost modeling

The cost-benefit model represents the cost structure of the automation system and allows the computation of different scenarios with respect to its setup and usage. It has to be distinguished between the information model and the associated run-time component. The information model contains the actual information relevant for cost-benefit computation. The run-time component is able to

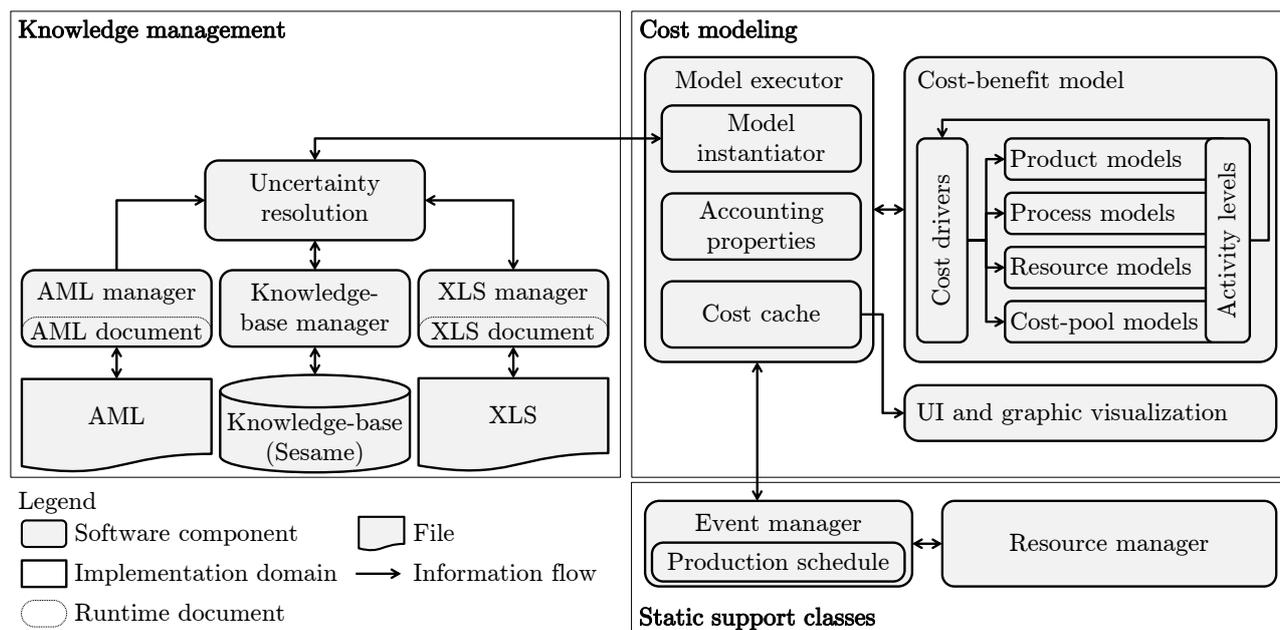


Figure 6.1 Architecture of the cost model implementation.

run computations based on this information model. However, as the model is instantiated during run-time, the two aspects of the model are closely interrelated. In the following, the term cost model refers to the information model while the run-time component is addressed in Section 6.1.2 where the calculation of the model is explained in detail. The run-time component is a dynamic network of activity models on all PPRC-levels and software components for executing this network, i.e. computing the activity models and handing over results to other activity models. The cost-benefit model is created automatically during run-time by interpreting information on the structure of the system stored in the AutomationML file. This process is called the instantiation of the cost-benefit model. This instantiation is handled through the cost model manager and instantiator component, and is described in detail together with the involved software components in Section 6.1.2.

All software components are implemented in Visual C#. The graphical user interface is realized as a Windows Presentation Foundation (WPF) application. Further components are supplied as .NET class libraries. The implementation makes wide use of C# delegates (Drayton et al., 2003), a concept very similar

to function pointers, to store links between the different components of the cost-benefit model and propagate results during execution.

6.1.1 Implementation of cost-benefit model

The cost-benefit model manages the single activity models and their connections. For this purpose, it contains references to all contained activity models, all available cost drivers, and all activity levels. This collection spans all involved PPRC-levels, i.e. product, process, resource, and cost-pool. The cost-benefit model provides capabilities to change, to add, and to delete activity models. Furthermore, it contains the interface to all the involved activity models to trigger the execution of the calculations in case new input data is available.

Implementation and uncertainty arithmetics

All instances of objects that store data are derived from a common, abstract ancestor class named `CostObject`. The `CostObject` class offers the infrastructure to identify and find all objects of the cost-benefit model through a uniform resource identifier (URI). This ensures that all components of the model can be identified and addressed unambiguously. The child classes that derive from this URI mechanism are the activity models (`ActivityModel`) also containing the activity levels, the event handlers (`EventHandler`), and the cost drivers (`CostDriver`) as shown in Figure 6.2.

The activity models in turn follow the class hierarchy shown in Figure 6.3 to differentiate models with respect to their usage and interface. It is distinguished between product models and calculation models. Product models do not express any computations, but provide information storage for the product information. They exist for the robot system and its components (`MachineProductModel`) and for the product manufactured on the machine (`ProductionProductModel`). Calculation models provide the infrastructure for handling uncertainty calculations.

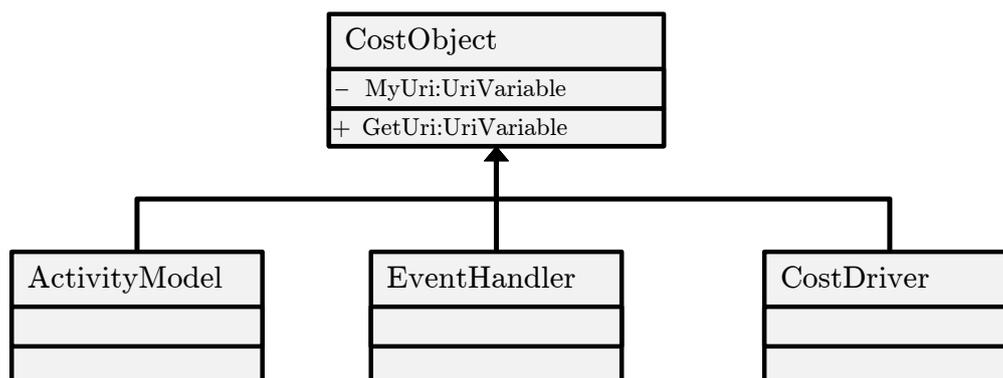


Figure 6.2 Top level class architecture.

Separate child classes exist for the cost-pool yielding cash flows (*AbsCostPoolModel*), process models (*ProcessModel*), and resource models that are registered in the resource manager (*ResourceModel*). The cost-pool models further exist as traditional cost-pool models taking input from the resource-level (*CostPoolModel*) and purchase models that take event based input directly from the product-level (*PurchaseCostPoolModel*).

To make uncertainty computation available throughout the model, a class for uncertain numbers with overloaded operators for basic arithmetic operations is used for all information processing. Special classes representing cash flows, interest rates, activities, and cost drivers use this class to represent values internally. This implementation ensures that interval arithmetic can be used throughout the entire cost-benefit model implementation. All input parameters of the model, including interest rates on capital, cash flows, assumed work effort, or basic costs like hourly material cost, can therefore use uncertain numbers as all classes for the cost model build on the uncertainty arithmetics. As interval numbers are a generalization of real numbers, this approach also works for computations without uncertainty, i.e. using intervals of the type $[a, a]$ with zero interval width.

Activity models for single PPR-levels

The activity models form the atomic building blocks in which knowledge about effort and cost information is stored. Each activity model expresses a relation

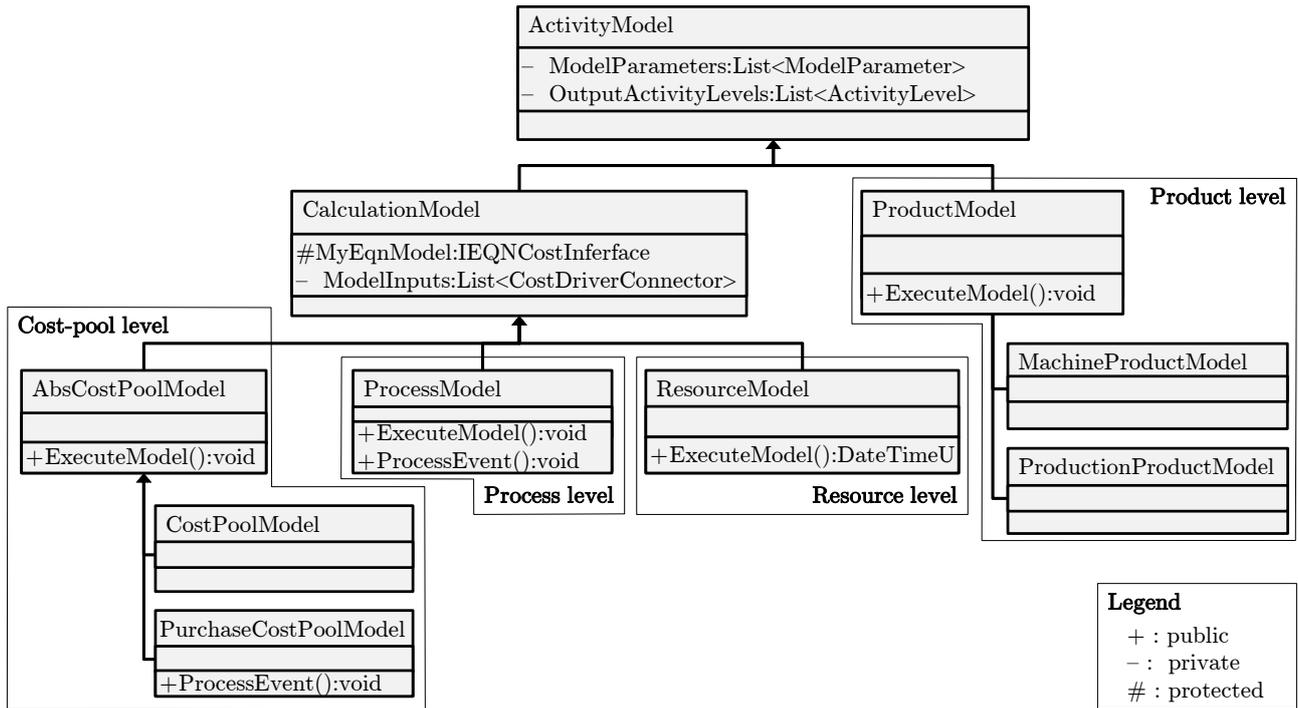


Figure 6.3 Class architecture of different types of activity models.

between cost drivers as inputs and activity levels as outputs. The run-time implementation of the activity models is required to be able to handle uncertainty in the form of interval numbers.

As outlined in Section 4.3.2, three approaches for information storage and related approaches for run-time representation of activity model have been implemented. Firstly, a proprietary approach in which activity models are serialized in a specialized XML-format which is parsed during model instantiation and represented by custom-made classes during run-time. Secondly, Python representations of the model can be included in the run-time component of the cost-benefit model. For both implementation types, the activity model class encapsulates the implementation details and offers an interface to the overall cost-benefit model. This capsule retrieves the input information from the governing cost drivers before execution and makes the inputs available to the related functions, either represented by the respective class structure for interval computations or by a Python script running in a Python environment for .NET. A third option is to utilize managed code DLLs implementing activity models. However, this option sacrifices readability of

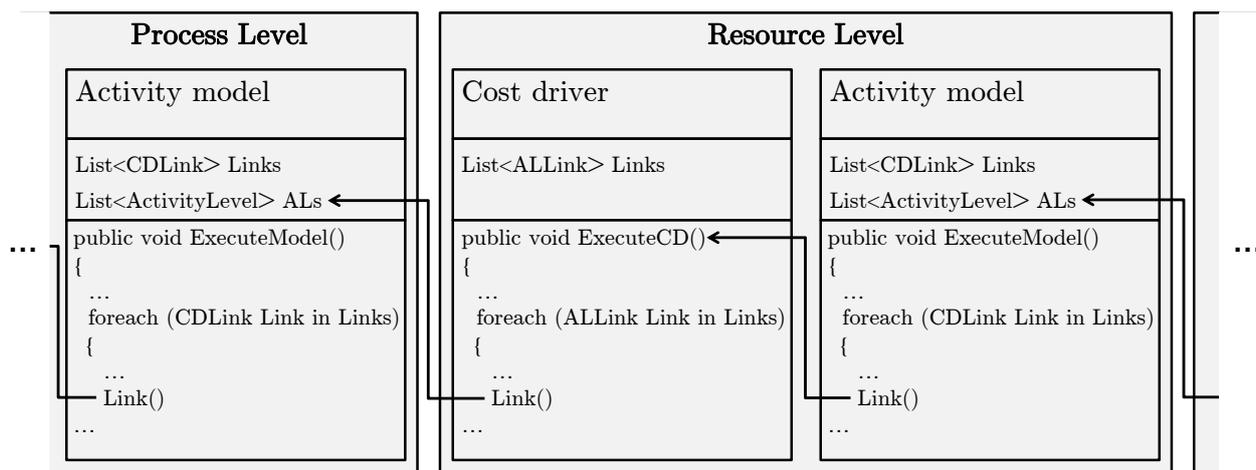


Figure 6.4 Mechanism for data retrieval and provision by activity models.

the models and requires prior compilation of models. On the other hand, classified information can be better hidden.

The activity models are linked to the input cost drivers through delegates. This mechanism ensures that activity models on a level i of the PPRC-hierarchy automatically collect the required input information from level $i-1$ upon computation. During instantiation (see Section 6.1.2), a delegate of the value retrieval function of each input cost driver is stored in the activity model. Through this delegate, the current value for the cost driver is retrieved before computation of the related activity model. The value retrieval function of the cost driver in turn triggers the update of the cost driver through retrieval and summation of the values of the activity levels that feed this cost driver, i.e. one level above in the PPRC-hierarchy. This is done through delegates of the activity levels that are called by the cost driver. These delegates have been created and stored in the cost driver during model instantiation. The activity model evaluation subsequently performs the computation of the related functions and writes the results to the activity level, which is managed by the individual activity model. Through this mechanism, it is possible to execute the entire model by consecutively executing all activity models on the product level, process level, resource level and cost-pool level. The process is depicted schematically in Figure 6.4.

In the following the three approaches for representing activity models are described in detail.

XML-representation of activity models The activity models can be described in a specialized XML-dialect that specifies the required computations. The XML-dialect describes the inputs and outputs of an equation so they can be interfaced with cost drivers and activity levels in the cost-benefit model. Furthermore, internal parameters can be specified. The XML-description specifies how inputs and outputs are connected by introducing one or more functions with nested operations in the form of XML-nodes. These XML-nodes take two inputs which can be an input to the activity model, an internal parameter or another operation. Listing 6.1 shows an example of a simple work time activity model

$$C_{W_0} = t_{W_0}c_{W_0}, \text{ where} \quad (6.1)$$

C_{W_0} is the resulting cost, t_{W_0} the required work time, and c_{W_0} the hourly cost rate. Line 1 specifies the name of the model. In lines 2-6 the input and output quantities as well as internal parameters are defined. In lines 7-12 equation (6.1) is defined. Lines 13-18 describe the equation for computing resource occupancy time, which is required for resource models.

Listing 6.1 Example of a simple, linear overhead model for personnel cost

```
1 <PPRCostModel name="GenericWorkTimeModel">
2     <input name="RequiredWorkTime" unit="Hours"></input>
3     <input name="PersonnelCostPerH" unit="CostPerHour"></
4         input>
5     <output name="Cost" unit="EUR"></output>
6     <output name="ResourceOccupancyTime" unit="Hours"></
7         output>
8     <parameter name="P2" unit="Dimensionless" iotype="
9         ScalingFactor">1</parameter>
10    <equation output="Cost">
11        <operation type="mult">
```

```

9           <inin>RequiredWorkTime</inin>
10          <inin>PersonnelCostPerH</inin>
11        </operation>
12    </equation>
13    <equation output="ResourceOccupancyTime">
14        <operation type="div">
15            <inin>RequiredWorkTime</inin>
16            <parain>P2</parain>
17        </operation>
18    </equation>
19 </PPRCostModel>

```

The import of XML-files allows to directly build a run-time representation of the activity model while being able to alter the activity models without the need for compilation. During instantiation of the model this description is parsed by a run-time component and class instances are created that implement the modeled functions. For this purpose, the parser works through the tree of operations that is specified in the XML-description and creates an instance of each respective operation. The operation tree is parsed recursively. Therefore, the implementation for the XML-representation of activity models does not permit the definition of computational loops. The constructor of the classes that implement the algebraic operations returns a delegate to the execution function of the operation. This allows each operation to call the operations that form its branches. Finally, the leaves of the operation tree, i.e. the inputs to the parsed function, are either cost drivers or parameters of the function. In order to retrieve these values during run-time, a delegate to a value retrieval function is stored during parsing.

The focus of the XML-based approach is implementing a simple solution to using activity models stored in a knowledge base. The approach has shortcomings as complex control flows like loops or operations involving multiple operators are not possible. As most costing operations are simple models (see Chapter 5) and due to the good availability of XML, this approach is sufficient for typical activity models. Other description formalisms like in particular MathML could be used. However, due to the limited feature set of the implementation and hence the low coverage of MathML features this approach was dismissed.

Python representation of activity models Alternatively to the basic capabilities of the proprietary approach, it is possible to utilize Python scripts implementing the models. The inclusion of Python scripts allows the specification of more complex activity models including matrix operations and loops. These scripts can be executed during run-time without prior compilation. Listing 6.2 shows an example of the simple overhead model in equation (6.1). In lines 1-10 the required dll-libraries for uncertainty arithmetics and required data types also used in the C#-context are imported. In lines 12-21 data structures for the inputs and outputs are created. The actual values of the inputs are read in lines 23-36. The computational operations are defined in lines 41 and 42. Lines 44-55 hand back the results to the C#-context. It is obvious that the python implementation involves more overhead for transferring input and output information between the C#-context and the python-context compared to the XML-approach. On the other hand the definition of computations is much easier in python. This allows to implement more complex computations, for example also involving control structures.

Listing 6.2 Example of a simple, linear overhead model for personnel cost in Python

```
1 #-----
2 #Import dll's required for models
3 #-----
4 import clr
5 clr.AddReferenceToFile("UncertaintyArithmetics.dll")
6 from UncertaintyArithmetics import doubleU
7 clr.AddReferenceToFile("Costing.dll")
8 from Costing import OperationInput
9 from Costing import ActivityLevel
10 from System.Collections.Generic import List
11
12 #-----
13 #Get the list of inputs and outputs
14 #-----
15 #Get outer inputs
16 InputLst = List[OperationInput]()
```

```
17 InputLst.AddRange(GetOuterInputs())
18
19 #Get outer outputs
20 OutputLst = List[ActivityLevel]()
21 OutputLst.AddRange(GetOuterOutputs())
22
23 #-----
24 #Read the inputs
25 #-----
26 #Find the required work time
27 RequiredWorkTime = 0;
28 for Input in InputLst:
29     if Input.IOType.VariableUri.ToString() == 'http://www.
        semanticweb.org/tad/ontologies/2013/8/RoboCostOntology
        #RequiredWorkTime':
30         RequiredWorkTime = Input.DoOperation()
31
32 #Find the personnel cost per hour
33 PersonnelCostPerH = 0;
34 for Input in InputLst:
35     if Input.IOType.VariableUri.ToString() == 'http://www.
        semanticweb.org/tad/ontologies/2013/8/RoboCostOntology
        #PersonnelCostPerH':
36         PersonnelCostPerH = Input.DoOperation()
37
38 #-----
39 #Do computations
40 #-----
41 Cost = RequiredWorkTime * PersonnelCostPerH
42 ResourceOccupancyTime = RequiredWorkTime
43
44 #-----
45 #Set the outputs
46 #-----
47 #Find and set the output cost
48 for Output in OutputLst:
49     if Output.IOType.VariableUri.ToString() == 'http://www.
```

```
        semanticweb.org/tad/ontologies/2013/8/RoboCostOntology
        #Cost':
50         Output.AddtoLevel(Cost)
51
52 #Find and set the output resource occupancy time
53 for Output in OutputLst:
54     if Output.IOType.VariableUri.ToString() == 'http://www.
        semanticweb.org/tad/ontologies/2013/8/RoboCostOntology
        #ResourceOccupancyTime':
55         Output.AddtoLevel(ResourceOccupancyTime)
```

For executing Python scripts in the .NET environment, the IronPython implementation is used. IronPython allows the utilization of .NET class libraries in the Python scripts. This allows to utilize all functions and classes available in the cost-benefit model in the Python scripts. The Python scripts are called during run-time from the C# code and the evaluation of equations is carried out in Python. Through this, activity models with much more complex control flows can be included. However, this flexibility comes at the cost of an increased overhead during execution. This overhead stems from the additional IronPython environment in which the Python scripts are executed and the overhead of Python.

Use of .NET-DLLs implementing activity models The inclusion of activity models as managed code DLLs offers a straight forward way of including additional models directly written in a high level programming language. The DLLs conform to the interface also used by the import classes for the XML-importer and the Python executor. The inclusion of managed code DLLs during run-time uses the standard means of the .NET framework. This alternative is easy to integrate into the cost-benefit model. It allows to easily hide implementation details of the activity models in order to protect business intelligence. However, the proprietary nature and need for pre-compilation makes this approach less flexible to use.

6.1.2 Run-time representation of the cost-benefit model

The complete cost-benefit model is built as a run-time representation dynamically from the activity model implementations that form its building blocks. It represents the costing structure of the robot system and its application. The implementation of the ability to store the created cost-benefit model is possible, but not realized at the moment, as the core focus here is on the structure of the model. In order to obtain a life-cycle spanning assessment of the robot system, a component controlling the execution, the so called model executor is implemented. This component manages the evaluation of the single activity models forming the cost-benefit model.

Model instantiation and management

The cost-benefit model is built from models for the single aspects of the robot system (costing models). These activity models are provided in serialized form and parsed into run-time representations as explained in the previous section. The so called model instantiator component integrates the single activity models into the overall cost-benefit model. The model instantiator processes the data on the current design status of the robot system once it is translated into RDF (cp. Figure 4.8 on page 113) and builds the cost-benefit model with information from the knowledge base.

The instantiation order follows the logical connection of different concerns in the model as introduced in Chapter 4 and as shown in Figure 4.8. Firstly, the model instantiator builds the parts of the model that represent the development and integration phase of the robot system. The manufactured product is the robot system itself, and hence the model instantiator iterates through the plant description, retrieves all assemblies and components of the robot system, and questions the knowledge base on required activities and purchase prices of these components. Information on related activities and used resources is iteratively retrieved from the knowledge base and the model is built starting from the product level down to the cost-pool level. Resources in this phase are mostly personnel

resources. The related cost parameters are retrieved from the selected system integrator.

Once the model for the integration and commissioning of the robot system is built, the model instantiator moves on to building the model for the operation phase of the robot system. Here, the robot system is the main resource used in production, while the actual products manufactured by the robot system govern which activities are carried out. Again, the model is built starting from the product level working down to the cost-pool. The resource models in the operation phase are parameterized through the product models during the integration phase (see Chapter 4.2.4). This means the same resource identifiers are used to retrieve system parameters from the knowledge base that were identified in the system setup. In case multiple components can carry out required production tasks, the ambiguity is resolved through requesting information from the user of the costing tool.

The instantiation process of different activity models is very similar. In a first step, a function for the creation of the model is called. This function parses the model information stored in the knowledge base. In case of process models, processes that are a precondition of the current activity are recursively instantiated by calling the model creation function. As a first step instantiating an activity model, the cost-benefit model is queried to find out if the model that is to be instantiated is already contained in the model. In order to build the models, the required downstream models in the PPRC-hierarchy are instantiated. For example for process models, the associated resource and cost-pool models are created before the process models are created themselves. Which downstream models are to be instantiated is contained in the model information parsed at the beginning of the instantiation process. Once this is completed, the actual activity model is built in a separate function. This build function first prepares the environment for the new activity model. This includes the creation of activity levels, model parameters, and cost drivers. Subsequently, the actual model class is instantiated. Figure 6.5 shows an overview of the different operations. Finally, for models that

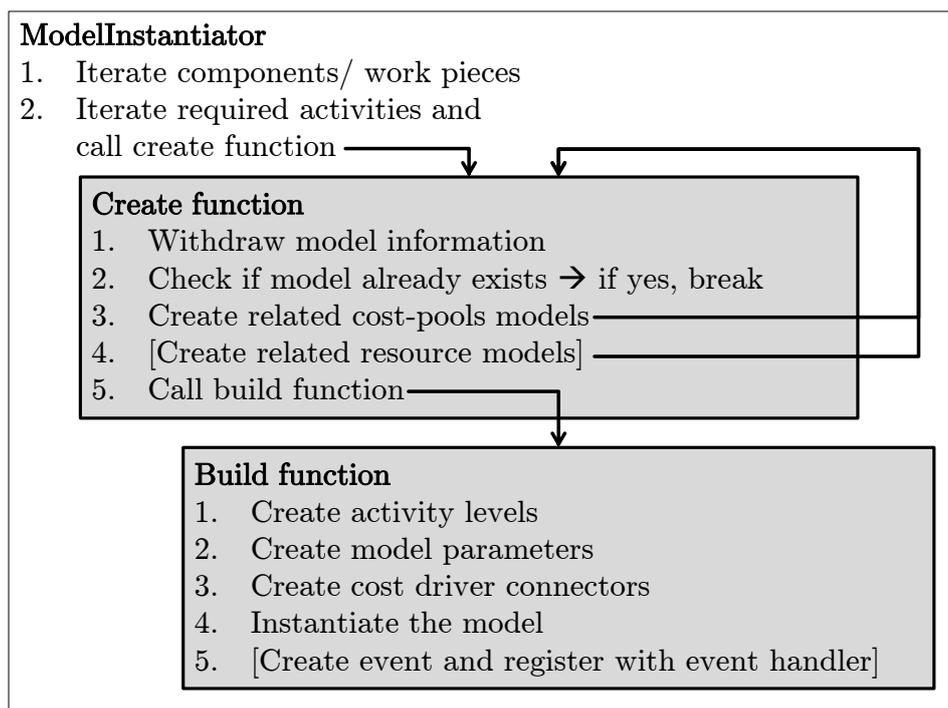


Figure 6.5 Called functions and their tasks during instantiation of activity models.

are event-triggered, i.e. activity and purchase cost models, the mechanisms for event handling (see also following section) are created.

Typically, during instantiation of the cost-benefit model, ambiguous information needs to be resolved. For this purpose, an uncertainty management component has been implemented. The uncertainty manager is implemented as a static class that can be called by all software components. The uncertainty manager stores applied resolution strategies for uncertainties to reapply these strategies in case an uncertainty already encountered arises in the same context.

During instantiation of the cost-benefit model, relations between cost drivers and activity levels have to be found. Subsequently, the completeness of the cost relations in the model has to be checked. For example, if a model requires a certain input which is not produced on the higher level, the model will not yield trustworthy results. The matching of cost drivers with model inputs and activity levels with model outputs is carried out through the input type of the contained information. The input type refers to a semantic description that annotates all cost drivers and activity levels in the knowledge base and allows to identify quant-

ities that are mutually compatible. This approach restricts the models to use only one input or output of the same input type. Furthermore, a sufficiently fine semantic grounding for activity levels and cost drivers is required. This semantic grounding allows to distinguish related information depending on its context. Examples for process-related input types are length of weld seam, path feed rate, and weld seam volume. The semantic grounding is collected in the knowledge base and queried during model instantiation.

Handling of units

As outlined before, the relation of cost drivers and activity levels are established by matching input types. Different models might produce their input in different units depending on the implementation of single components. Therefore, means for unit conversion are required in order to produce an executable model without the need for manual intervention.

For this purpose, all quantities refer to a semantic description of units contained in the knowledge base. Unit conversions are contained in a separate class. The unit conversions specify a left side unit X_l , a right side unit X_r , an offset ω_U and a scaling factor f_U . The conversion is defined as

$$X_l = f_U X_r + \omega_U. \quad (6.2)$$

In case the instantiator discovers activity levels and cost drivers which are compatible in terms of input type, but not in terms of unit, it searches for an appropriate unit conversion and generates a unit conversion element implementing equation (6.2) or the inverse equation solved for X_r . The unit conversion elements are derived from the same ancestor class as cost drivers and can be inserted into the delegate-based calling scheme between activity level and cost driver. This type of unit conversion can only be applied to linear conversions. It allows to connect the different activity models even if their units mismatch. More elaborate research on semantic unit descriptions exists (Hodgson et al., 2011) and could be integrated in the future.

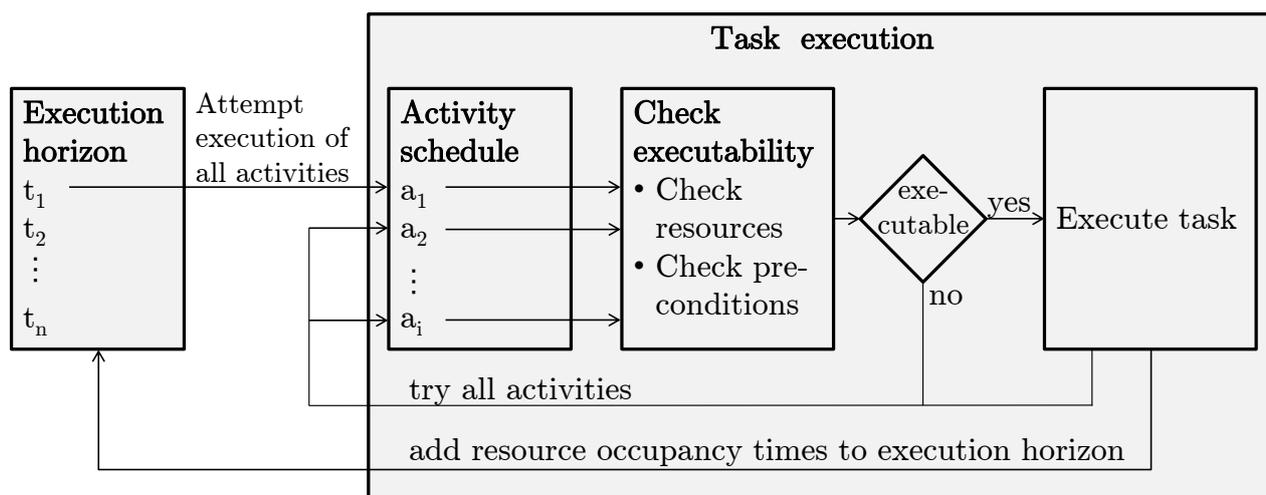


Figure 6.6 Iterative process for cost-benefit model computation.

Model computation

The instantiated cost-benefit model constitutes a description of the cost-benefit relations for a specific robot application that can be used to compute different scenarios. This is done by computing the cost-benefit model. Computation refers to stepping through the activity schedule and simulating the efforts and cost resulting from the required activities. Details of the scheduling required for executing different tasks can be found in Section 4.2.6.

The model is computed by trying to process the task that ranks highest in the task schedule. If all required resources can be allocated and all precondition tasks have been completed, the task is executed and the processing times of all resources are stored in an execution horizon. The execution horizon lists times in the life-cycle of the model when resources become available after completing a task, i.e. points in time where the occupation state of the resource pool changes and at which the execution of additional tasks might be possible. Then, the executor iteratively moves through all tasks in the schedule and checks their executability, i.e. if all preconditions are fulfilled and if all resources are available. After this, the execution horizon is ordered with respect to the next change in the resource pool and the task execution process is repeated at this point in time. Figure 6.6 shows a schematic of the computation process.

For the computation, the handling of events related to the start of production activities and the change in resource allocation are of high importance and explained in more detail below. These events allow to synchronize aspects of the model and start the execution of activities.

Managing events Activities shall only be carried out and produce efforts in case the related product is actually produced or some related activity is actually carried out, i.e. assigned to a resource. The evaluation of the activity models for activities and purchases are therefore triggered through an event mechanism that allows to activate and deactivate the computation of these activity models. This allows to activate these models only when required. For activity models, an event causes the computation of the functions of the activity models and the assignment of the resulting output activity levels. This ensures that activity models are only evaluated in case the respective work activity is carried out at the moment, i.e. when a product causing this activity is currently produced. The cost-pool models related to equipment purchase are only activated once when the effort causing product is produced to reflect the cost for purchase of components or raw material.

The activation and deactivation of models is triggered through the so called event manager. Process and cost-pool models register with the event manager when they are created. The event manager is a static class which stores relevant events and delegates to the activation of the related models in a list of so-called event handlers. The event handlers store delegates that allow the activation of the targeted activity models by setting an internal activation flag in the model. For example, all related production activities and the cost-pool model for purchase of raw material are triggered in case the event for production of this product is triggered.

Figure 6.7 schematically shows the computation pipeline of the cost-benefit model through the different PPRC-levels from product to cost-pool. The figure shows how the event handlers act on the purchase cost-pool models and process models in order to control the activity execution in the pipeline.

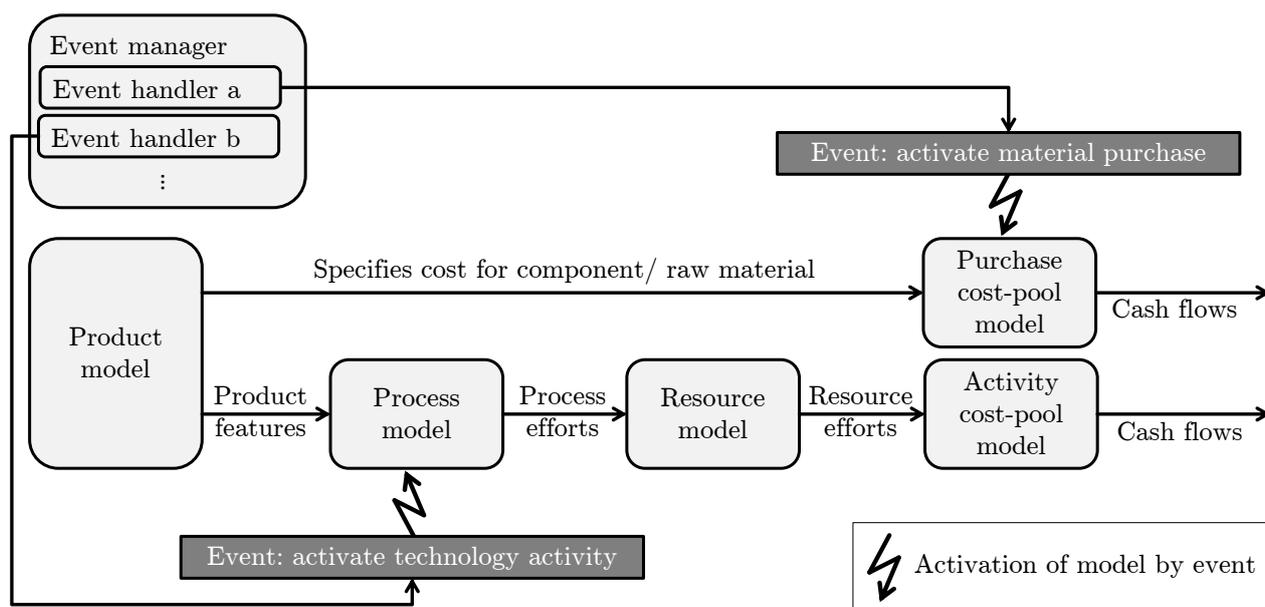


Figure 6.7 Computation pipeline and handling of events.

Managing resource allocation Resources play a particular role in the cost-benefit model as resources relate to physical assets and not only to semantic concepts describing work tasks, activities, or cost. Hence, resources have limited cardinality and limited capacity and therefore control the speed with which activities are processed in the model. The resource manager ensures that only as many activities are assigned to a resource as this resource is able to process simultaneously. It is a static class. The resource manager administers a list in which all resource models are registered when they are created during model instantiation. It offers functionality for determining during model computation whether all required resources for a particular activity are available and can be allocated to the activity or not. Given a list of required resources, the resource manager can therefore answer whether a certain activity, for example the production of a new part, can currently be executed. In case all required resources are available, the resource manager can compute the task by executing the related resource models through the event manager.

6.1.3 User interface

Graphic user interface

As outlined in Chapter 4, interaction with the user is required in order to complete missing information and resolve ambiguities during the matching of local system description and the knowledge base. Hence, the interaction of the user with the software is a central part of the developed cost-benefit assessment tool.

The cost-benefit assessment software uses a ribbon style user interface to present different contexts for interacting with the model. This user interface and the main interaction areas are shown in Figure 6.8. The user interface consists of three main areas. The ribbon pane allows the user to load AutomationML files from plant design and to specify cost-relevant information required for the model such as interest rates, shift model, purchase price of the robot system, and company size. The ribbon pane is only used for information that can be considered a precondition for the execution of the model. An additional ribbon for managing the knowledge base, e.g. for import and export of data, is available. The pane for model execution and results is the main area for interaction with the model. This area consists of several tabs that are used to display information that requires the attention of the user during the model instantiation, the model execution or to input different production scenarios. The main functions are:

Visualization: Different tabs visualize the results of the model. The software offers to display the results in form of an amortization graph (see also next section) for the overall project, from the perspective of the end-user, and from the point of view of the system integrator. Furthermore, the cost distribution over the entire life-cycle and the relation between hardware cost and activity cost can be displayed in separate tabs. Finally, the model allows to display the Gantt chart of integration and production activities.

Resolution of ambiguity: The uncertainty resolution tab is used for requesting user input in case of ambiguous or missing information.

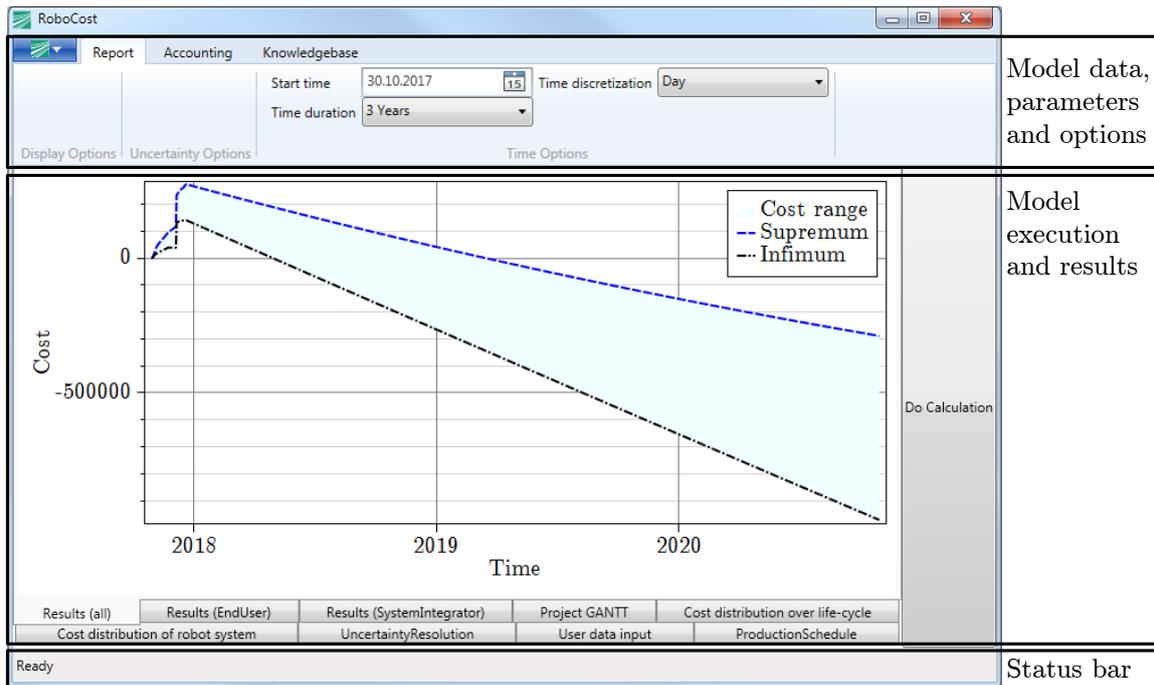


Figure 6.8 User interface of the costing software.

Data input: The user data input tab is used for input of information on the robot system that cannot be extracted from the AutomationML file, for example the classification of the robot system on the Genefke Scale.

Production scenario: The production schedule tab allows to modify the production scenario, i.e. which product is produced in which quantity over the life-cycle of the robot system.

A status bar informs about the current processing status and running activities of the cost-benefit assessment.

Visualization of model results

Results of the cost-benefit model are visualized in an amortization graph implementing the differential cost-benefit assessment introduced in Section 4.2.2. The sum of discounted cash flows accruing since project start up to a certain time horizon is plotted on the ordinate. The time horizon is plotted on the abscissa. The

amortization graph displays the upper and lower limits of amortization computed using interval arithmetics.

To obtain the amortization graph, the cost model is computed for a time interval $[T_{\text{start}}, T_{\text{end}}]$. In order to adjust this time interval, a discretization time interval ΔT_{inc} is introduced and the end time of the interval is chosen as $T_{\text{end}} = T_{\text{start}} + n\Delta T_{\text{inc}}$. Because usually the production schedules for robot systems are planned on daily basis, work days are chosen for discretization interval ΔT_{inc} of the cost-benefit model. This offers a good compromise between time resolution and computational effort. T_{start} is set to the first event of interest, typically the first payment or the first cost-causing activity. The time interval in which the model is computed is then continuously expanded from starting from $n = 1$ to obtain the amortization curve of the robot system. The accumulated cost for each observed interval is computed as the sum of the cash flows accruing in this time interval. These cash flows are discounted to the present time and added to the diagram at T_{end} on the abscissa of the amortization diagram.

The OxyPlot library (Bjorke et al., 2016) is used for graphing. An area series is used to plot the amortization interval of the robot system. Actual plots from the software using the OxyPlot library are shown in Chapter 7 where the evaluation of the costing system is presented.

6.2 Knowledge management

The cost-benefit model integrates data from different sources in order to obtain as much information as possible while minimizing the need for manual input from the user. Three sources of information are used to obtain application-specific data and general information on components, prices, and required work efforts as described in the following.

Robot system design information: The information from PLM plant design tools stored in AutomationML files is used to structure the cost-benefit model. The AutomationML file contains application-specific information.

Knowledge base: The knowledge base is used to store general, application independent information that is used to complement the information from AutomationML file and fill information gaps (see Chapter 4.3).

Human input: As a third source of information, human input is used. This source of information is in particular used if search in the knowledge base does not yield a result or the uncertainty of the result is too high.

Figure 4.8 on page 113 shows the general schematic of knowledge management in the costing system. The matching of information that is specific to the individual robot system with the knowledge base has been described in detail in Section 4.3. The querying required to retrieve information from the knowledge base is handled by a dedicated component for query formation and response evaluation called SPARQL Query Manager. This component can also resolve ambiguities, for example through multiple entries or matches of components in the knowledge base by asking the user. The result of the model instantiator is an executable and parameterized cost-benefit model for the robot system. More details on the model instantiator can be found in Section 6.1.2. In the following, the implementation of the handling of AutomationML data, the implementation of the knowledge base and the user interface for retrieval of user inputs are outlined.

6.2.1 AutomationML data management

AutomationML data is parsed using the AutomationML engine supplied by the AutomationML consortium. The fundamental relations of AutomationML information are formalized in the knowledge base and used as vocabulary for the resulting RDF description. The transcription process is carried out in an explicitly coded component.

All concepts and relations in the AutomationML file are translated into RDF. In this process, also all cross relations through the URI mechanism of AutomationML are checked for consistency and created as object restrictions in the resulting RDF representation. This, in contrast to the AutomationML engine, results in the advantage that links between elements can be followed and directly addressed

in queries. The unique identifiers of internal elements of the AutomationML file are retained in the conversion process allowing to trace back information to the original AutomationML document. The handling of the RDF description and information retrieval from the knowledge base are handled in the library `dotnetrdf`, which offers an easy way of creating, manipulating, and querying RDF structures in .NET.

6.2.2 Knowledge base

The knowledge base is hosted on a Sesame triple store running on an Apache Tomcat server. The knowledge engineering of the ontology is carried out using the software tool Protégé. For the import and export of component data, a specialized interface has been programmed as add-on to the cost-benefit model client that allows the management of the knowledge base. Through this management console, templates for component information can be exported and filled templates can be read back into the knowledge base. The used knowledge base architecture supports multi-user access. This allows multiple users to use the same information. Updates of the knowledge base become immediately available to all users. On top of the currently implemented dedicated knowledge base in Sesame, it is possible to extend the knowledge retrieval to further knowledge repositories through the capabilities of federated SPARQL queries. This is easily possible if the addressed knowledge storage offers an SPARQL endpoint and references the semantic grounding of the RoboCost ontology.

On the side of the cost-benefit assessment software, a query management has been implemented that supplies trunks of SPARQL queries required to withdraw typical information such as class or role relations from the knowledge base without manual composition of the queries. This component also allows to read back the query results into the native data types of the cost-benefit model and hence abstracts the querying process. The key components of the query management and the interaction of the different knowledge sources are schematically shown in Figure 6.9.

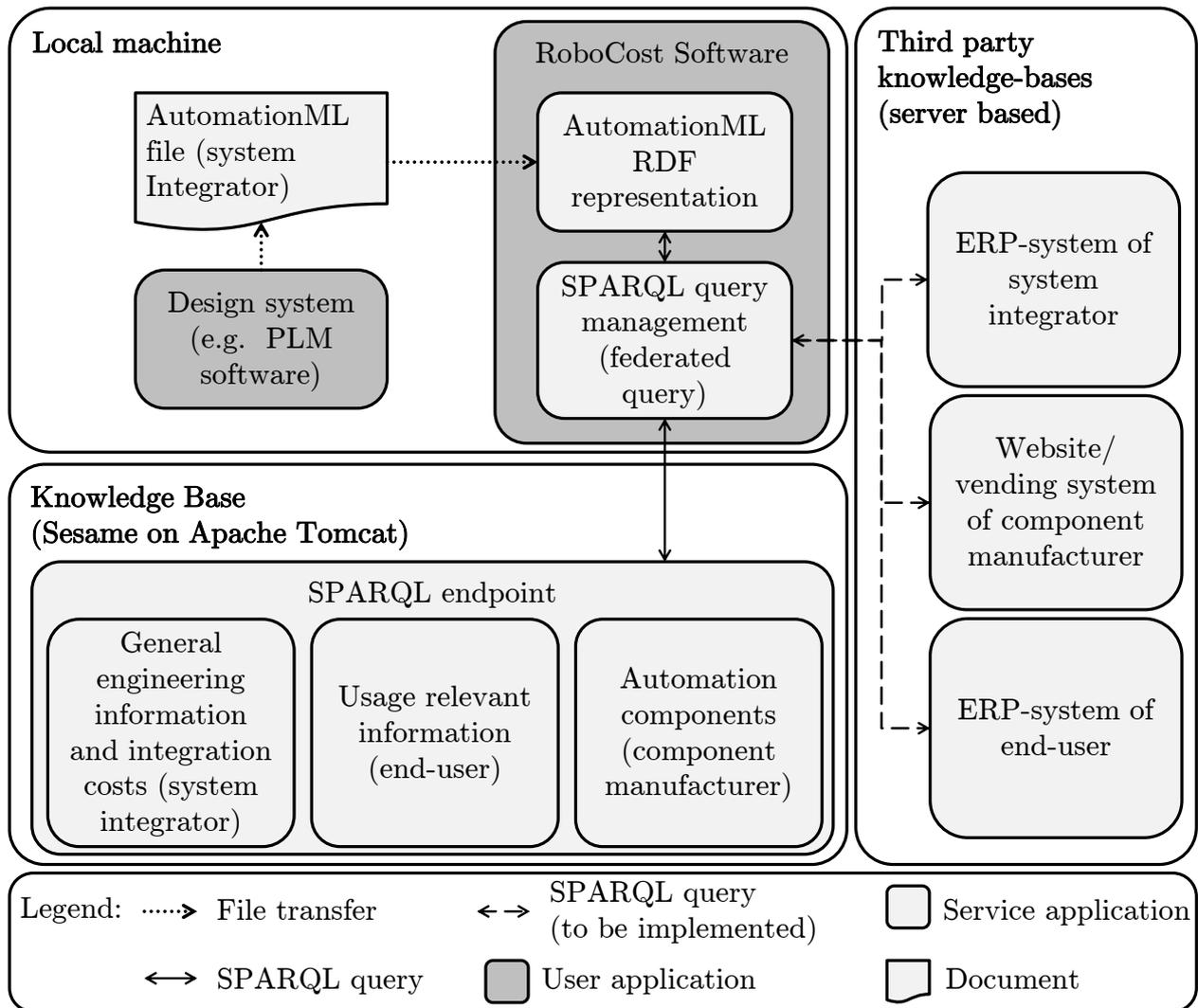


Figure 6.9 Query router and handling of SPARQL queries.

Closely related to the query management is the management of the resolution of uncertainties in the queries. This uncertainty resolution component is triggered by the parsing function of the query answers in the SPARQL query management. Here is the natural place for resolution of association uncertainties or remaining parameter value uncertainties. For example, if a component property is queried, the result might be a single, multiple or no estimate of the component property. While the first case corresponds to no or a consolidated uncertainty in the knowledge base, the second case corresponds to an association uncertainty. In the last case the query did not allow to resolve the parameter value uncertainty. The translation of query results into a consolidated uncertain number to be used by

the cost-benefit model is one of the key tasks of the query parser. The uncertainty resolution component has access to the uncertainty resolution tab of the user interface in order to request input from the user.

All entities in the RoboCost ontology use the common referencing mechanism through URIs in order to unambiguously refer to concepts in the ontology. In order to relate the concepts to real world instances which will possess ambiguous names, all entities further specify human readable names through the *hasName* data property. These human readable names specify differing denominations of the entity and allow the matching of entities from different storages of knowledge and the resolution of ambiguous real world denominations. The URIs in the developed software are managed centrally in a component called URIManager. This component is implemented as static class and can be accessed from all classes and functions. It implements a register of components and their URIs and is therefore capable of identifying run-time elements through their URI. Furthermore, it can generate new, unambiguous URIs.

6.2.3 Knowledge interaction

Interaction with the user takes place in two ways. The user can supply missing knowledge through the completion of automatically generated XLS spreadsheets. This way of input relates to the specification of information that was missing in the knowledge base for the purpose of improving the information content of the knowledge base. The second way of inputting information relates to the data that is required in the context of the analyzed robotic application but is not stored in the knowledge base. This relates in particular to the definition of the production schedule of the machine and the resolution of ambiguities that cannot be resolved automatically with the information contained in the robot system's AutomationML description and knowledge base. For example, certain activities during integration can either be carried out by the end-user or by the system integrator. In order for the cost-benefit assessment software to use the correct costing information, it will ask the user to specify the company or at least

type of company that will carry out a specific activity. Uncertainty resolution is integrated into the GUI of the costing system (see Figure 6.8) as separate tab behind the results pane that are brought to the front in case the resolution of uncertainties is required by the user.

The role library for AutomationML is generated from the roles stored in the knowledge base. The role library is exported from the knowledge base by the cost-benefit assessment software as AutomationML compliant XML-file. For this, the cost-benefit assessment software queries the knowledge base for all roles and generates the XML-file with this information.

7 Evaluation

In this chapter, the developed method for cost-benefit assessment of robot systems is evaluated. This includes the discussion of typical model results for the examples presented in Section 3.1. First, the amortization behavior for the GMAW use case is presented. Subsequently, possible scenarios of the amortization behavior are discussed. The sharing of cost and risk between the system integrator and end-user as key stakeholders in the realization process are explained. Finally, the ability of the cost-benefit model to plausibly schedule work activities is scrutinized. A comparable analysis is subsequently carried out for the handling use case. The results are analyzed with respect to plausibility and in relation to experience from realization projects. Finally, the integration of the costing method into the design process for robot systems is discussed based on the usage experience.

7.1 Analysis of model behavior

The cost-benefit model uses the activity models introduced in Chapter 5. It is assumed that two GMAW welded products are produced with the robot system. These products are shown in Figure 7.1. The parameter values used in the evaluation of the GMAW use case can be found in the Appendix 4.1.

7.1.1 Typical amortization behavior

The amortization graph introduced in Section 6.1.3 provides a good starting point for the analysis of the model performance. This amortization graph includes all



Figure 7.1 GMAW welded products used for the cost calculation: excavator bucket (left) and packing roller (right).

relevant aspects as it represents the cost and time frame for the realization and the usage of the robot system in production in one single figure.

Figure 7.2 shows the amortization behavior for the GMAW use case. This graph accumulates all efforts that occur during the life-cycle of the robot system, i.e. it fuses the system integrator and end-user view into one graph. The figure includes cash flows of types 1 and 3 as introduced in Section 4.2.2. Therefore, it neglects profit and risk sharing between system integrator and end-user and allows a statement regarding the overall cost effectiveness of this robot system.

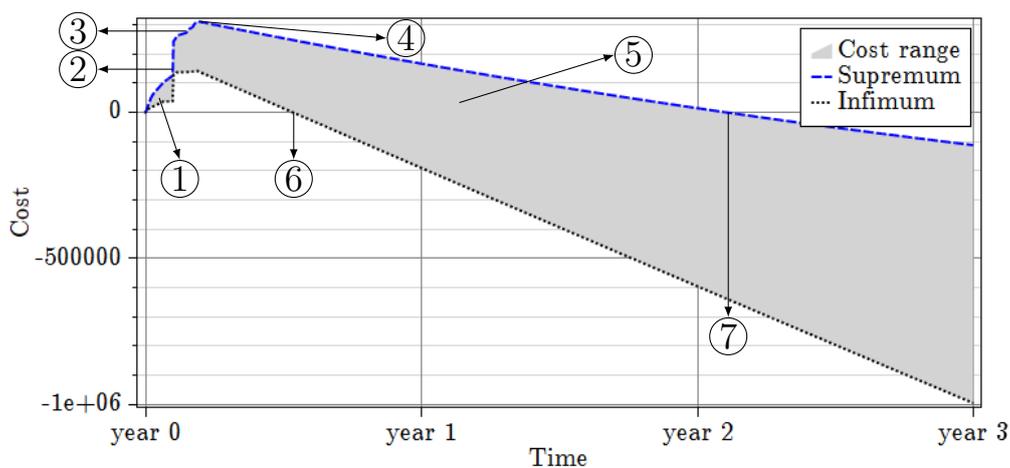


Figure 7.2 Simulated amortization behavior of the welding use case.

At the beginning of the life-cycle (see label 1 in Figure 7.2) the efforts for the

concept phase can be seen, i.e. efforts for performance specification, functional specification and feasibility testing, and the early processes of the realization phase, such as the detailed layout design. Subsequently, the cost increases steeply due to the procurement of the system components (label 2). Further efforts are required for integration, commissioning and ramp-up (label 3). This leads to the maximum of the cost curve (label 4), which represents the realization cost¹. The overall realization effort of the system is [141 000, 311 000] EUR, which is in alignment with the project experience of the analyzed use case. Subsequently, the robot system starts to amortize itself by reducing the labor cost compared to manual process execution (label 5). For the welding use case shown in Figure 7.2, the break even for the overall system is predicted to occur in the time interval [212, 768] days (see labels 6 for best case and label 7 for worst case). As can be expected, the width of the return on investment funnel increases over time as more and more uncertain cost items are taken into account and discounted to the present day as time advances.

Amortization views of end-user and system integrator

As already outlined, the amortization graph shown in Figure 7.2 neglects the transaction between system integrator and end-user and provides an overall statement of the cost and benefit of building and operating the robot system. Further analysis allows to assess the profitability from the perspective of system integrator and end-user individually.

Figures 7.3 and 7.4 show the separated views on the project of system integrator and end-user. The details of this separation have been introduced in Section 4.2.2. For this purpose, the conditions of payment and height of down payments between system integrator and end-user have to be input into the system. This input corresponds to the contractual agreement between system integrator and end-user, in particular the price of the system, which is project-specific and contains the profit of the system integrator.

¹Neglecting realization efforts occurring during the amortization phase, such as performance optimization, etc..

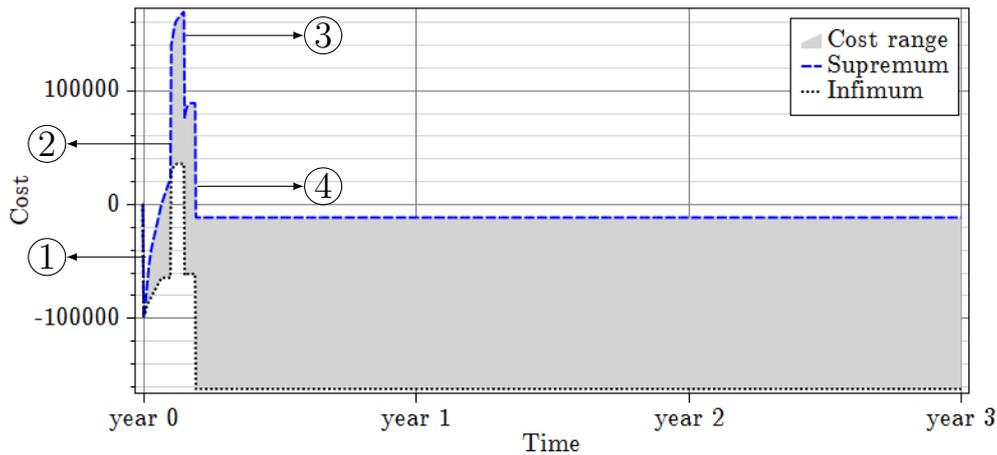


Figure 7.3 Amortization behavior of the GMAW welding use case from the point of view of the system integrator.

Figure 7.3 shows the project from the view of the system integrator. As can be seen, the system integrator carries most integration efforts. Also, the payments for equipment in early project phases accrue on the side of the system integrator as the system integrator has to purchase the components for system integration (see label 2 in Figure 7.3). The down payments by the end-user constitute income on the side of the system integrator. The down payments can be seen in labels 1 (pre-payment), 3 (factory acceptance test payment) and 4 (final payment). Therefore, this figure allows to assess risk and profitability of the robot system on the side of the system integrator. As the system integrator does not benefit during the usage phase of the robot system except for service activities, the amortization curve for the system integrator is concentrated in the realization phase.

Figure 7.4 shows the view of the end-user on the project. The main cost items here stem from payments to the system integrator (see labels 1, 2 and 4 in Figure 7.4). Some of the integration cost, e.g. commissioning and ramp-up accrue directly at the end-user (see label 3). The amortization of the investment does only take place during production. Hence, the interesting part of the amortization curve for the end-user is the usage phase of the robot system, where cost and benefit of the system are weighted against each other (see label 5). The projected return on investment for the end-user for the welding use case is in the time interval [340, 802] days (see labels 6 and 7). Note that the projected return on investment for

the end-user is later in time than for the overall return on investment as the robot system also has to “earn” the profit of the system integrator in this view.

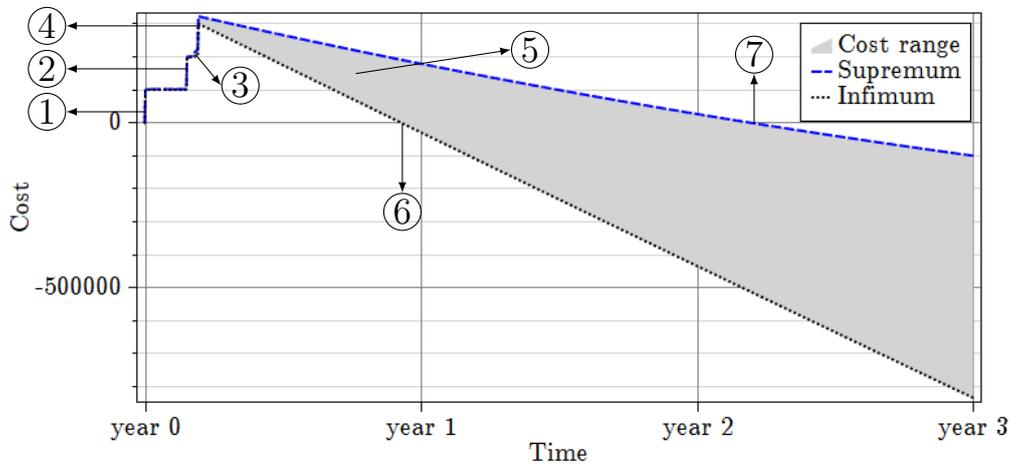


Figure 7.4 Amortization behavior of the GMAW welding use case from the point of view of the end-user.

Amortization scenarios

In the following, different scenarios for the amortization of robot systems are discussed by modifying parameters in the GMAW welding example. This allows to reflect on the ability of the cost-benefit model to accurately reflect and represent different amortization scenarios that might occur in reality. It also increases the confidence that the causal relations in the cost-benefit model are sound.

Regular amortization Regular amortization as shown before in figures 7.2, 7.3 and 7.4 is characterized by a steady increase of cost in the realization phase. At the end of the realization phase, the uncertainty should not be excessive. The amortization graph should then decrease monotonously in the usage phase until amortization. During this decrease, the associated uncertainty should slowly increase due to uncertainties in actual resource consumption by the robot system, uncertainties in cost benchmarks and interest uncertainties. The width of the amortization time interval is the temporal uncertainty of the point in time when break-even is reached. The width of the amortization interval should not be larger than one year for most robot systems in order to allow for a well-founded

evaluation of the cost-effectiveness of the robot system. The amortization graph of overall system and end-user should look similar with a slightly longer amortization time frame and a lower amount of uncertainty on the side of the end-user.

Depending on the specific project and its parameters, the amortization graph might look different from this nominal shape. This can be the result of a number of reasons. In the following paragraphs, different deviations from the nominal amortization shape are described. These paragraphs also point out possible reasons for these deviations, implications on the assessment of cost and benefits and possible resolution strategies.

Pricing mismatch: overpricing of robot system Pricing mismatch occurs when the price for the robot system or payment modalities agreed between system integrator and end-user do not correspond well with the actual efforts on either side. Figure 7.5 shows the amortization graphs of system integrator and end-user in case of overpricing. The plots were created by increasing the system price paid by the end-user from € 300 000 to € 500 000. In case of overpricing, the amortization curve of the system integrator is deeply in the profit zone. The amortization time frame for the end-user is significantly longer than to the overall system. This is the result of the system integrator reaping a large portion of the profits resulting from automation. Only in few cases, where the realized robot system exhibits excellent cost savings, will the project still be economically sound for the end-user in case of strong overpricing. The overall amortization curve does not change compared to Figure 7.2 as it is agnostic to the transaction of system integrator and end-user.

Pricing mismatch: underpricing of robot system Figure 7.6 shows the amortization graphs of system integrator and end-user for underpricing. To create underpricing, the system price paid by the end-user was decreased from € 300 000 to € 150 000. In case of underpricing, the amortization graph of the system integrator at the end of the project considerably intersects with positive figures. This means there is a significant risk for loss on the side of the system integrator

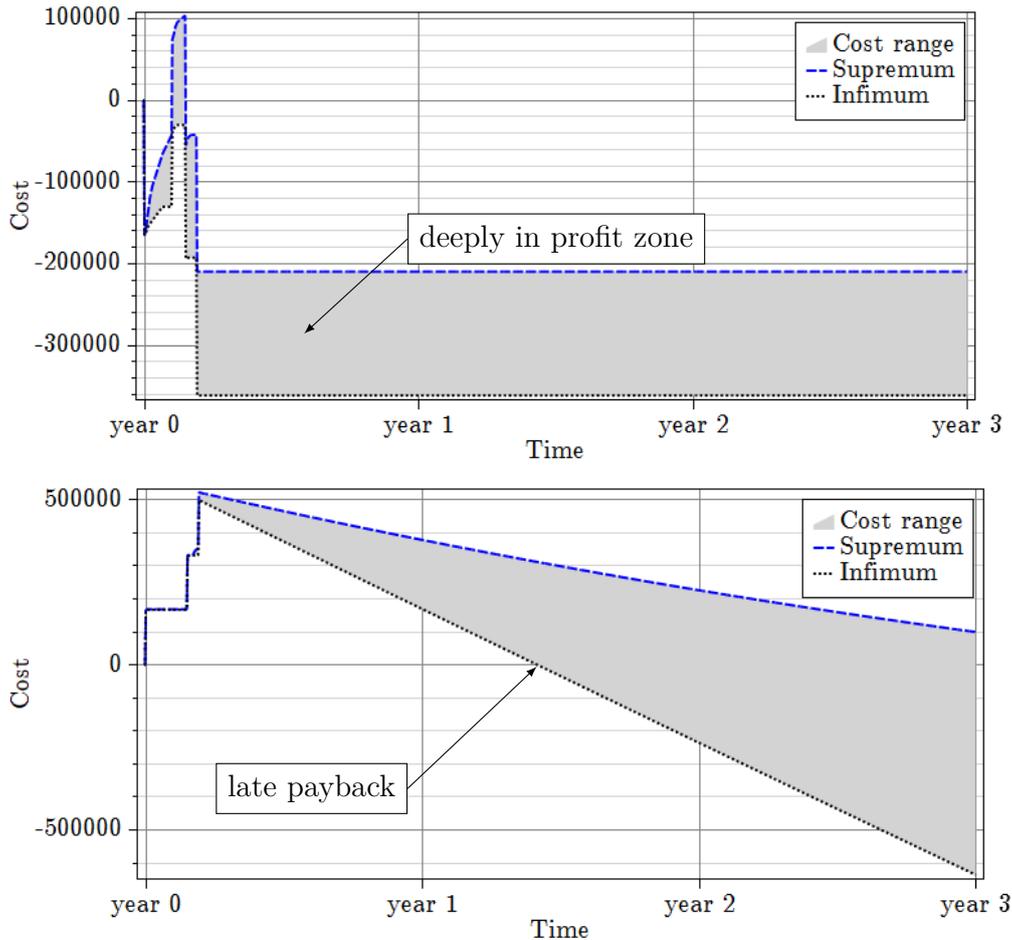


Figure 7.5 Amortization curves of system integrator (top) and end-user (bottom) for overpricing.

at the end of the project. As automation projects typically involve a significant realization risk, it is not uncommon that the supremum of the amortization graph of the system integrator is positive. This corresponds to the situation with a worst case chance of loss on the project for the system integrator. As the system integrator also profits from servicing the robot system, the project might still be profitable, even if the worst case scenario materializes. However, if the figures are too far on the positive side, this will harm the system integrator. Sometimes, system integrators accept underpricing or knowingly underprice robot systems due to strategic considerations. For example, if the robot system is a pilot installation, the system integrator might hope to profit from follow-up realizations that are sold at the same or similar price. Or, the system integrator might accept a project even if the project is not profitable in order to ensure a high degree of capacity

utilization of his organization and generate a contribution margin to fixed costs. Hence sometimes, system integrators knowingly accept a project loss in hope of future profits. Also for underpricing the overall shape of the amortization curve is unaltered and shown in Figure 7.2.

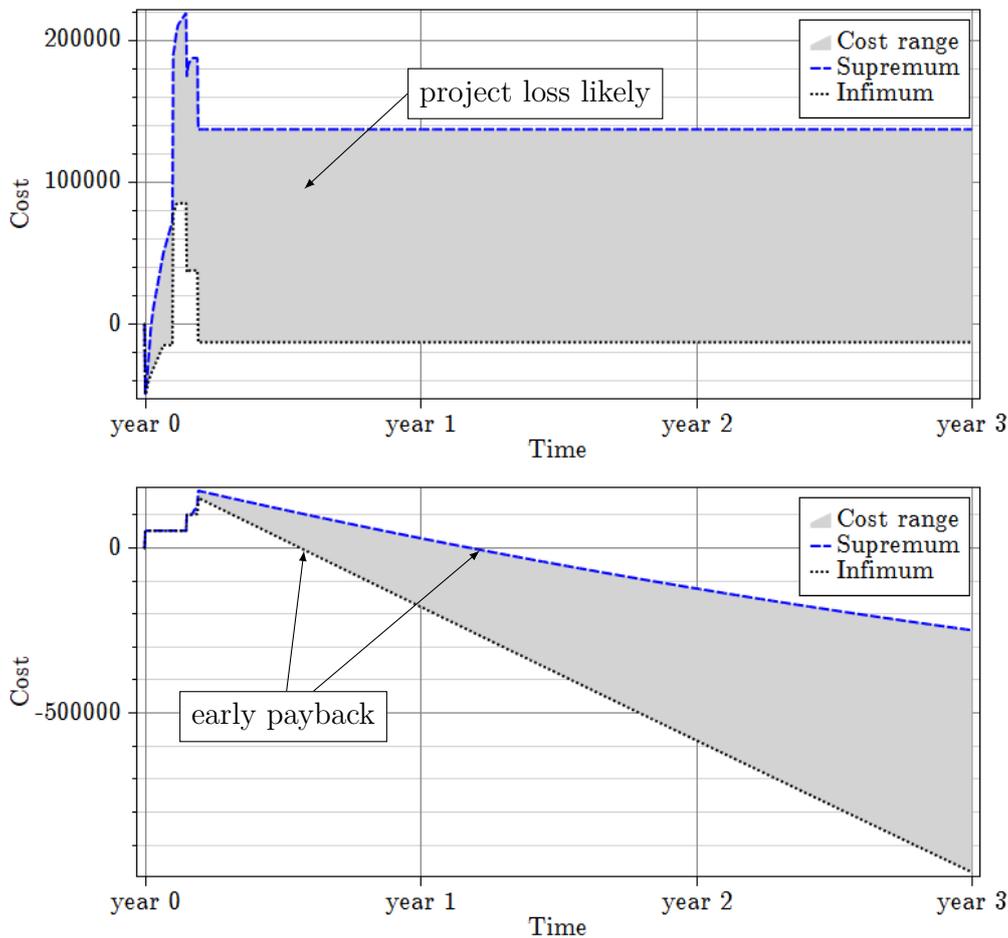


Figure 7.6 Amortization curves of system integrator (top) and end-user (bottom) for underpricing.

High realization risk High realization risk can be witnessed by a high level of uncertainty of the cost graph at the end of the integration phase (see Figure 7.7). This figure has been created by tripling the uncertainties of all purchase prices of system components, i.e. increasing the width of the uncertainty interval threefold. This case signifies a high level of uncertainty regarding the actual effort for the successful realization of the robot system in terms of required activities or investments. The associated risk stays with the system integrator as usually

a fixed price for the purchase of the system is agreed between system integrator and end-user. This can be witnessed by comparing the amortization of end-user and system integrator as shown in Figure 7.8. Realization risk is typically high for new types of robot systems (i.e. high number on the Genefke Scale). In case the realization risk prevents making a meaningful assessment of the cost effectiveness, it might be useful to include an experimental study or simulation of the robot system in order to gather more data and thus possibly reduce the level of uncertainty in the realization estimates.

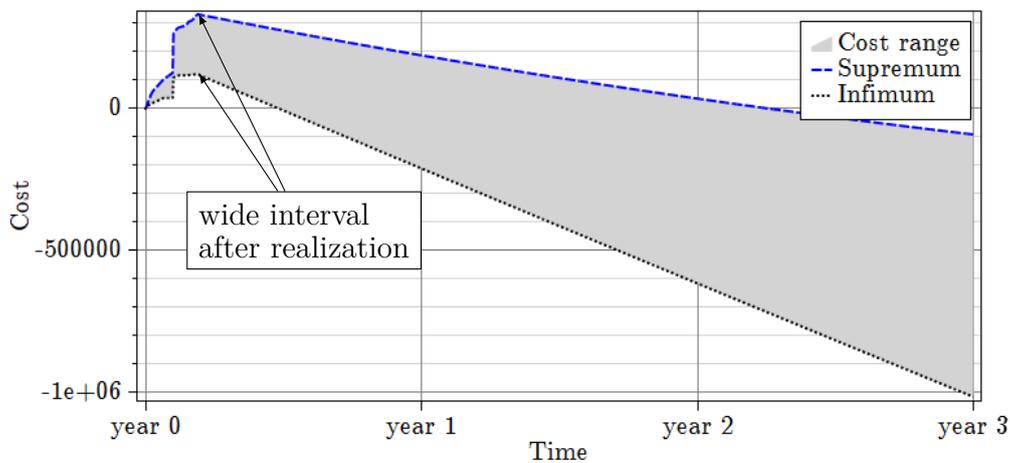


Figure 7.7 Amortization curve for the case of high realization risk (overall amortization curve).

High usage risk High usage risk becomes evident in a strongly widening cost funnel in the realization phase. This yields an extremely wide amortization interval as shown in Figure 7.9. Here the associated uncertainty for the benchmark cost has been increased three times, i.e. the interval width for the benchmark cost has been tripled. This effect has two potential causes. Either, the actual production cost of existing products is not exactly known or the actual usage cost of the robot system has a high level of uncertainty. The former cause hints that a more detailed analysis of existing processes is required before investing in automation. This is highly advisable as this analysis might also yield measures to realize efficiency gains in the existing production as cost causes are not exactly known today. In case the uncertainty cannot be reduced by this detailed analysis, it is likely due to fluctuations in production performance. In this case, automa-

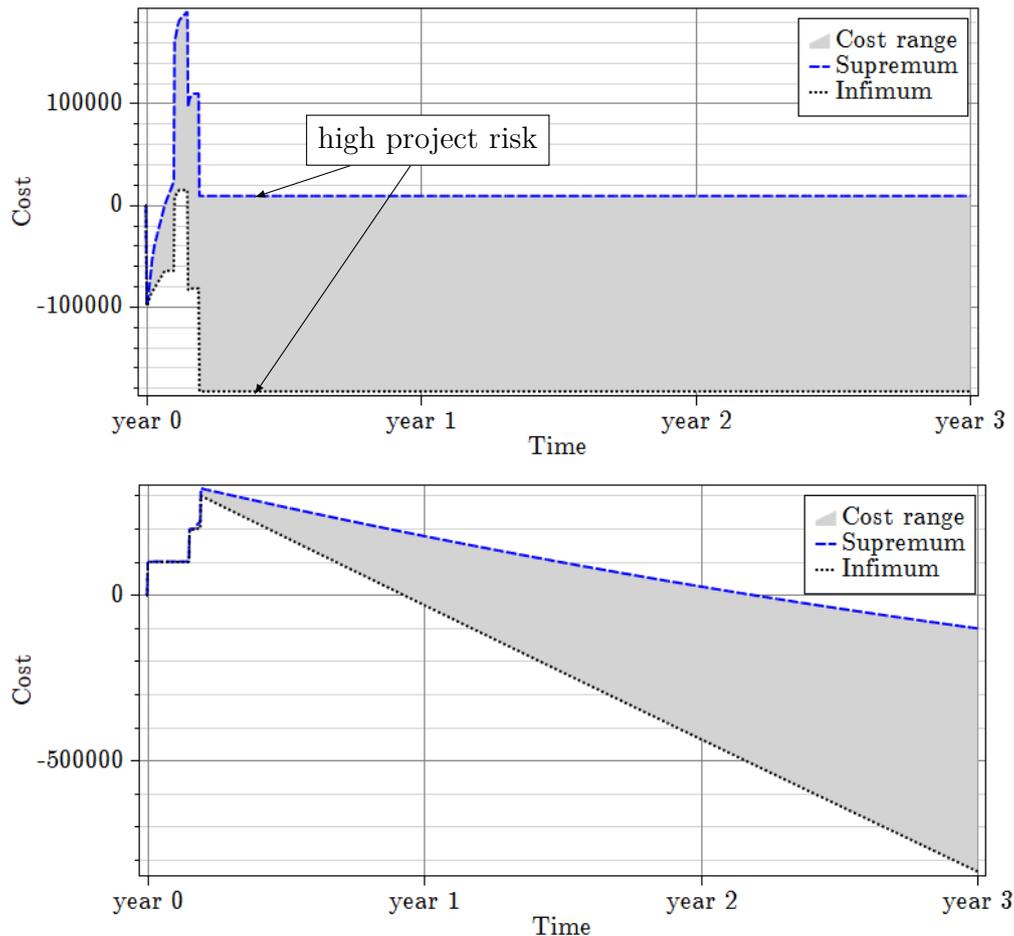


Figure 7.8 Amortization curves of system integrator (top) and end-user (bottom) with high realization risk.

tion might be a useful solution to achieve more reliable production processes. The latter case hints that one or several models in the usage phase cause significant uncertainty. In this case, the models and in particular their parameters should be reviewed to decrease the level of uncertainty and improve the expressiveness of the model. High usage risk only affects the end-user of the system. Figure 7.10 shows the amortization curves for system integrator and end-user. As the system integrator does not participate in the usage of the robot system, the associated risk stays with the end-user.

Project runaway Project runaway occurs when in the worst case the robot system does not produce cost savings in comparison to the benchmark. This

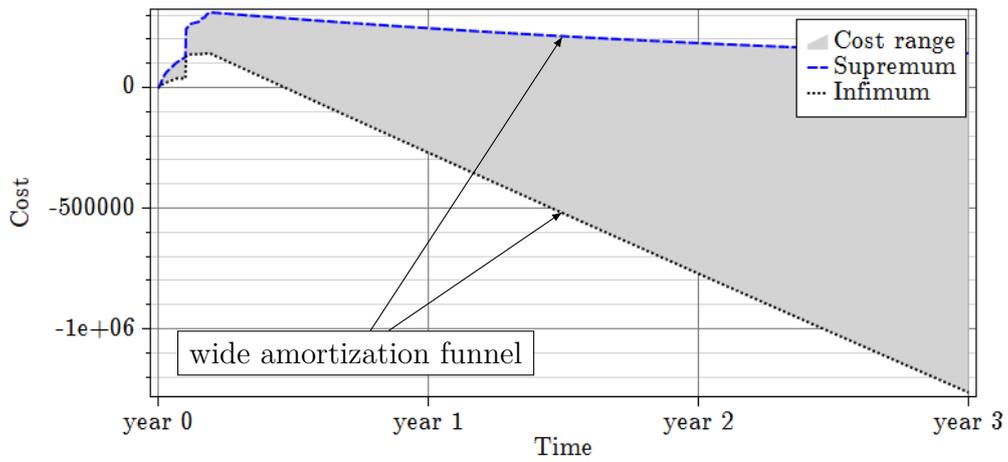


Figure 7.9 Amortization curve with high usage risk (overall amortization curve).

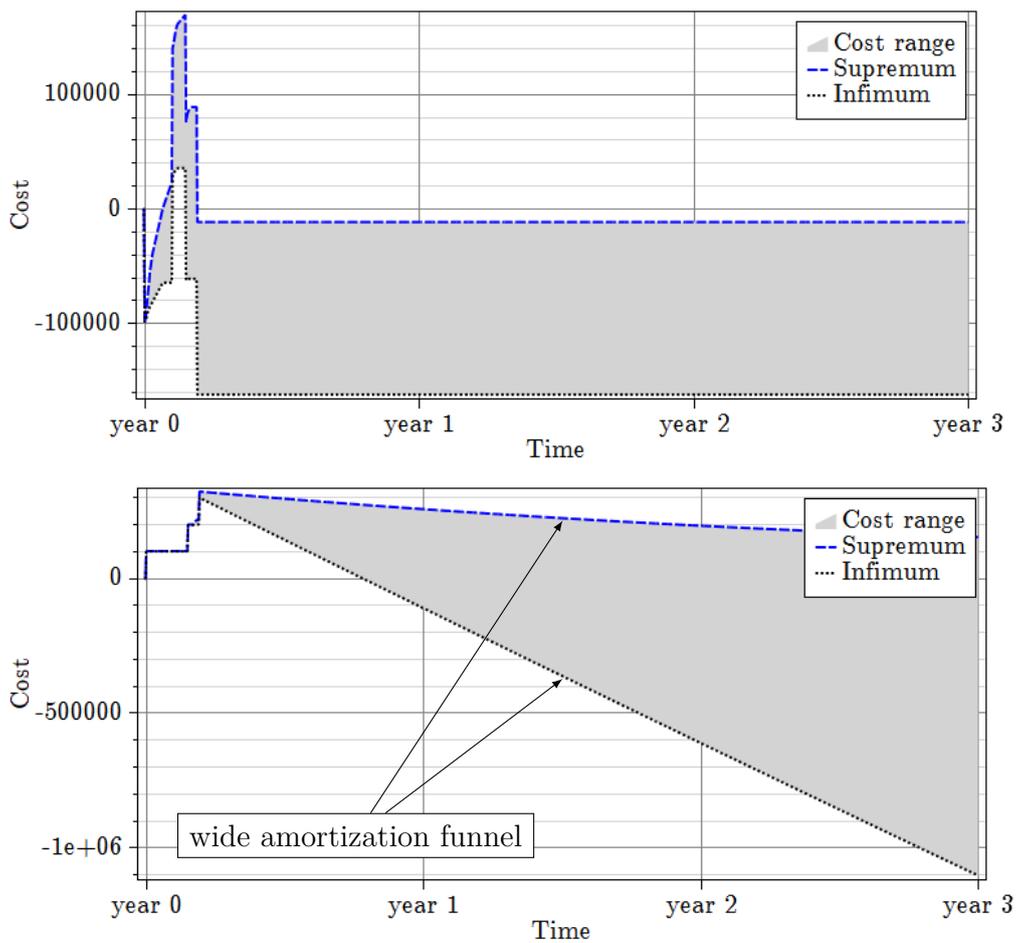


Figure 7.10 Amortization curves of system integrator (top) and end-user (bottom) with high usage risk.

leads to an ever increasing supremum of the cost funnel and hence an amortization cannot be achieved. Figure 7.11 shows this situation for a decrease of the supremum of the benchmark cost per part by € 200. Runaway cost only concerns the end-user in most payment modalities of robot system installations. Project runaway can affect the entire amortization funnel or only its supremum. In case of project cost runaway with a sufficient amortization time frame for the infimum of the cost funnel, a more detailed analysis is required in order to attempt reducing uncertainties on the supremum side of the cost funnel. In case also the infimum points at a critical amortization time frame, i.e. also the best case does not lead to an amortization of the robot system, the usefulness of the project in cost benefit terms should be questioned. Figure 7.12 shows the amortization curves for system integrator and end-user for project runaway. Again, only the end-user is affected by the project runaway as the effect occurs during the usage phase.

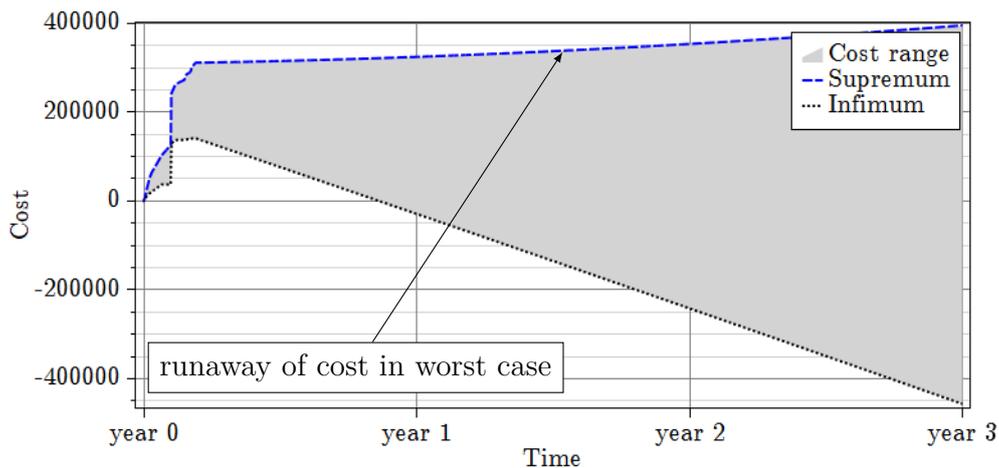


Figure 7.11 Runaway of project cost (overall amortization curve).

Summary of amortization scenarios As can be seen from the introduced amortization scenarios, the cost model is capable of representing typical situations that can occur in the amortization of robot systems. The amortization graph offers an intuitive and easily accessible representation of the cost effectiveness over the entire life-cycle of the robot system. It also allows to isolate the views of end-user and system integrator and represents the risk associated to the project. This is believed to foster cost transparency in the cooperation of the

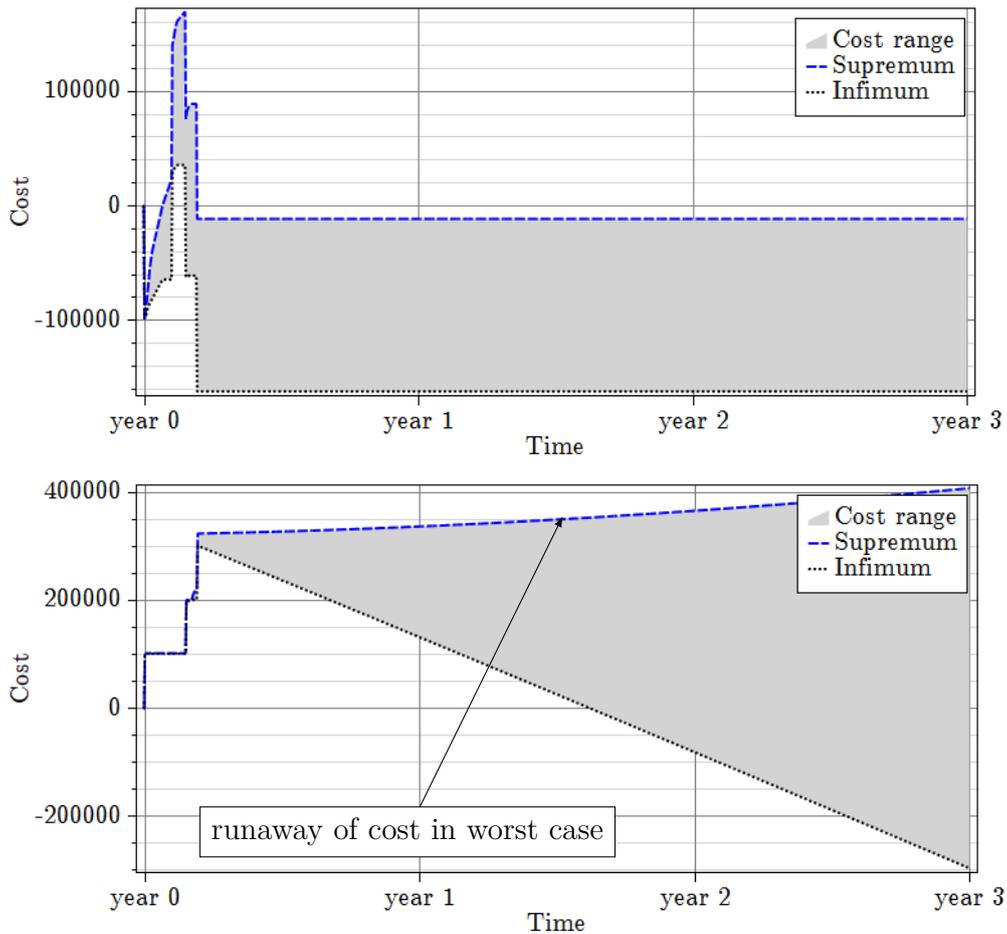


Figure 7.12 Amortization curves of system integrator (top) and end-user (bottom) with project runaway.

different stakeholders, while effectively hiding sensitive information such as profit margins.

7.1.2 Distribution of risk and uncertainties among stakeholders

When comparing the views of system integrator in Figure 7.3 and end-user in Figure 7.4, it becomes evident that the risk reflected in the width of the amortization funnel is distributed highly unevenly between system integrator and end-user. While the risk of the end-user is very low and the amortization time rather certain, the system integrator bears the bulk of the risk related to the system

realization. This uneven sharing of risks results from the typical contractual arrangement in the realization of robot systems and the fact that the realization risk in special machinery is very high. The end-user buys the system at a fixed price from the system integrator. Hence, the realization risk almost solely rests with the system integrator. The risk is then paid-off with a price premium by the end-user. This uneven risk sharing also motivates the system integrator to pass on some of the risk by implicitly or explicitly excluding certain tasks from the contract in order to bill them separately to the end-user. The high focus on the system price as outlined before and the opacity of the overall life-cycle cost is one of the main motivations for developing the cost-benefit assessment tool. The cost-benefit tool increases the transparency of cost and risk for system integrator and end-user. Therefore, it allows an interaction that is more aligned with the actual interests of all stakeholders.

It has to be noted here that the model does not incorporate the indirect effects associated with project risk which are often very relevant for the end-user. These include for example the profitability loss due to decreased performance of the robot system or the production loss due to delayed commissioning. These outcomes are highly project specific and depend on the relation between system integrator and end-user. Consequently, in reality the end-user also carries part of the risk in case the project cannot reach the desired goal. However, also in this case a significant part of the risk is on the side of the system integrator that usually will be obliged to pay contractual penalties. Also, risks for the end-user associated with the bankruptcy risk of the system integrator are not considered in the cost-benefit model. The same applies for the perspective of the system integrator. E.g. the bankruptcy risk of the end-user during the project is not accounted for in the cost-benefit model.

7.1.3 Cost distribution for system realization

In a second analysis, the cost distribution for realizing the system is dissected in more detail. In the following, the relative contribution of single tasks and

investments in equipment to the overall system cost is analyzed. Figure 7.13 shows the cost distribution between the investment in hardware and the cost caused by activities during the development phase of the robot system up to the deployment and ramp-up of production, i.e. excluding operation of the robot system. It becomes evident that for the welding robot a high portion of the cost for realization of the robot system flows into the purchase of components. The share of hardware cost of the overall cost for the realization of the robot system is 76%. This figure is unusually high when compared to typical robot systems. For the welding robot system, this is plausible as the robot system is highly standardized. It can be tailored to customer needs with few order options. Furthermore, it is not connected to an automated material flow. Hence, it can be deployed in many different settings and companies without extensive engineering effort.

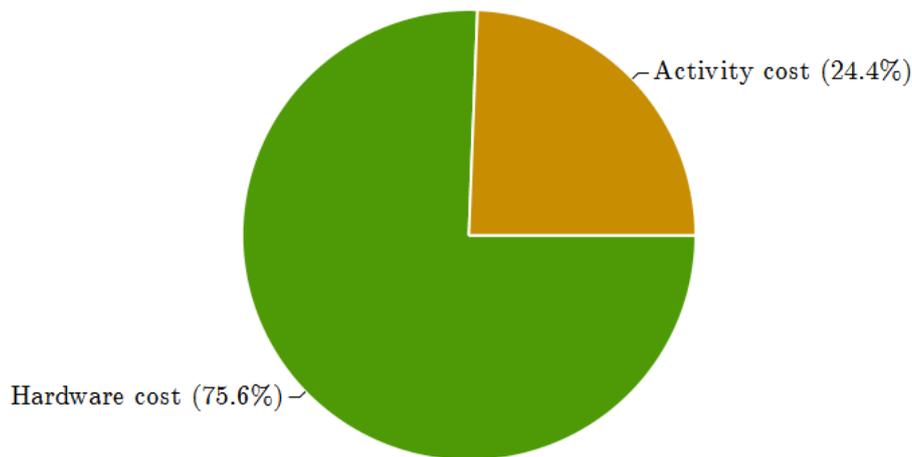


Figure 7.13 Distribution of cost between hardware cost and activity cost for the GMAW use case.

Figure 7.14 shows the distribution of activity cost between the different life-cycle phases during realization. The pie chart lists the relative percentage of cost caused by activities in the phases concept and development, integration, commissioning and ramp-up. As expected, the bulk of activity cost occurs during the integration phase of the robot system. Also here, it can be seen that design and development efforts are very low due to the low customization effort for this type of robot cell. Also, there are no high costs for ramp-up in this type of robot system. The reason is that programming is largely carried out by the end-user. Furthermore, due to the lack of a material handling system, the run-in efforts are moderate.

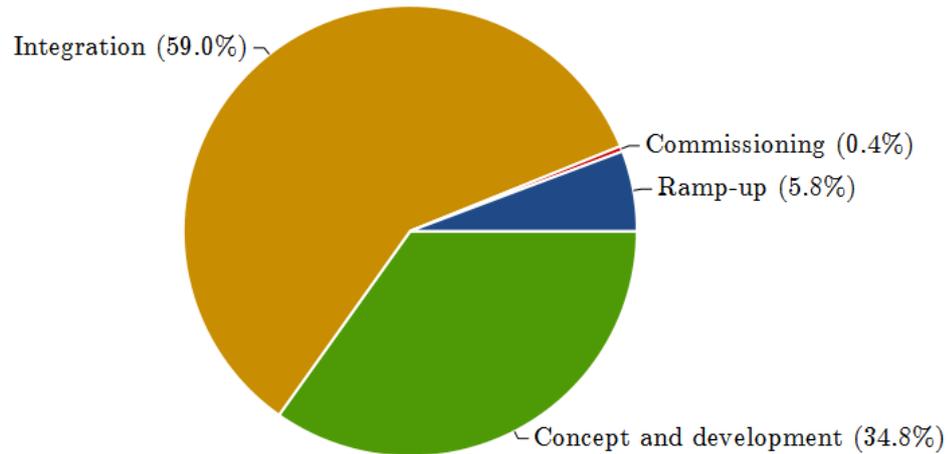


Figure 7.14 Distribution of cost between different life-cycle phases during design and integration for the GMAW use case.

7.1.4 Task scheduling and project schedule

The cost-benefit assessment presented in this thesis relies on the capability of the cost-benefit assessment tool to accurately schedule work tasks during system realization and production usage. The project and usage schedule generated by the cost-benefit assessment software provides another entry point for evaluation of the cost-benefit model's applicability. In the following, the scheduling of the integration and production activities as introduced in Chapter 4.2.6 is analyzed. Figure 7.15 shows the executed project tasks in an auto-generated project Gantt chart.

Due to the resource-based scheduling, the cost-benefit assessment tool is capable of scheduling activities in parallel where this is possible. The sequential order of activities is kept where dependencies between activities exist or the same resource is required by two activities. For example, the logical integration of components is only carried out after they have been mechanically installed. The interfacing of different components is carried out simultaneously as it is done by different experts. The cost-benefit model accurately schedules integration milestones, i.e. FAT and final acceptance testing. The predicted project schedule is in good agreement with experience from similar projects. Hence, it can be concluded that the cost-benefit assessment tool is capable of producing plausible project schedules

7 Evaluation

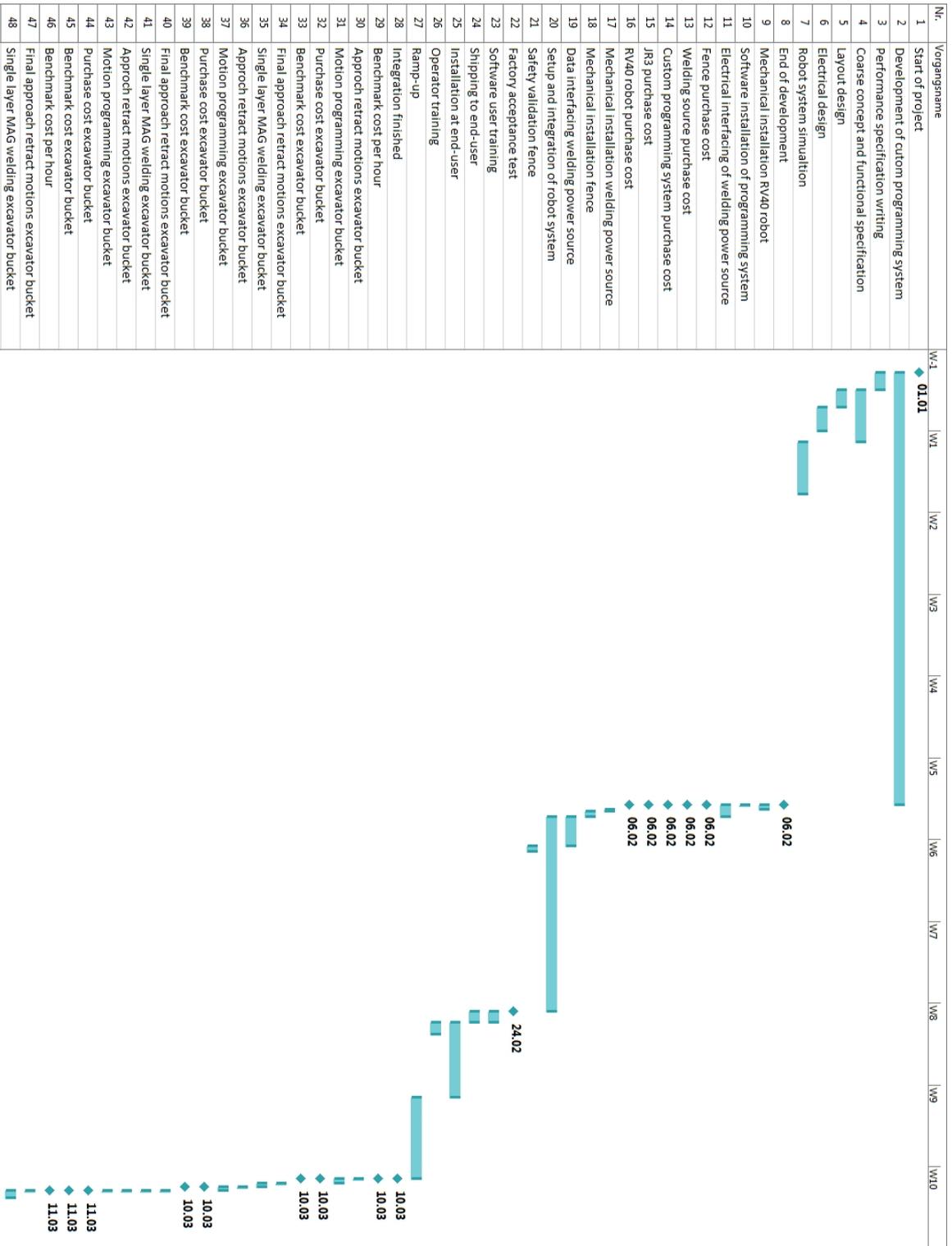


Figure 7.15 Gantt chart of the realization of the GMAW use case.

for the realization of the robot system. Although these project schedules are not suitable for project management and lack refinement in the details, they are useful for the estimation of the cost and the project time frame.

7.2 Model results and discussion for the handling use case

In the following, the model results for the handling use case, which has been introduced in Chapter 3.1, are given. The parameter values used in the evaluation of the handling use case can be found in the Appendix 4.2.

7.2.1 Amortization behavior

Figure 7.16 shows the overall amortization view for the handling use case. The overall system is highly profitable. The break even is achieved after [104, 182] days. The realization cost of the system is € [70 000, 110 000]. Compared to the welding use case, the associated risk with the usage of the robot system is significantly lower. This is the case because the hourly savings in work effort are easier to quantify and independent of the produced quantity. The operation cost of the robot system are almost negligible. Even though the estimated development efforts are plausible, the estimated realization time of the robot system is very low. This is in particular the case, because the cost-benefit model currently does not include delivery times of components and time delays due to project management. This fact becomes particularly obvious for a robot system with low realization efforts as the one analyzed here.

Figure 7.17 shows the amortization view for the handling use case from the perspective of the system integrator and the end-user. With the assumed system price of € 200 000, the amortization cost of the system integrator is clearly in the profit zone. For the end-user, the break even for the investment in the system is reached after [262, 342] days. The difference in the break even time for the overall

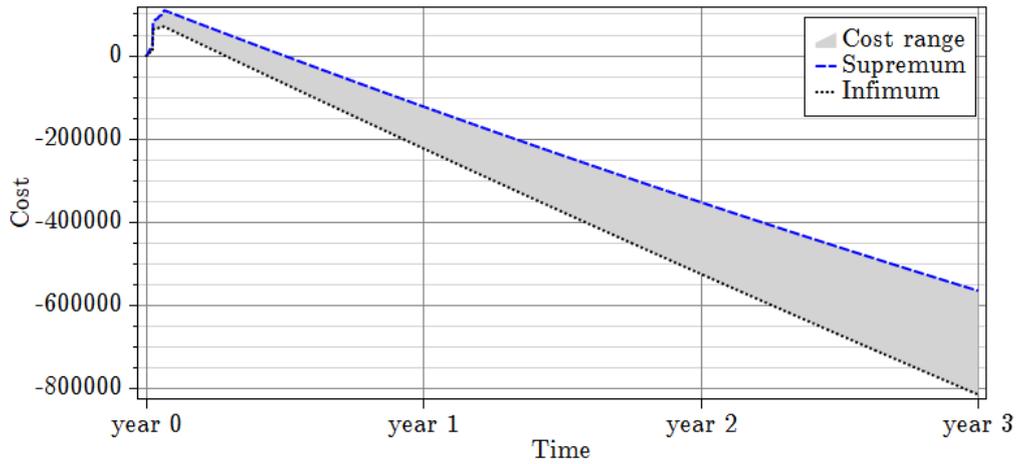


Figure 7.16 Overall amortization view for the handling use case.

robot system and the end-user shows slight signs of overpricing. However, still the robot system is highly profitable for the end-user. Again, the bulk of the realization risk rests with the system integrator due to the fixed price arrangement.

Concluding, the cost-benefit tool produces reasonable forecasts for amortization, efforts and cost distribution for the handling use case. The predicted efforts are in line with experience from past projects. Considering how few information on the actual application is supplied to the cost-benefit assessment software, this allows an assessment of system cost and benefit in very early phases of the life-cycle. In order to reflect more realistic realization times the delivery times of components should be included in the future.

7.2.2 Cost distribution

The cost distribution between hardware investment and activity cost is shown in Figure 7.18. The activity-cost makes up 23% of the realization cost. Compared to typical robot system realization projects this figure is rather low. The high percentage of the hardware cost of 77% stems from the fact that the robot work cell does not require specialized integration tasks and does not require integration

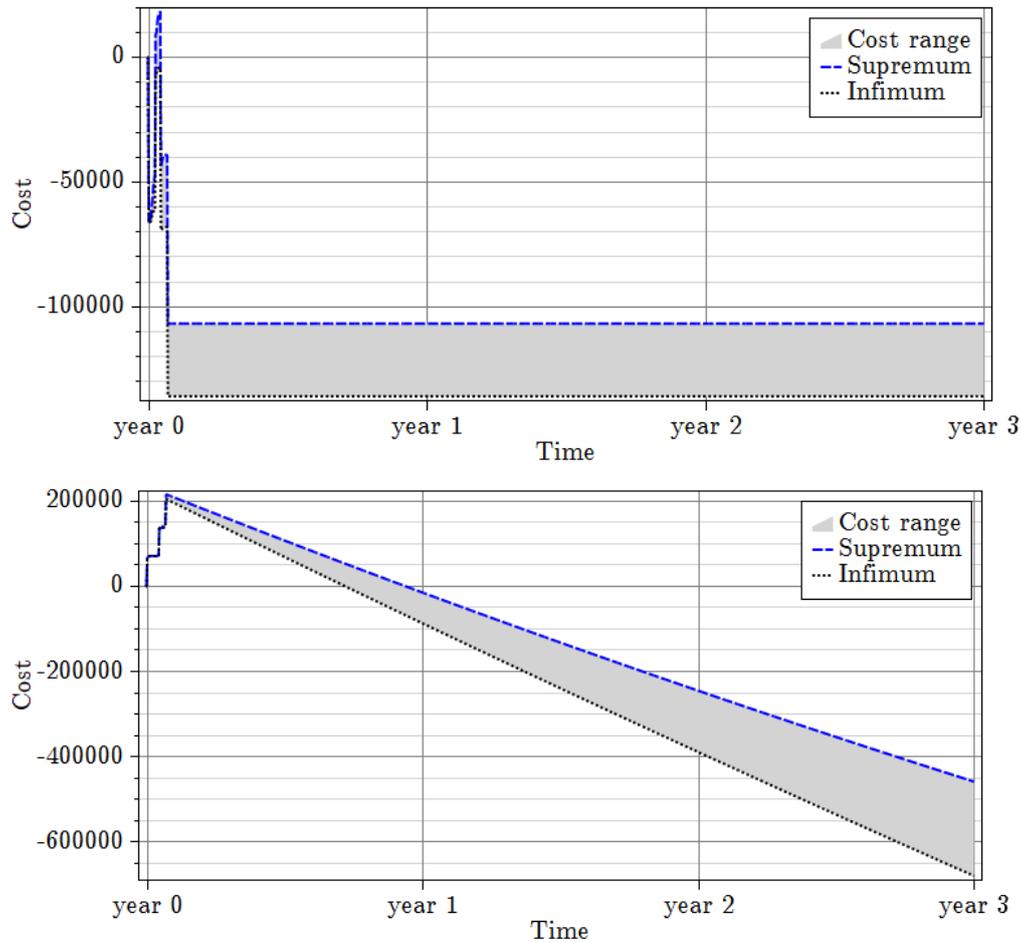


Figure 7.17 Amortization views of system integrator (top) and end-user (bottom) for the handling use case.

into upstream or downstream processes. Only the integration with the existing machine tool causes additional efforts.

Figure 7.19 shows the distribution of activity cost over the different life-cycle phases. As expected, with 75% the bulk of the activity cost accrues during the system integration. Here, many activities are required for installation and interfacing of components. Concept and development activities causes the second largest portion of activity-cost (16%). The cost for commissioning and ramp-up is combined 9%. This cost is the smallest portion of activity cost, but by far not negligible.

The cost distribution of the robot system is reasonable with a slight overestimation

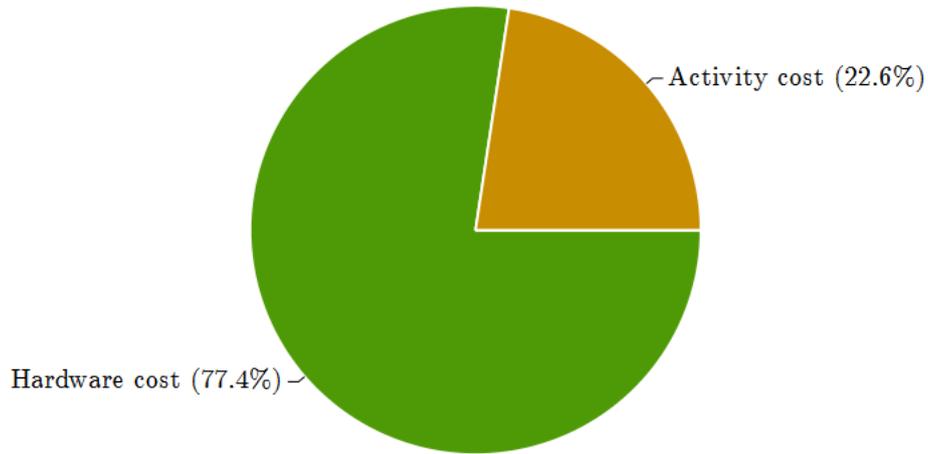


Figure 7.18 Cost distribution between hardware investment and activities for the handling use case.

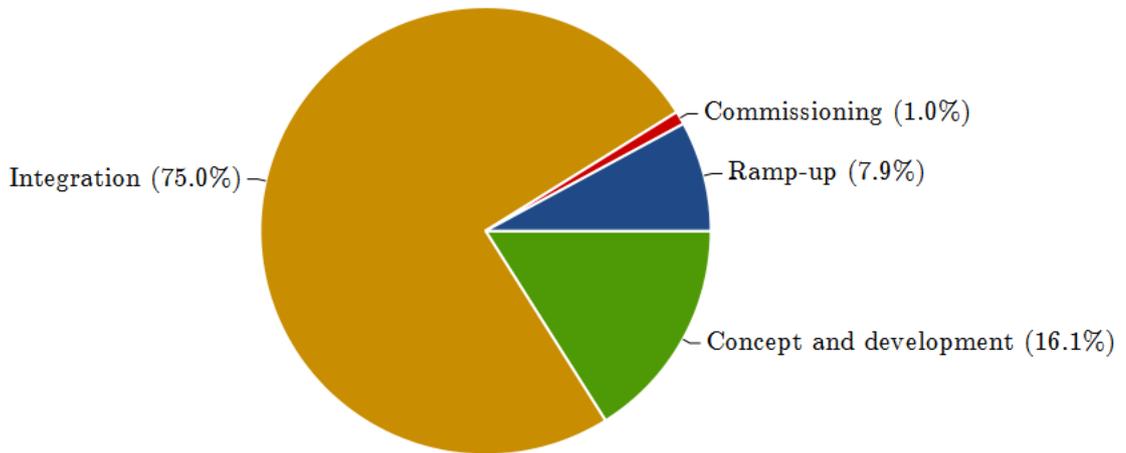


Figure 7.19 Distribution of cost between different life-cycle phases during design and integration.

of hardware efforts. This might improve if more information from past projects is provided to the database.

7.2.3 Project schedule

The generated project schedule for the handling use case is shown in Figure 7.20. The project schedule is reasonable and is in alignment with experience from past projects. The dependencies of different tasks are accurately reflected. The schedule appears to be too ambitious compared to real projects as the system does not

include additional buffer times and does not allow for delays, e.g. due to delayed components delivery. Furthermore, it is expected that by improving the used activity models and their data base, the quality of the model predictions can be gradually improved.

7.3 Evaluation of the developed method for cost-benefit assessment

The current implementation of the cost-benefit assessment software requires about 60 seconds to carry out the cost-benefit simulation. During this time, constant user interaction is required in order to resolve ambiguities and to provide information. The time is required to perform the matching of information in the local AutomationML description and in the knowledge base. This time frame is acceptable for integration of the cost-benefit assessment into many design processes in the development life-cycle of the system. Figure 7.21 gives an overview of different usage scenarios of the cost-benefit assessment tool throughout the development life-cycle of the robot system.

The first usage scenario is in very early phases of the project. Directly after ideation, the cost-benefit assessment software can be used to obtain a first cost-benefit assessment based on incomplete information. This can be used for the formalization of the project in the contract negotiations. The input in this phase is the application type of the robot system. The cost-benefit assessment software can complete a coarse system bill of material based on the required roles in the robot system. This allows the cost-benefit assessment software to perform a preliminary cost-benefit assessment based on rough work piece and production schedule data. This assessment and its refinement in the concept phase (see below) can be used to support the contract negotiations between system integrator and end-user. The cost-benefit assessment software is capable of covering this usage step due to its ability to complete missing information.

The second usage scenario is during the concept phase of the robot system. Here,

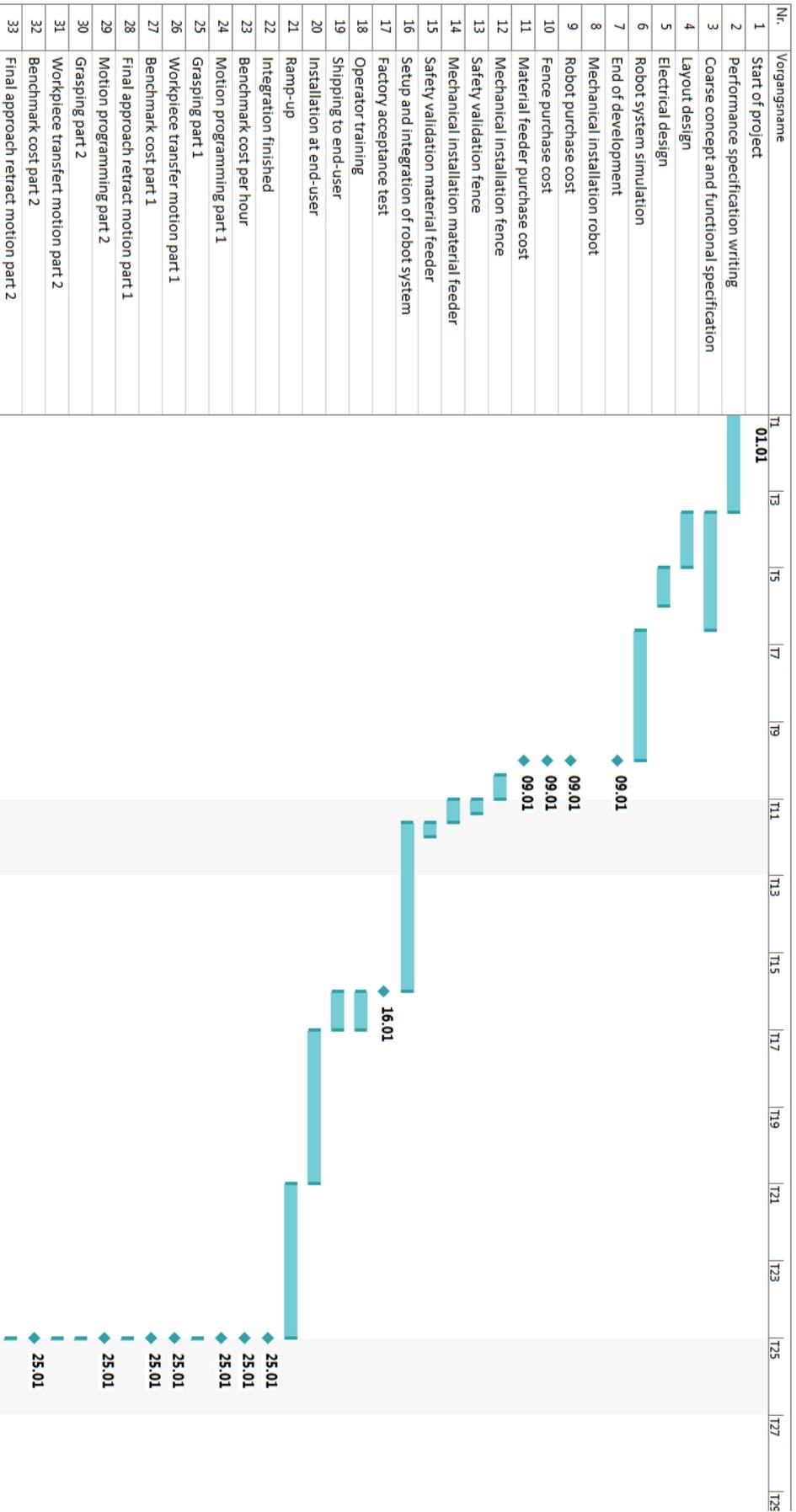


Figure 7.20 Project schedule for the handling use case.

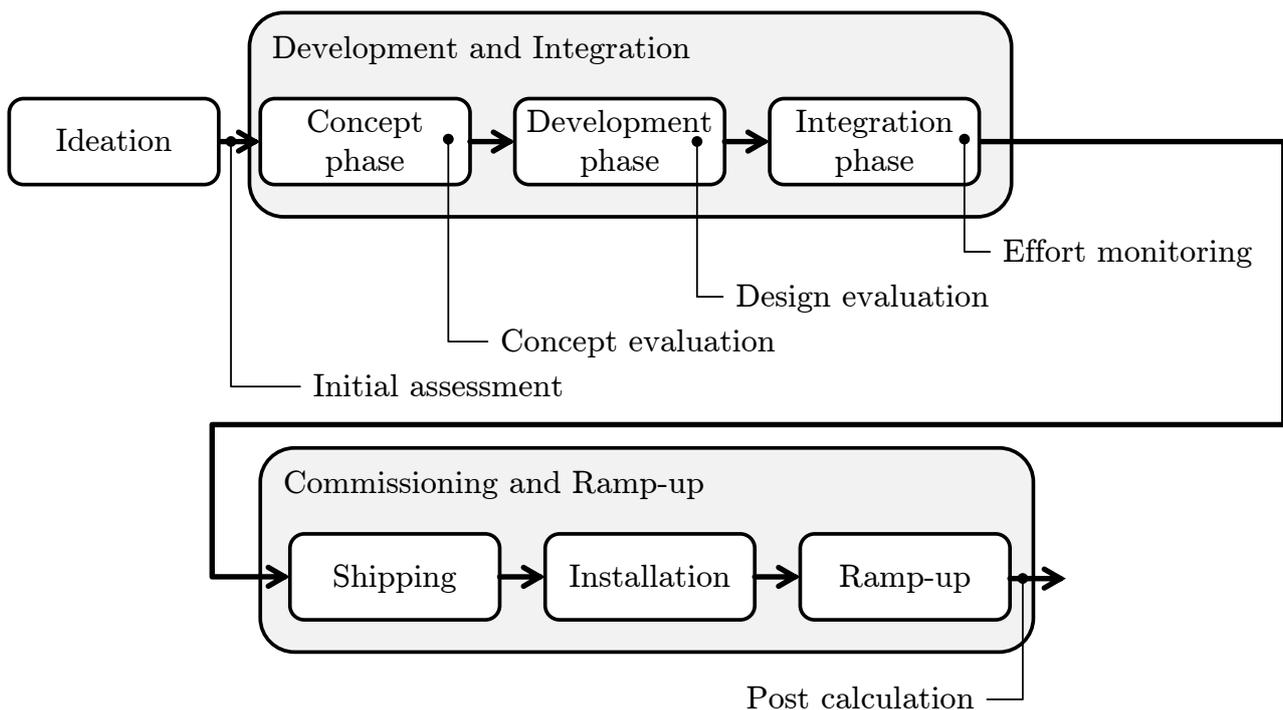


Figure 7.21 Integration of the cost-benefit assessment into the development life-cycle of the robot system.

the cost-benefit assessment software is used to evaluate different conceptual alternatives of the robot system design. The input in this phase is a coarse bill of material of the robot system specifying the contained component roles. Due to role-based reasoning, it informs the design engineer about missing components. Additionally, information on produced work pieces is used. The cost-benefit assessment software allows to compare different fundamental system concepts in order to make an informed decision on the system concept. It further facilitates the targeted discussion of system integrator and end-user as it makes the advantages and disadvantages of different robot system concept variants over the entire life-cycle transparent. Due to the execution time of one to two minutes, the cost-benefit assessment can easily be integrated into the iterative concept process.

The third possible usage scenario is the development phase of the robot system. Here, the main focus of development for the cost-benefit assessment software lies on concept design and development. Hence, these are the most important usage scenarios. In development, the input to the cost-benefit assessment is a bill

of material of the robot system already specifying the components by manufacturer and type rather than only specifying their role. In the development phase, the cost-benefit assessment software allows to compare different detailed design solutions, for example the type of component that is chosen. Also here, the low execution time permits rapid iterations.

Beyond the development, the cost-benefit assessment software produces information that can be used for project monitoring during integration and commissioning and post calculation after the robot system has been delivered. Here, the cost-benefit assessment software is a source of data for the project monitoring. As activity models can be deployed in different projects, the cost-benefit assessment software builds on benchmark data that are more reliable than individual estimates by project engineers.

As the cost-benefit assessment software utilizes AutomationML as data exchange format, design information can easily be imported from and exported to design tools. In practice, many design tools do not yet support the import and export of AutomationML in their standard configuration. Often, the import and export of AutomationML are experimental features. Furthermore, AutomationML allows for different structures and the definition of specific semantic groundings. As long as no common semantic grounding in form of a standardized role library is available, universal compatibility can hardly be reached. However, as the use of AutomationML spreads, this aspect is expected to be addressed by the AutomatioML consortium.

8 Conclusions and outlook

Aiming at business-decision support for investments in highly flexible, SME-suitable robotic systems, a new approach to cost-benefit assessment has been presented. Primary concerns have been management and integration of knowledge from different stakeholders, handling of technical and economic uncertainties, common understanding, and protection of stakeholder-internal information.

The following findings are directly related to the contributions listed in Section 8.1, and they also relate to future work according to Section 8.2, but these findings are here expressed from an application and community point of view to form a basis for the statement of the thesis:

1. The interests of system integrators and end-users in the realization of robot systems are not well-aligned. To overcome this misalignment of interests, accepted measures for the evaluation of robot systems over their entire life-cycle need to be established by the robotics community.
2. Lack of awareness of risk distribution between system integrator and end-user hampers proper risk management, resulting in unambitious robot systems that harm the ability to react to technical and economic changes. Instead, a new form of stakeholder collaboration needs to be established, based on transparency of costs and benefits.
3. Established approaches in cost-benefit assessment lack integration with the technical properties of robot systems and inadequately model the related mechanisms for cost generation. There is a need for extending existing cost-benefit assessment methods to obtain and utilize those properties, as required for improved decision-making support.

4. There is a lack of methodical approaches and underlying semantics for integrating knowledge from different stakeholders in order to obtain the required knowledge for cost-benefit assessment. In order to devise this semantics and to establish it in the market, a multidisciplinary effort is required with involvement from knowledge-engineering.
5. Apart from the technical aspect, the lack of knowledge integration also relates to business aspects. In order to overcome this limitation, all stakeholders need to contribute information to a shared knowledge base, even though the current business models in robotics build on obscuring this information. This can be achieved by providing mechanisms for knowledge management that allow to selectively expose information to other stakeholders as is possible with the architecture for knowledge management as also proposed here.

With the insight that technology developments need sound business prospects to enable delivery of the promises to applications, and thereby for human benefit, we can now conclude with the statement of this thesis:

Since investments in highly flexible SME-suitable robotic system are needed for a sustainable impact on the combination of profitability, productivity, maintained quality, manufacturing of customized products, reuse/reconfiguration of equipment, and good working conditions, and because cost-benefit for such flexible machines spans life-cycle phases that extend beyond the established methods for profitability analysis, the robotics community should form cross-disciplinary efforts for providing supported tools that facilitate the proposed vendor-neutral risk and uncertainty-aware cost-benefit assessment.

Here, the robotics community refers to technology providers, system integrators, end-users, and academia. Earlier chapters have shown the technical feasibility of the proposed efforts, for which the following contributions are believed to form a basis.

8.1 Contributions

The following summarizes the contributions of this thesis, formulated as five major results with sub-contributions itemized for three of them.

Formulation of the integration-aware cost-benefit problem. There is an existing gap between economics and technologies for flexible robotized production that is believed to not having been adequately addressed before. In this respect, the problem statement as defined in this thesis is in itself a contribution. It highlights the problem of market fragmentation and hence knowledge fragmentation in the robotics sector, and pays attention to the lack of analysis of inherent risks. While previously existing cost-benefit models focus on the usage phase of production machinery, this work gives emphasis to the system integration aspect.

A cost-benefit model for robot systems in SME-like production. The cost-benefit model structure builds on aspects of ABC, LCC and the PPR-approach. These approaches are combined and utilized in a new way. In particular, the combination of the structure of business-economic models with the level of detail of robotic engineering models is believed to be new. The cost-benefit model comprises the following sub-contributions:

- It is proposed to extend the PPR-approach with an additional view capturing accounting information and describing how actual costs are created based on consumption of commodities and usage of resources. This additional view is named cost-pool and the resulting approach is called PPRC-approach.
- Models for cost and benefit relevant aspects for the applications of welding (GMAW) and handling are provided. These models describe cost relevant aspects on product, process, resource and cost-pool level and show that the different concerns of knowledge can be represented independently from each other.

- A method for scheduling of activities based on priority and availability of required resources is given. This method for execution of the model produces realistic work plans for integration of the robot system and its usage in production.
- The cost-benefit assessment has been implemented and evaluated as a reference implementation based on two real-world application scenarios.
- The developed cost-benefit model is believed to be one of the first models in the domain of robotics to explicitly address the aspect of uncertainty in computations.

Interval arithmetics for uncertainty analysis. In order to represent uncertainty, interval arithmetics are used throughout the entire model. The use of interval arithmetics in the model lead to the following sub-contributions:

- By reflecting best and worst case scenarios through interval computations, the risk associated with the automation project can be analyzed. This increases the expressiveness of the cost-benefit model results when compared to pure nominal computations.
- The use of interval numbers offers a representation of uncertainty that is easily accessible for automation engineers, domain experts and managers. It is therefore easily possible for a domain expert to provide input to the model and understand the results including uncertainty.
- Interval numbers in capital expenditure budgeting, in particular in dynamic methods, have rarely been utilized. The specifics of discounting of uncertain cash-flows described by interval numbers are introduced. The problem of numeric overestimation is addressed by discounting uncertain cash-flows unidirectionally in time.

An SME-suitable knowledge-management architecture. Knowledge management and reasoning based on knowledge in the cost-benefit model are new concepts. The knowledge management was implemented and found

to give plausible results for the evaluated applications. The relevant sub-contributions in this respect are:

- The developed cost-benefit model strictly separates the concerns of the different PPRC-levels and allows to represent knowledge on each of these levels independently. This builds on the separation of concerns inherent in the structure of the model, and it supports integration of knowledge from different stakeholders (component manufacturer, system integrator and end-user) that do not share internal information such as profit margins. This aspect of knowledge integration considering business aspects has not been adequately addressed in earlier research.
- Another contribution is a reasoning technique that allows to match an incomplete description of a particular robot work cell with a knowledge base containing general information. The reasoning allows to rapidly build application-specific cost-benefit models from generic building blocks, and then to estimate missing information.
- The knowledge base used for reasoning rests on a semantic grounding supplied by an ontology. This ontology is believed to be the first domain ontology specific for cost aspects in robotics and is an important contribution of this thesis.

Risk-transparent decision support. The results of the cost-benefit model are presented in so called amortization graphs that highlight the cost aspects of the entire life-cycle of the robot system and the inherent uncertainty. This amortization graph is believed to be a new and more expressive form of presentation compared to traditional techniques in capital budgeting. The amortization view can be generated for the different stakeholders separately to reflect their view on costs and benefits. It allows to analyze the risk sharing between system integrator and end-user, which in investigated cases shows that a large portion of the realization risk rests with the system integrator. The transparency of the risk sharing is inspiring from a research point of view, in particular for research on future business models in machine building.

8.2 Further research

This thesis showed the applicability of the cost-benefit model for two common and transferable applications scenarios. An application to more scenarios and a strong increase of the number of available activity models and the parameter base would be beneficial and should be addressed in future research. This expansion in scope would help to prove the scalability of the presented approach and the reusability of knowledge in the knowledge base. The aim of this thesis is to establish a general framework in which exchangeable activity models for different aspects of systems engineering can be combined into an aggregated cost-benefit model. The development of more elaborate activity models in terms of expressiveness and refinement is subject to further work.

A standardization of the semantic grounding used in the RoboCost ontology and the AutomationML role libraries would highly improve interchangeability of information in the engineering process of robot systems. This would not only facilitate the cost-benefit assessment but generally improve data exchange in the design of automation systems. Promising activities to establish an accepted semantic grounding have already been initiated by the IEEE (Schlenoff et al., 2012) and the Mechanical Engineering Industry Association (Heister et al., 2017). Finally, methods for conciliating different semantic groundings will be necessary. The scientific basis for this challenge requires further research.

The parameterization of activity models through a connection to industrial and statistical data bases is expected to be a useful addition. For example, default personnel cost for certain job functions might be drawn from statistics if no detailed information is available. Through the use of SPARQL and the inherent capability of SPARQL for federated queries, the integration of new information sources is easily possible, if these data bases have semantic grounding. However, in order to use this potential more semantically annotated data bases are required. Furthermore, simulation techniques should be considered to parameterize activity models and to improve their expressiveness as well as their accuracy. For example, activity models in the development phase of the robot system could use

a simulation model of the robot system in order to estimate cycle times, energy consumption and other performance criteria. This would improve the quality of estimates in comparison to the coarse heuristics which are currently used. As simulation models are updated and refined along the development life-cycle also the accuracy of cost-benefit estimates would improve.

Further research might also address how complex dependencies inside the automation system can be handled. The developed cost-benefit model is currently not capable of describing automation systems whose processes are interlinked in a complex way. An example for this interlinking are robot systems with multiple robots in one work cell that perform partially sequential and partially parallel tasks. This can also be exemplified with the addressed handling use case where the effect of the actual hand-over time of human and robot to the machine on the achievable cycle time is not taken into account in the cost-benefit assessment.

As has been shown in this thesis, usually a high portion of the realization risk of robot systems sides with the system integrator. The aspect of risk should play a major role in research on future business models in machine building. For example, through pay-per-use business models the end-user further hedges its risks on the system integrator, because in addition to the realization risk also the financial risk of operating the robot system is shifted to the system integrator. It is questionable whether system integrators, which are often SMEs, are able to bear this risk.

Last but not least current technical developments and trends in robotics, such as for example plug and produce, automatic program planning and software-defined manufacturing, point at a reduction of explicit integration tasks in robotics. It is therefore a plausible future scenario that integration tasks, except equipment installation, are completely eliminated. In fact, this would reduce the role of the system integrator to a software vendor or equipment fitter and strengthen the role of the component manufacturers. This development would make the cost-benefit assessment much easier because estimation of integration efforts becomes easier and integration efforts become less important. At the same time the reduction of integration tasks will make the automatic assessment of robotic solutions more

important because the mediating role of the system integrator no longer exists and the end-user requires decision support in order to make automation decisions. The role of knowledge-based cost-benefit assessment of robotic solutions is therefore expected to increase in the future.

Appendix

1 Requirements for the cost-benefit assessment system

Subsequently, requirements for the cost-benefit assessment tool for robot systems are derived from the problem statement and usage scenario for the costing system. The requirements are given as overall system requirements and requirements for the key research areas, i.e. the cost model, knowledge management, and life-cycle integration.

1.1 Overall requirements

In Table 1 the overall system requirements, spanning all of the core research areas, are listed.

Table 1 Overall requirements for the cost-benefit assessment tool.

ID	Requirement	Comment
S1	The model shall incorporate the views of system integrator and end-user.	System integrator vs. end-user have fundamentally different views on the investment cost and how they are financed.
S2	The model shall be usable for executives from the robotics end-users and system integrators without dedicated knowledge in cost calculations of automation systems or statistics.	n/a
S3	The model shall allow to obtain a cost-benefit estimate from an existing system design in less than 10 minutes.	n/a
S4	The model shall indicate the related uncertainty of the cost-benefit assessment.	The uncertainty in the cost-benefit assessment conveys an understanding of the associated project risk.
S5	The model shall compare the cost and benefit of two or more system options for the application.	The benefits of a certain robot system solution can only be assessed with respect to an alternative (see Section 3.4.2)

1.2 Requirements on cost model

The specific requirements for the cost model are listed in Table 2.

Table 2 Requirements for the cost model.

ID	Requirement	Comment
S6	The model shall cover all cost-relevant life-cycle phases of the robot system.	Design decisions and system performance can only be assessed in the light of the complete system life-cycle.
S7	The cost-benefit model shall be modular with respect to the used activity models.	The ability to adapt and change single activity models offers the possibility to tailor the model for the specific products, processes and resources of the end-user.
S8	The inter-dependencies between different life-cycle phases shall be contained in the model.	Subsequent life-cycle phases shall use the information provided by previous life-cycle phases, e.g. system performance information is derived from the system design.
S9	The model shall incorporate all activities required for the development and use of the robot system.	Typically, cost items such as ramp-up and training are not taken into account in cost models. The result are faulty cost estimates neglecting important cost items.

1.3 Requirements for knowledge management

The requirements for knowledge management in the cost-assessment tool are listed in Table 3.

Table 3 Requirements for knowledge management in the cost assessment tool.

ID	Requirement	Comment
S10	The model shall integrate knowledge from component suppliers, system integrator and end-user.	Required knowledge for cost-benefit assessment in robotics is distributed among different stakeholders.
S11	The cost-benefit system shall integrate information from different stakeholders into a consolidated picture without making confidential information available to competitors or negotiation partners.	Information on price formation or internal cost of the project shall not be visible to the end-user, while information on competing bids shall not be available to the competing system integrators.
S12	Missing information shall be estimated from one or several knowledge repositories utilizing direct associations, estimates from similar projects or components and estimates based on the semantic meaning of components and activities.	The model shall have means to estimate missing data and produce meaningful estimates including an assessment of the uncertainty of these estimates.
S13	The knowledge management system shall be capable of resolving contradicting information on cost items.	Prices and required efforts depend on a large numbers of factors such as time, customer, salaries and competence of employees. This leads to conflicting cost estimates in the knowledge repository that need to be resolved.
S14	It shall be possible for the expert using the system to insert estimates of missing information.	For many cost items, an expert will be capable of giving educated estimates of upper and lower bounds for missing information.

1.4 Integration into development life-cycle

The requirements on the integration of the cost-benefit assessment in the engineering life-cycle are listed in Table 4.

Table 4 Requirements for the integration of the cost assessment tool in the development life-cycle.

ID	Requirement	Comment
S15	It shall be able to import design information of a robot system from common PLM and CAX into the costing system.	Seamless data transfer helps to ensure a tight integration of the cost estimation into the design process.
S16	The costing system shall allow rough cost-benefit assessments in very early phases of the development life-cycle.	Application of the cost-benefit tool during concept design, where few information on the system setup is available, shall be possible to obtain a first assessment of economic feasibility.
S17	The system shall support contract negotiations for the end-user by making life-cycle cost of different alternatives transparent.	Consequences of alternate designs over the life-cycle of the robot system are today hard to assess for the end-user.
S18	The system shall be able to support the contract negotiations for the system integrator by making the life-cycle costs and risks transparent. Confidential information shall not be shared.	Cost-benefit arguments over the entire life-cycle of the robot system might justifying a higher system price. The tool shall establish cost transparency over the entire life-cycle.
S19	The cost-benefit tool shall support design decisions during detailed design and use available design information to produce an elaborate cost estimate.	n/a
S20	The cost-benefit tool shall allow post calculation of the finalized system.	Using the cost model in post calculation might allow new forms of cooperation between end-user and system integrator by awarding bonuses for improved life-cycle performance.

2 Developed AutomationML role library

Key concepts of the AutomationML role library are introduced in Table 5.

Table 5 The SMERobotics role library for AutomationML.

Role type	Role subtype	Role examples	Comment
Software roles		Planning, offline programming, execution, configuration, custom programming	Roles relating to software in robot systems
Safety equipment roles		Separating and non-separating safety equipment, safety sensors, safety controllers	Roles relating to safety relevant parts that require a safety life-cycle
Peripheral roles	Tool roles	Welding tool, suction gripper, claw gripper, sensor head, machining tool	Roles of robot end-effectors
	Material commissioning roles	Handover station, feeder, crate supply bay, conveyor	Roles related to material feeding
	Fixture roles	Universal fixture, welding table, sheet metal fixture	Roles relating to fixing of work piece during processing
Automation system roles		Robot system, welding robot system, machine tending robot system, assembly robot system	Roles relating to the robot system as a whole
Automation component roles	Sensor roles	Point-cloud sensor, triangulation sensor, force sensor, position switch	Roles of sensors in the robot cell
	Process peripherals	Welding power source, vacuum pump	Roles related to peripheral devices required for the process
	Mechanical parts	End-effector structure, robot support structure, cell frame	Roles related to part for mechanical integrity
	Manipulators	Robot, work piece positioner	Roles of robots and other manipulators
	End-effector components	Welding torch, tool changer, suction cup, machining spindle, gripper module	Roles related to components of the end-effector
	Controller roles	PLC, industrial PC	Roles of controller hardware
	Auxiliary components	Drive, motor, switch	Roles of basic building blocks for automation

3 Details on product functions for determination of design efforts

Table 6 Criteria for product complexity in the design phase.

ID	Name	Type	Source	Explanation
\mathcal{C}_1	Clocked production	boolean	Request from user	Is the system part of a clocked production system in which the tact of the entire system depends on the robot system?
\mathcal{C}_2	IT-Chain integration	boolean	Request from user	Is the robot system integrated into an IT-chain for generating orders and robot programs involving robot programming?
\mathcal{C}_3	Interfacing with external machines	boolean	Request from user	Does the robot system interact with additional machinery, e.g. a machine tool which is fed by the robot, that need to be integrated in the control system?
\mathcal{C}_4	Supervisory control	boolean	Request from user	Has the robot system to be integrated into a supervisory control system such as a line PLC or MES system?
\mathcal{C}_5	Number of robots	Count	Extract from AML data	How many robots are installed in the robot system?
\mathcal{C}_6	Number of robot work stations	Count	Extracted from AML data	How many workstations for robots are in the work cell? A work station is a fixture or feeding station with which the robot interacts.
\mathcal{C}_7	Part accessibility	boolean	Request from user	Are there portions of the work piece on which the robot works that are hard to access, e.g. because of narrow geometry or undercuts?
\mathcal{C}_8	Handling of parts	boolean	Request from user	Does the robot interact with bulky parts or with parts' surfaces that are very sensitive?
\mathcal{C}_9	End-user	boolean	Request from user	Is the user of the robot system a large company? Large companies typically have specialized processes that need to be followed and generate overhead.
\mathcal{C}_{10}	Process related risk	boolean	Request from user	Is the process known to be stable or do questions exist regarding the ability to automate the process?

Table 7 Criteria for product complexity of the end-effector.

ID	Name	Type	Source	Explanation
\mathcal{C}_{11}	Exact End-effector calibration	boolean	Request from user	Determine whether the end-effector of the robot system needs to be calibrated accurately, i.e. with an absolute accuracy of better than 1 mm, internally or with respect to the robot, e.g. for specific sensory functions.
\mathcal{C}_{12}	Sensory capabilities	boolean	Extract from AML	Does the sensor possess advanced sensory capabilities such as force-torque sensing, displacement sensing, 3D-sensing, etc.?
\mathcal{C}_{13}	Mechanical capabilities	boolean	Request from user	Does the end-effector possess advanced mechanical capabilities such as a floating mounting to the robot, compensation element, etc.?
\mathcal{C}_{14}	Additional axis	Count	Extract from AML info	Does the end-effector have additional axes that need to be controlled?
\mathcal{C}_{15}	Tool changer	boolean	Extract from AML	Does the end-effector have a tool changer?
\mathcal{C}_{16}	Solid process media	boolean	Extract from AML description and process model	Does an installed process device or the process itself require the supply of solid process media, such as for example wire, powder, rivets?)
\mathcal{C}_{17}	Geometry-based gripping	boolean	Request from user	Does a gripper copy the shape of the work piece in order to work?
\mathcal{C}_{18}	Part stability	boolean	Request from user	Are the parts that are handled in the work cell stable or is the final exact shape determined by the gripper and gripping action?
\mathcal{C}_{19}	Surface sensitivity	boolean	Request from user	Is the surface of the handled parts easy to be damaged and requires special precautions to avoid damaging of the surface?

Table 8 Criteria for product complexity of fixtures and jigs.

ID	Name	Type	Source	Explanation
\mathcal{C}_{20}	Number of parts	Count	Extract from AML description	How many parts are joined by the fixture?
\mathcal{C}_{21}	Positioning	boolean	Request from user	Is the fixture responsible for accurate positioning of the parts with respect to each other, i.e. in the range of < 0.5 mm?
\mathcal{C}_{22}	Jigging	boolean	Request from user	Does the fixture hold the exact part position against influence of the process, e.g. deflection?
\mathcal{C}_{23}	Additional axis	boolean	Extract from AML description	Does the fixture possess an additional axis for positioning of the entire fixture?
\mathcal{C}_{24}	Automatic feeding	boolean	Request from user	Are the parts fed automatically to the fixture?
\mathcal{C}_{25}	Automatic operation	boolean	Request from user	Is the fixture operated, i.e. opened and closed, automatically?

4 Parameters of the addressed use cases

In the following the parameters used for the cost-benefit assessment in the evaluation chapter of this thesis (Chapter 7) are given.

4.1 Parameters of the GMAW use case

Table 9 shows the used parameters for the cost-benefit calculation of the GMAW welding use case.

Table 9 Parameters for the cost-benefit calculation of the GMAW use case.

Property		Regular	Overpricing	Underpricing	High realization risk	High usage risk	Project runaway
Benchmark cost per hour (€)		[0, 0]					
Fixed cycle time		no					
System cost (€)		300 000	500 000	150 000	300 000		
Initial payment		33%					
FAT payment		33%					
Excavator bucket	Seam length (m)	20					
	Lot size	5					
	Relative volume	3					
	Benchmark cost per piece (€)	[700, 800]			[600, 900]	[500, 600]	
Packing roller	Seam length (m)	3					
	Lot size	10					
	Relative volume	1					
	Benchmark cost per piece (€)	[400, 500]			[300, 600]	[200, 300]	

4.2 Parameters of the handling use case

Table 10 gives an overview of the parameters used for the cost-benefit assessment of the handling use case.

Table 10 Parameters for the cost-benefit calculation of the handling use case.

Property	Value	
Benchmark cost per hour (€)	[45, 55]	
Fixed cycle time	yes	
Cycle time (s)	[55, 70]	
System cost (€)	200 000	
Initial payment	33%	
FAT payment	33%	
Shaft blank type 1	Mass (kg)	1
	Maximum acceleration (m s^{-2})	10
	Lot size	50
	Relative volume	2
	Transfer distance (m)	1.5
	Number of points	9
	Benchmark cost per piece (€)	[0, 0]
Shaft blank type 2	Mass (kg)	0.5
	Maximum acceleration (m s^{-2})	10
	Lot size	50
	Relative volume	1
	Transfer distance (m)	1
	Number of points	10
	Benchmark cost per piece (€)	[0, 0]

Bibliography

Abele et al., 2013

Abele, Lisa; Legat, Christoph; Grimm, Stephan; Müller, Andreas W., 2013.
Ontology-based validation of plant models.
In: *11th IEEE International Conference on Industrial Informatics (INDIN)*,
Pp. 236–241

Abrahamsson et al., 2009

Abrahamsson, Pekka; Conboy, Kieran; Wang, Xiaofeng, 2009.
‘Lots done, more to do’: the current state of agile systems development research.
European Journal of Information Systems, **18**
(4), pp. 281–284

Abrahamsson et al., 2007

Abrahamsson, Pekka; Moser, Raimund; Pedrycz, Witold; Sillitti, Alberto; Succi, Giancarlo, 2007.
Effort Prediction in Iterative Software Development Processes – Incremental Versus Global Prediction Models.
In: *First International Symposium on Empirical Software Engineering and Measurement (ESEM), 2007*.
Madrid, 20/09/2007–21/09/2007, pp. 344–353

- Aggarwal et al., 1991** Aggarwal, Raj; Edward, J.; Mellen, Louise E., 1991.
Justifying investments in flexible manufacturing technology: Adding strategic analysis to capital budgeting under uncertainty.
Managerial Finance, **17** (2/3), pp. 77–88
- Agyapong-Kodua et al., 2011** Agyapong-Kodua, Kwabena; Wahid, Bilal; Weston, Richard H., 2011.
Towards the derivation of an integrated process cost-modelling technique for complex manufacturing systems.
International Journal of Production Research, **49** (24), pp. 7361–7377
- Ajoudani et al., 2018** Ajoudani, Arash; Zanchettin, Andrea Maria; Ivaldi, Serena; Albu-Schäffer, Alin; Kosuge, Kazuhiro; Khatib, Oussama, 2018.
Progress and prospects of the human-robot collaboration.
Autonomous Robots, **42** (5), pp. 957–975
- Anandan, 2016** Anandan, Tanya M., 2016.
Calculating Your ROI for Robotic Automation: Cost vs. Cash Flow https://www.robotics.org/content-detail.cfm/Industrial-Robotics-Industry-Insights/Calculating-Your-ROI-for-Robotic-Automation-Cost-vs-Cash-Flow/content_id/5285
Visited on: 10/12/2017

- Asiedu et al., 1998** Asiedu, Y.; Gu, P., 1998.
Product life cycle cost analysis: State of the art review.
International Journal of Production Research, **36** (4), pp. 883–908
- AutomationML, 2013** Whitepaper AutomationML - Part 1: Architecture.
AutomationML consortium
- Bailey et al., 2009** Bailey, James; Bry, François; Furche, Tim; Schaffert, Sebastian, 2009.
Semantic web query languages.
In: Liu, Ling; Öszu, M. Tamer (Hrsg.): *Encyclopedia of database systems*,
Pp. 2583–2586.
ISBN 978-0-387-39940-9
- Bashir et al., 1999** Bashir, Hamdi A.; Thomson, Vince, 1999.
Metrics for design projects: a review.
Design Studies, **20** (3), pp. 263–277
- Bashir et al., 2001** Bashir, Hamdi A.; Thomson, Vince, 2001.
Models for estimating design effort and time.
Design Studies, **22** (2), pp. 141–155
- Beraldi et al., 2013** Beraldi, Patrizia; Violi, Antonio; De Simone, Francesco; Costabile, Massimo; Massabò, Ivar; Russo, Emilio, 2013.
A multistage stochastic programming approach for capital budgeting problems under uncertainty.
IMA Journal of Management Mathematics, **24** (1), pp. 89–110

- Berners-Lee et al., 2001** Berners-Lee, Tim; Hendler, James; Lassila, Ora, 2001.
The Semantic Web.
Scientific American, visited on: 02/11/2018.
Available from: https://www-sop.inria.fr/acacia/cours/essi2006/Scientific%20American_%20Feature%20Article_%20The%20Semantic%20Web_%20May%202001.pdf
- Beynon et al., 2000** Beynon, Malcolm; Curry, Bruce; Morgan, Peter, 2000.
The Dempster–Shafer theory of evidence: an alternative approach to multicriteria decision modelling.
Omega, **28** (1), pp. 37–50
- Bhalerao et al., 2009** Bhalerao, Shilpa; Ingle, Maya, 2009.
Incorporating Vital Factors in Agile Estimation through Algorithmic Method.
IJCSA, **6** (1), pp. 85–97
- Birkhofer et al., 2012** Birkhofer, Rolf et al., 2012.
Life-Cycle-Management for Automation Products and Systems: A Guideline by the System Aspects Working Group of the ZVEI Automation Division.
The German Electric and Electronic Manufacturers' Association (ZVEI), Automation Division.
ISBN 978-3-939265-00-9

- Bjorke et al., 2016** Bjorke, Oystein; DNV GL AS; LECO Corporation, 2016.
OxyPlot <http://www.oxyplot.org/>
Visited on: 10/12/2017
- Black et al., 1973** Black, Fischer; Scholes, Myron, 1973.
The pricing of options and corporate liabilities.
Journal of political economy, **81** (3), pp. 637–654
- Boehm, 1984** Boehm, Barry W., 1984.
Software Engineering Economics.
IEEE Transactions on Software Engineering, **SE-10** (1), pp. 4–21
- Boehm et al., 2000** Boehm, Barry W.; Clark; Horowitz; Brown; Reifer; Chulani; Madachy, Ray; Steece, Bert, 2000.
Software Cost Estimation with Cocomo II.
Upper Saddle River, NJ : Prentice Hall.
ISBN 978-0-137-02576-3
- Borison, 2005** Borison, Adam, 2005.
Real options analysis: where are the emperor's clothes?
Journal of applied corporate finance, **17** (2), pp. 17–31
- Campolongo et al., 2007** Campolongo, Francesca; Cariboni, Jessica; Saltelli, Andrea, 2007.
An effective screening design for sensitivity analysis of large models.
Environmental modelling & software, **22** (10), pp. 1509–1518

- Chandrasegaran et al., 2013** Chandrasegaran, Senthil K.; Ramani, Karthik; Sriram, Ram D.; Horváth, Imré; Bernard, Alain; Harik, Ramy F.; Gao, Wei, 2013.
The evolution, challenges, and future of knowledge representation in product design systems. *Computer-aided design*, **45** (2), pp. 204–228
- Chella et al., 2002** Chella, Antonio; Cossentino, Massimo; Pirrone, Roberto; Ruisi, Andrea, 2002.
Modeling Ontologies for Robotic Environments.
In: *Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering (SEKE), 2002*,
Pp. 77–80
- Chemnitz et al., 2011** Chemnitz, Moritz; Schreck, Gerhard; Krüger, Jörg, 2011.
Analyzing energy consumption of industrial robots.
In: *2011 IEEE 16th Conference on Emerging Technologies Factory Automation (ETFA)*,
Pp. 1–4
- Chiu et al., 1994** Chiu, Chui-Yu; Park, Chan S., 1994.
Fuzzy cash flow analysis using present worth criterion.
The Engineering Economist, **39** (2), pp. 113–138
- Clinton et al., 2004** Clinton, B. Douglas; Webber, Sally A., 2004.
RCA at Clopay.
Strategic Finance, **86** (4), pp. 21–26

-
- Coase, 1937** Coase, Ronald Harry, 1937.
The Nature of the Firm.
Economica, **4** (16), pp. 386–405
- Cohn, 2005** Cohn, Mike, 2005.
Agile estimating and planning.
Upper Saddle River, NJ : Prentice Hall.
ISBN 978-0-131-47941-8
- CubicWeb, 2015** CubicWeb, 2015
<http://www.cubicweb.org/>
Visited on: 10/12/2017
- Damghani et al., 2011** Damghani, Kaveh Khalili; Sadi-Nezhad, Soheil;
Aryanezhad, Mir-Bahador, 2011.
A modular Decision Support System for optimum investment selection in presence of uncertainty: Combination of fuzzy mathematical programming and fuzzy rule based system.
Expert Systems with Applications, **38** (1), pp. 824–834
- Daumas et al., 2009** Daumas, Marc; Lester, David; Muñoz, César., 2009.
Verified Real Number Calculations: A Library for Interval Arithmetic.
IEEE Transactions on Computers, **58** (2), pp. 226–237
- De Finetti, 2017** De Finetti, Bruno, 2017.
Theory of Probability. A Critical Introductory Treatment.
London : John Wiley & Sons.
ISBN 978-1-119-28637-0

- Dempster, 1967** Dempster, Arthur P., 1967.
Upper and Lower Probabilities Induced by a Multivalued Mapping.
The Annals of Mathematical Statistics, **38** (2), pp. 325–339
- Dietmair et al., 2008** Dietmair, Anton; Verl, Alexander, 2008.
Energy consumption modeling and optimization for production machines.
In: *2008 IEEE International Conference on Sustainable Energy Technologies*.
Singapore, 24/11/2008–27/11/2008, pp. 574–579
- Dietz et al., 2013** Dietz, Thomas; Pott, Andreas; Hägele, Martin; Verl, Alexander, 2013a.
A new, uncertainty-aware cost-model for cost-benefit assessment of robot systems.
In: *44th International Symposium on Robotics (ISR), 2013*.
Seoul, 24/10/2013–26/10/2013
- Dietz et al., 2015** Dietz, Thomas; Pott, Andreas; Hägele, Martin; Verl, Alexander, 2015.
Knowledge-based cost engineering for industrial robot systems.
In: *Proceedings of the 11th Conference on Automation Science and Engineering (CASE), 2015*.
Gothenburg, 24/08/2015–28/08/2015

-
- Dietz et al., 2013** Dietz, Thomas; Pott, Andreas; Verl, Alexander, 2013b.
Practice for planning and realization of advanced industrial robot systems.
In: *ISR 2013 : The 44th International Symposium on Robotics, Oct. 24-26, 2013, Seoul, Korea. Proceedings*,
PaerA5–3, 4
- Dolgui et al., 2004** Dolgui, Alexandre; Pashkevich, Anatol, 2004.
Cluster-level operations planning in arc-welding robotic cell with positioning table :
Research Report G2I-EMSE 2004-500-002 of Ecole des Mines de Saint-Etienne
- Draht, 2010** Draht, Rainer (ed.), 2010.
Datenaustausch in der Anlagenplanung mit AutomationML — Integration von CAEX, PLCopen XML und COLLADA.
Berlin and Heidelberg : Springer-Verlag.
ISBN 978-3-642-04674-2
- Drayton et al., 2003** Drayton, Peter; Neward, Ted; Albahari, Ben, 2003.
C# in a Nutshell
2nd ed.
Sebastopol, CA : O'Reilly & Associates, Inc.
ISBN 978-1-491-98765-0
- Dubois, 2006** Dubois, Didier, 2006.
Possibility theory and statistical reasoning.
Computational statistics & data analysis, **51**
(1), pp. 47–69

- Dubois et al., 2000** Dubois, Didier; Kerre, Etienne; Mesiar, Radko; Prade, Henri, 2000.
Fundamentals of Fuzzy Sets.
In: Dubois, Didier; Prade, Henri (Hrsg.):
Boston : Springer, chap. Fuzzy Interval Analysis, pp. 483–581.
ISBN 978-1-4615-4429-6
- Edwards, 2012** Edwards, Stephanie, 2012.
Activity Based Costing - Topic gateway series No. 1
The Chartered Institute of Management Accountants, technical report
- Ehmann et al., 1997** Ehmann, K. F.; Kapoor, S. G.; DeVor, R. E.; Lazoglu, I., 1997.
Machining Process Modeling: A Review.
Journal of Manufacturing Science and Engineering, **119** (4B), pp. 655–663
- Ellram et al., 1998** Ellram, Lisa; Siferd, Sue P., 1998.
Total Cost of Ownership: A Key Concept in Strategic Cost Management Decisions.
Journal of Business Logistics, **19** (1), pp. 55–84
- EFWF, 2007** *Example of Cost Calculation in Welding*, 2007
Porto Salvo, EWF — European Federation for Welding, Joining and Cutting, technical report
- Ferrin et al., 2002** Ferrin, Bruce G.; Plank, Richard E., 2002.
Total Cost of Ownership Models: An Exploratory Study.
Journal of Supply Chain Management, **38** (3), pp. 18–29

- Fikes et al., 2004** Fikes, Richard; Hayes, Patrick; Horrocks, Ian, 2004.
OWL-QL — a language for deductive query answering on the Semantic Web.
Web semantics: Science, services and agents on the World Wide Web, **2** (1), pp. 19–29
- Fine et al., 1990** Fine, Charles H.; Freund, Robert M., 1990.
Optimal investment in product-flexible manufacturing capacity.
Management Science, **36** (4), pp. 449–466
- Fleischer et al., 2007** Fleischer, Jürgen; Wawerla, Marc; Niggeschmidt, Stephan, 2007.
Machine life cycle cost estimation via Monte-Carlo simulation.
In: Takata, Shozo; Umeda, Yasushi (Hrsg.): *Advances in Life Cycle Engineering for Sustainable Manufacturing Businesses*,
Pp. 449–453.
ISBN 978-1-84628-935-4
- French, 1995** French, Simon, 1995.
Uncertainty and Imprecision: Modelling and Analysis.
Journal of Operational Research Society, **46** (1), pp. 70–79
- Friedl et al., 2009** Friedl, Gunther; Hammer, Carola; Pedell, Burkhard; Küpper, Hans-Ulrich, 2009.
How do German Companies run their cost accounting systems?
Management Accounting Quarterly, **10**, pp. 28–34

Graham et al., 2001

Graham, John R.; Harvey, Campbell R., 2001.
The theory and practice of corporate finance:
evidence from the field.

Journal of Financial Economics, **60**, pp. 187–
243

Hägele et al., 2011

Hägele, Martin et al. (eds.), 2011.

*Wirtschaftlichkeitsanalysen neuartiger Service-
robotik - Anwendungen und ihre Bedeutung für
die Robotik-Entwicklung : EFFIROB-Studie.
Eine Analyse der Fraunhofer-Institute IPA
und ISI im Auftrag des BMBF (Kennzeichen
01M09001) zwischen dem 1. Dezember 2009
und dem 30. November 2010.*

Stuttgart visited on: 31/10/2018.

Available from: [http://fhgonline.fhg.de/
bibliotheken/ipa/Studie-EFFIROB.pdf](http://fhgonline.fhg.de/bibliotheken/ipa/Studie-EFFIROB.pdf)

Hälsig, 2015

Hälsig, André, 2015.

Energetische Bilanzierung von Fügeverfahren
Chemnitz, Universitätsverlag Chemnitz, Diss.
2015

- Heister et al., 2017** Heister, Reinhard; Mosch, Christian; Hoppe, Stefan, 2017.
VDMA Robotics partners with OPC Foundation to achieve standardized information integration across robots/machines to realize requirements of Industrie4.0 machine-to-machine (M2M) <https://opcfoundation.org/news/press-releases/vdma-robotics-partners-opc-foundation-achieve-standardized-information-integration-across-robotsmachines-realize-requirements-industrie4-0-machine-machine-m2m/>
Visited on: 10/12/2017
- Helmer, 1965** Helmer, Olaf, 1965.
Social Technology
DTIC Document, technical report
- Helton et al., 1996** Helton, Jon C.; Burmaster, David E., 1996.
Guest editorial: treatment of aleatory and epistemic uncertainty in performance assessments for complex systems.
Reliability Engineering & System Safety, **54** (2-3), pp. 91–94
- Helton et al., 2004** Helton, Jon C.; Johnson, Jay D.; Oberkampf, William L., 2004.
An exploration of alternative approaches to the representation of uncertainty in model predictions.
Reliability Engineering & System Safety, **85** (1), pp. 39–71

- Helton et al., 2006** Helton, Jon C.; Johnson, Jay Dean; Sallaberry, Cedric J.; Storlie, Curt B., 2006.
Survey of sampling-based methods for uncertainty and sensitivity analysis.
Reliability Engineering & System Safety, **91** (10), pp. 1175–1209
- Hertz, 1964** Hertz, David Bendel, 1964.
Risk analysis in capital investment.
Harvard Business Review, **42** (1), pp. 95–106
- Hilhorst et al., 2008** Hilhorst, Cokky; Ribbers, Piet; van Heck, Eric; Smits, Martin, 2008.
Using Dempster-Shafer theory and real options theory to assess competing strategies for implementing IT infrastructures: A case study.
Decision Support Systems, **46** (1), pp. 344–355
- Hodgson et al., 2011** Hodgson, Ralph; Keller, Paul J.; Hodges, Jack; Spivak, Jack, 2011.
QUDT-quantities, units, dimensions and data types in OWL and XML <http://www.qudt.org>
Visited on: 10/12/2017
- Hofmann et al., 2012** Hofmann, Erik; Maucher, Daniel; Hornstein, Jens; den Ouden, Rainer, 2012.
Capital equipment purchasing: Optimizing the total cost of CapEx sourcing.
Berlin and Heidelberg : Springer Science & Business Media.
Professional Supply Management.
ISBN 978-3-642-25737-7

-
- Homma et al., 1996** Homma, Toshimitsu; Saltelli, Andrea, 1996.
Importance measures in global sensitivity analysis of nonlinear models.
Reliability Engineering & System Safety, **52** (1), pp. 1–17
- Horváth et al., 2011** Horváth, Péter; Mayer, Reinhold, 2011.
Was ist aus der Prozesskostenrechnung geworden?
Controlling & Management, **55** (2), pp. 5–10
- Huang, 2008** Huang, Xiaoxia, 2008.
Mean-variance model for fuzzy capital budgeting.
Computers & Industrial Engineering, **55** (1), pp. 34–47
- Huss, 1988** Huss, William R., 1988.
A move toward scenario analysis.
International Journal of Forecasting, **4** (3), pp. 377–388
- IEC 62424, 2016** IEC 62424:2016.
Representation of process control engineering - Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools
- IEEE 1872-2015, 2015** IEEE 1872-2015.
ORA — Ontologies for Robotics and Automation — IEEE Standard Ontologies for Robotics and Automation

- Iooss et al., 2015** Iooss, Bertrand; Lemaître, Paul, 2015.
A review on global sensitivity analysis methods.
In: Dellino, Gabriella (Hrsg.): *Uncertainty Management in Simulation-Optimization of Complex Systems: Algorithms and Applications*,
Pp. 101–122.
ISBN 978-1-4899-7547-8
- ISO 10303, 2011** ISO 10303:2011.
Industrial automation systems and integration — product data representation and exchange
- ISO 18629, 2004** ISO 18629:2004.
Industrial automation systems and integration — process specification language
- ISO 8373, 2012** ISO 8373:2012.
Robots and robotic devices — Vocabulary
- Jaulin, 2001** Jaulin, Luc, 2001.
Applied interval analysis: with examples in parameter and state estimation, robust control and robotics.
London : Springer Science & Business Media.
ISBN 978-1-4471-0249-6
- Jena, 2015** Jena, Apache, 2015
<http://jena.apache.org/>
Visited on: 10/12/2017
- Jerrell, 1997** Jerrell, Max E., 1997.
Interval Arithmetic for Input-output Models with Inexact Data.
Computational Economics, **10** (1), pp. 89–100

-
- Jorgensen et al., 2007** Jorgensen, Magne; Shepperd, Martin, 2007.
A systematic review of software development cost estimation studies.
IEEE Transactions on Software Engineering, **33** (1), pp. 33–53
- Kaplan, 1986** Kaplan, Robert S., 1986.
Must CIM be justified by faith alone?
Harvard Business Review, **64** (2), pp. 87–95
- Kaplan, 1997** Kaplan, Robert S., 1997.
Introduction to activity-based costing.
Harvard Business School
- Kaplan et al., 2004** Kaplan, Robert S.; Anderson, Steven R., 2004.
Time-Driven Activity-Based Costing.
Harvard Business Review, **82** (11), pp. 131–138
- Kaplan et al., 1992** Kaplan, Robert S.; Norton, David P., 1992.
The Balanced Scorecard: Measures that Drive Performance.
Harvard Business Review, **70** (1), pp. 71–79
- Kaplan et al., 2008** Kaplan, Robert S.; Norton, David P., 2008.
Mastering the management system.
Harvard Business Review, **86**, pp. 62–77
- Karvounarakis et al., 2003** Karvounarakis, Gregory; Magganaraki, Aimilia; Alexaki, Sofia; Christophides, Vassilis; Plexousakis, Dimitris; Scholl, Michel; Tolle, Karsten, 2003.
Querying the semantic web with RQL.
Computer networks, **42** (5), pp. 617–640
- Kennedy et al., 2001** Kennedy, Marc C.; O’Hagan, Anthony, 2001.
Bayesian calibration of computer models.
Journal of the Royal Statistical Society. Series B, Statistical Methodology, **63** (3), pp. 425–464

- Khatib, 2019** Khatib, Oussama, 2019.
The Age of human-robot collaboration.
In: Arakelian, Vigen; Wenger, Philippe (Hrsg.):
ROMANSY 22–Robot Design, Dynamics and Control.
Rennes, 25/06/2018–28/06/2018, p. 9.
ISBN 978-3-319-78963-7
- Kilger et al., 2012** Kilger, Wolfgang; Rampel, Jochen R.; Vikas, Kurt, 2012.
Flexible Plankostenrechnung und Deckungsbeitragsrechnung.
Wiesbaden : Springer Gabler.
ISBN 978-3-8349-3758-2
- Koltai et al., 2000** Koltai, Tams; Lozano, Sebastian; Guerrero, Fernando; Onieva, Luis, 2000.
A flexible costing system for flexible manufacturing systems using activity based costing.
International Journal of Production Research,
38 (7), pp. 1615–1630
- Koren et al., 1999** Koren, Yoram; Heisel, Uwe; Jovane, Francesco; Moriwaki, Toshimichi; Pritschow, Gumter; Ulsoy, Galip; Van Brussel, Hendrik, 1999.
Reconfigurable manufacturing systems.
CIRP Annals-Manufacturing Technology, **48**
(2), pp. 527–540
- Kroese et al., 2014** Kroese, Dirk P.; Brereton, Tim; Taimre, Thomas; Botev, Zdravko I., 2014.
Why the Monte Carlo method is so important today.
Wiley Interdisciplinary Reviews: Computational Statistics, **6** (6), pp. 386–392

- Krüger et al., 2009** Krüger, Jörg; Lien, Terje K.; Verl, Alexander, 2009.
Cooperation of human and machines in assembly lines.
CIRP Annals-Manufacturing Technology, **58** (2), pp. 628–646
- Krumwiede, 2005** Krumwiede, Kip R., 2005.
Rewards and realities of german cost accounting.
Strategic finance, pp. 27–34
- Kulatilaka, 1988** Kulatilaka, Nalin, 1988.
Valuing the flexibility of flexible manufacturing systems.
IEEE Transactions on engineering management, **35** (4), pp. 250–257
- Kuss et al., 2016** Kuss, Alexander; Diaz Posada, Julian Ricardo; Hollmann, Rebecca; Dietz, Thomas; Hägele, Martin, 2016.
Manufacturing Knowledge for Industrial Robot Systems: Review and Synthesis of Model Architecture.
In: *2016 IEEE International Conference on Automation Science and Engineering (CASE)*. Fort Worth, Texas, 21/08/2016–24/08/2016, pp. 348–354
- Kuss et al., 2017** Kuss, Alexander; Dietz, Thomas; Ksensow, Konstatin; Verl, Alexander, 2017.
Manufacturing Task Description for Robotic Welding and Automatic Feature Recognition on Product CAD Models.
Procedia CIRP, **60**, pp. 122–127

- Kusumoto et al., 2004** Kusumoto, Shinji; Matukawa, Fumikazu; Inoue, Katsuro; Hanabusa, Shigeo; Maegawa, Yuusuke, 2004.
Estimating effort by use case points: method, tool and case study.
In: *Proceeding of the 10th International Symposium on Software Metrics (METRICS), 2004*.
Chicago, 11/09/2004–17/09/2004, pp. 292–299
- La Rocca, 2012** La Rocca, Gianfranco, 2012.
Knowledge based engineering: Between {AI} and CAD. Review of a language based technology to support engineering design.
Advanced Engineering Informatics, **26** (2), pp. 159–179
- Lauven et al., 2010** Lauven, Lars; Wiedenmann, Susanne; Geldermann, Jutta, 2010.
Lebenszykluskosten als Entscheidungshilfe beim Erwerb von Werkzeugmaschinen.
Schwerpunkt Unternehmensführung, Georg-August-Universität Göttingen, **10**
- Lawson, 1988** Lawson, Tony, 1988.
Probability and Uncertainty in Economic Analysis.
Journal of Post Keynesian Economics, **11** (1), pp. 38–65
- Layer et al., 2002** Layer, Alexander; Brinke, Erik Ten; Houten, Fred Van; Kals, Hubert; Haasis, Siegmar, 2002.
Recent and future trends in cost estimation.
International Journal of Computer Integrated Manufacturing, **15** (6), pp. 499–510

-
- Lerch et al., 2010** Lerch, Christian; Weissfloch, Ute; Kinkel, Stefan, 2010.
Surplus of service-based business models – the integration of multiple perspectives for assessing win-win potentials.
International Journal of Services Operations and Informatics, **5** (4), pp. 400–417
- Leung et al., 2002** Leung, Hareton; Fan, Zhang, 2002.
Software cost estimation.
In: Chang, Shi-Kuo (Hrsg.): *Handbook of Software Engineering and Knowledge Engineering: Volume II: Emerging Technologies*,
Pp. 307–324.
ISBN 978-981-02-4974-8
- Lim et al., 2011** Lim, Gi Hyun; Suh, Il Hong; Suh, Hyowon, 2011.
Ontology-Based Unified Robot Knowledge for Service Robots in Indoor Environments.
IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, **41** (3), pp. 492–509
- Linsmeier et al., 2000** Linsmeier, Thomas J.; Pearson, Neil D., 2000.
Value at risk.
Financial Analysts Journal, **56** (2), pp. 47–67
- Litzenberger, 2015** Litzenberger, Gudrun (ed.), 2015.
World Robotics Industrial Robots 2014: Statistics, Market Analysis, Forecasts and Case Studies
Visited on: 12/02/2016.
Available from: <http://www.worldrobotics.org>

Lohse, 2006

Lohse, Niels, 2006.

Towards an ontology framework for the integrated design of modular assembly systems

Nottingham, University of Nottingham, Diss. 2006

Lotter et al., 2006

Lotter, Bruno; Wiendahl, Hans-Peter, 2006.

Montage in der industriellen Produktion: Ein Handbuch für die Praxis.

Berlin and Heidelberg : Springer-Verlag.

VDI-Buch.

ISBN 978-3-540-36669-0

Luehrman, 1998

Luehrman, Timothy A., 1998.

Investment opportunities as real options: Getting started on the numbers.

Harvard business review, **76** (4), pp. 51–66

Mahnič et al., 2012

Mahnič, Viljan; Hovelja, Tomaž, 2012.

On using planning poker for estimating user stories.

Journal of Systems and Software, **85** (9), pp. 2086–2095

Mäkeläinen, 1999

Mäkeläinen, Esa, 1999.

Introduction to Economic Value Added http://www.evanomics.com/download/EVA_English.pdf

Visited on: 10/12/2017

-
- Mannuß et al., 2012** Mannuß, Oliver; Schloske, Alexander; Bauernhansl, Thomas, 2012.
Availability Focused Risk Analysis to Support Life Cycle Costing Prognosis.
In: Dornfeld, David A.; Linke, Barbara S. (Hrsg.): *Leveraging Technology for a Sustainable World*.
Berkeley, 23/05/2012–25/05/2012, pp. 497–502.
ISBN 978-3-642-29069-5
- Masmoudi et al., 2007** Masmoudi, Faouzi; Bouaziz, Zoubeir; Hachicha, Wafik, 2007.
Computer-aided cost estimation of weld operations.
International Journal of Advanced Manufacturing Technology, **33** (3-4), pp. 298–307
- Matthews et al., 1990** Matthews, John; Broadwater, Robert; Long, Leland, 1990.
The application of interval mathematics to utility economic analysis.
IEEE Transactions on Power Systems, **5** (1), pp. 177–181
- McNeil et al., 2005** McNeil, Alexander; Frey, Rüdiger; Embrechts, Paul, 2005.
Quantitative Risk Management: Concepts Techniques and Tools.
Princeton and Oxford : Princeton University Press.
ISBN 978-0-691-16627-8

- Meyer, 2011** Meyer, Christian, 2011.
Aufnahme und Nachbearbeitung von Bahnen bei der Programmierung durch Vormachen von Industrierobotern.
ISBN 978-3-939890-75-1
Stuttgart, Stuttgart University, Jost-Jetter, Heimsheim, Diss. 2011
- Moore et al., 2003** Moore, Ramon; Lodwick, Weldon, 2003.
Interval analysis and fuzzy set theory.
Fuzzy Sets and Systems, **135** (1), pp. 5–9
- Mustajoki et al., 2006** Mustajoki, Jyri; Hämäläinen, Raimo P.; Lindstedt, Mats R. K., 2006.
Using intervals for global sensitivity and worst-case analyses in multiattribute value trees.
European Journal of Operational Research, **174** (1), pp. 278–292
- Myers, 1977** Myers, Stewart C., 1977.
Determinants of corporate borrowing.
Journal of financial economics, **5** (2), pp. 147–175
- Neely et al., 1995** Neely, Andy; Gregory, Mike; Platts, Ken, 1995.
Performance measurement system design: A literature review and research agenda.
International Journal of Operations & Production Management, **15** (4), pp. 80–116
- Neo4j, 2015** Neo4j, 2015
<http://www.neo4j.com>
Visited on: 10/12/2017

- Newman, 2015** Newman, Andrew, 2015.
JRDF — An RDF Library in Java <http://jrdf.sourceforge.net/>
Visited on: 17/07/2017
- Niazi et al., 2005** Niazi, Adnan; Dai, Jian S.; Balabani, Stavroula; Seneviratne, Lakmal, 2005.
Product cost estimation: technique classification and methodology review.
Journal of Manufacturing Science and Engineering, **128** (2), pp. 563–575
- Nielsen, 2012** Nielsen, Kurt, 2012.
Strategic automation in European manufacturing industry: how to transform the European manufacturing industry into a competitive and sustainable local-based industry for the future — Danish experience.
In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2012*.
Vila Moura, Algarve, 07/10/2012–12/10/2012
- Nilsen et al., 2003** Nilsen, Thomas; Aven, Terje, 2003.
Models and model uncertainty in the context of risk analysis.
Reliability Engineering & System Safety, **79** (3), pp. 309–317
- Pahl et al., 2007** Pahl, Gerhard; Beitz, Wolfgang; Feldhusen, Joerg; Grote, Karl-Heinrich, 2007.
Engineering Design — A Systematic Approach.
London : Springer.
ISBN 978-1-84628-319-2

- PAIB, 2008** PAIB: Professional Accountants in Business Committee, 2008.
International Good Practice Guidance: Project Appraisal Using Discounted Cash Flow.
International Federation of Accountants IFAC.
ISBN 978-1-934779-39-2
- PAIB, 2009** PAIB: Professional Accountants in Business Committee, 2009.
International Good Practice Guidance: Evaluating and Improving Costing in Organizations.
International Federation of Accountants IFAC.
ISBN 978-1-60815-037-3
- Paryanto et al., 2015** Paryanto; Brossog, Matthias; Bornschlegl, Martin; Franke, Jörg, 2015.
Reducing the energy consumption of industrial robots in manufacturing systems.
International Journal of Advanced Manufacturing Technology, **78**, pp. 1215–1328
- Paul et al., 1979** Paul, Richard P.; Nof, Shimon Y., 1979.
Work methods measurement — a comparison between robot and human task performance.
International Journal of Production Research, **17** (3), pp. 277–303
- Perkins et al., 2011** Perkins, David; Stovall, O. Scott, 2011.
Resource Consumption Accounting Where Does It Fit?
Journal of Applied Business Research (JABR), **27** (5), pp. 41–52

- Persson et al., 2010** Persson, Jacob; Gallois, Axel; Bjoerkelund, Anders; Hafdell, Love; Haage, Mathias; Malec, Jacek; Nilsson, Klas; Nugues, Pierre, 2010. A Knowledge Integration Framework for Robotics.
In: *41st International Symposium on Robotics (ISR) and 6th German Conference on Robotics (ROBOTIK)*.
Munich, 07/06/2010–09/06/2010, pp. 1068–1075
- Pike, 1996** Pike, Richard, 1996.
A Longitudinal Survey on Capital Budgeting Practices.
Journal of Business Finance & Accounting, **23** (1), pp. 79–92
- Polejewski, 2013** Polejewski, Shirley A., 2013a.
German Cost-Accounting vs. Activity-based Costing <http://blog.cengage.com/wp-content/uploads/2014/07/FALL-2009-Polejewski.German-Cost-Accounting.pdf>
Visited on: 10/12/2017
- Polejewski, 2013** Polejewski, Shirley A., 2013b.
Resource Consumption Accounting <http://blog.cengage.com/wp-content/uploads/2014/07/SUMMER-2009-Polejewski.Resource-Consumption-Acct.pdf>
Visited on: 10/12/2017

- Postma et al., 2005** Postma, Theo J. B. M.; Liebl, Franz, 2005.
How to improve scenario analysis as a strategic management tool?
Technological Forecasting and Social Change, **72** (2), pp. 161–173
- Pott, 2007** Pott, Andreas, 2007.
Analyse und Synthese von Werkzeugmaschinen mit paralleler Kinematik.
ISBN 978-3-18-340920-4
Düsseldorf, Universität Duisburg-Essen, Diss. 2007
- Putnam, 1978** Putnam, Lawrence H., 1978a.
A General Empirical Solution to the Macro Software Sizing and Estimating Problem.
IEEE Transactions on Software Engineering, **SE-4** (4), pp. 345–361
- Putnam, 1978** Putnam, Lawrence H., 1978b.
A general empirical solution to the macro software sizing and estimating problem.
IEEE transactions on Software Engineering, (4), pp. 345–361
- Quintana et al., 2011** Quintana, Guillem; Ciurana, Joaquim, 2011.
Cost estimation support tool for vertical high speed machines based on product characteristics and productivity requirements.
International Journal of Production Economics, **134** (1), pp. 188–195

-
- Ramasesh et al., 1997** Ramasesh, Ranga V.; Jayakumar, Maliyakal D., 1997.
Inclusion of flexibility benefits in discounted cash flow analyses for investment evaluation: A simulation/optimization model.
European Journal of Operational Research, **102** (1), pp. 124–141
- Rezaie et al., 2007** Rezaie, Kamran; Ostadi, Bakhtiar, 2007.
A mathematical model for optimal and phased implementation of flexible manufacturing systems.
Applied mathematics and computation, **184** (2), pp. 729–736
- Ringland, 1998** Ringland, Gill, 1998.
Scenario planning: Managing for the future.
Chicester : John Wiley & Sons.
ISBN 978-0-470-01881-1
- Roy et al., 2001** Roy, Rajkumar; Kelvesjo, Sara; Forsberg, Sara; Rush, Chris, 2001.
Quantitative and qualitative cost estimating for engineering design.
Journal of Engineering Design, **12** (2), pp. 147–162

- Runde et al., 2009** Runde, Stefan; Güttel, Knut; Fay, Alexander, 2009.
Transformation von CAEX-Anlagenplanungsdaten in OWL — Eine Anwendung von Technologien des Semantic Web in der Automatisierungstechnik.
In: Verein Deutscher Ingenieure (VDI) (Hrsg.): *Tagungsband zum Kongreß Automation 2009, VDI-Berichte*.
Baden-Baden, 16/06/2009–17/06/2009, pp. 175–178.
ISBN 978-3-180-92067-2
- Ryan et al., 2002** Ryan, Patricia A.; Ryan, Glenn P., 2002.
Capital Budgeting Practices of the Fortune 1000: How Have Things Changed?
Journal of Business and Management, **8** (4), pp. 355–364
- Saltelli et al., 2008** Saltelli, Andrea; Ratto, Marco; Andres, Terry; Campolongo, Francesca; Cariboni, Jessica; Gattelli, Debora; Saisana, Michaela; Tarantola, Stefano, 2008.
Global Sensitivity Analysis: The Primer.
Chiccester : Wiley.
ISBN 978-0-470-05997-5
- Saltelli et al., 2004** Saltelli, Andrea; Tarantola, Stefano; Campolongo, Francesca; Ratto, Marco, 2004.
Sensitivity analysis in practice: a guide to assessing scientific models.
Chiccester : John Wiley & Sons.
ISBN 978-0-470-87093-8

- Schleipen et al., 2009** Schleipen, Miriam; Draht, Rainer, 2009.
Three-view-concept for modeling process or manufacturing plants with AutomationML.
In: *IEEE Conference on Emerging Technologies Factory Automation (ETFA), 2009*.
Palma de Mallorca, 22/09/2009–25/09/2009
- Schlenoff et al., 2012** Schlenoff, Craig; Prestes, Edson; Madhavan, Raj; Goncalves, Paulo; Li, Howard; Balakirsky, Stephen.; Kramer, Thomas; Migueláñez, Emilio, 2012.
An IEEE standard Ontology for Robotics and Automation.
In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2012*.
Vila Moura, Algarve, 07/10/2012–12/10/2012, pp. 1337–1342
- Schneider et al., 2001** Schneider, Geri; Winters, Jason P., 2001.
Applying use cases: a practical guide.
Boston : Addison-Wesley.
ISBN 978-0-201-70853-0
- Sesame, 2013** Sesame, OpenRDF, 2013
<http://www.openrdf.org>
Visited on: 10/06/2016
- Shadbolt et al., 2006** Shadbolt, Nigel; Hall, Wendy; Berners-Lee, Tim, 2006.
The Semantic Web Revisited.
IEEE Intelligent Systems, **21** (3), pp. 96–101
- Sharman et al., 2004** Sharman, Paul A.; Vikas, Kurt, 2004.
Lessons from German Cost Accounting.
Strategic finance, pp. 28–34

- Sharpe, 1963** Sharpe, William F., 1963.
A simplified model for portfolio analysis.
Management science, **9** (2), pp. 277–293
- Smets et al., 1998** Smets, Philippe; Gabbay, Dov M., 1998.
Handbook of defeasible reasoning and uncertainty management systems. Vol. 1: Quantified representation of uncertainty and imprecision.
Dordrecht : Kluwer Academic Publishers.
ISBN 978-0-7923-5100-9
- Smith, 1994** Smith, David J., 1994.
Incorporating risk into capital budgeting decisions using simulation.
Management decision, **32** (9), pp. 20–26
- Staab et al., 2009** Staab, Steffen; Studer, Rudi, 2009.
Handbook on Ontologies.
Dordrecht : Springer.
ISBN 978-3-540-92673-3
- Stahel, 2002** Stahel, Werner A., 2002.
Statistische Datenanalyse — Eine Einführung für Naturwissenschaftler.
Braunschweig : Vieweg & Sohn.
ISBN 978-3-8348-0410-5
- Staubus, 1990** Staubus, George J., 1990.
Activity costing. Twenty Years On.
Management Accounting Research, **1**, pp. 249–264
- Stenmark, 2017** Stenmark, Maj, 2017.
Intuitive instruction of industrial robots: a knowledge-based approach
Lund, Lund University, Diss. 2017

-
- Stenmark et al., 2015** Stenmark, Maj; Malec, Jacek; Nilsson, Klas; Robertsson, Anders, 2015.
On Distributed Knowledge Bases for Robotized Small-Batch Assembly.
IEEE Transactions on Automation Science and Engineering, **12** (2), pp. 519–528
- Stenmark et al., 2013** Stenmark, Maj; Malec, Jacek; Robertsson, Anders, 2013.
On Distributed Knowledge Basis for Industrial Robotics Needs.
In: *Cloud Robotics Workshop at IROS 2013*
- Swart et al., 2004** Swart, Rob J.; Raskin, Paul; Robinson, John, 2004.
The problem of the future: sustainability science and scenario analysis.
Global environmental change, **14** (2), pp. 137–146
- Tausworthe, 1979** Tausworthe, Robert C., 1979.
The work breakdown structure in software project management.
Journal of Systems and Software, **1** (1), pp. 181–186
- Teittinen et al., 2013** Teittinen, Henri; Pellinen, Jukka; Järvenpää, Marko, 2013.
ERP in action — Challenges and benefits for management control in SME context.
International Journal of Accounting Information Systems, **14** (4), pp. 278–296

- Tenorth et al., 2013** Tenorth, Moritz; Beetz, Michael, 2013.
KnowRob: A knowledge processing infrastructure for cognition-enabled robots.
The International Journal of Robotics Research, **32** (5), pp. 566–590
- Thompson, 1990** Thompson, Paul B., 1990.
Risk objectivism and risk subjectivism: When are risks real.
Risk: Health, Safety & Environment, **1** (1), pp. 3–22
- Thrun, 2002** Thrun, Sebastian, 2002.
Probabilistic robotics.
Communications of the ACM, **45** (3), pp. 52–57
- Tofallis, 2008** Tofallis, Chris, 2008.
Investment volatility: A critique of standard beta estimation and a simple way forward.
European Journal of Operational Research, **187** (3), pp. 1358–1367
- Tsao, 2012** Tsao, Chung-Tsen, 2012.
Fuzzy net present values for capital investments in an uncertain environment.
Computers & Operations Research, **39** (8), pp. 1885–1892.
Special issue: Advances of Operations Research in Service Industry

-
- Tseng et al., 2000** Tseng, Y. J.; Jiang, B. C., 2000.
Evaluating Multiple Feature-Based Machining Methods Using an Activity-Based Cost Analysis Model.
The International Journal of Advanced Manufacturing Technology, **16** (9), pp. 617–623
- Usman et al., 2014** Usman, Muhammad; Mendes, Emilia; Weidt, Francila; Britto, Ricardo, 2014.
Effort estimation in agile software development: a systematic literature review.
In: *Proceedings of the 10th International Conference on Predictive Models in Software Engineering (Promise), 2014*.
Torino, 17/09/2014, pp. 82–91
- Van der Merwe et al., 2002** Van der Merwe, Anton; Keys, David E., 2002.
The Case For Resource Consumption Accounting.
Strategic Finance, **83** (10), pp. 31–36
- VDI 2221, 1993** VDI 2221:1993.
Systematic approach to the development and design of technical systems and products
- VDMA 34160-06, 2006** VDMA 34160-06:2006.
Forecasting Model for Lifecycle Costs of Machines and Plants
- Von Reibnitz, 1988** Von Reibnitz, Ute, 1988.
Scenario techniques.
Nürnberg : McGraw-Hill.
ISBN 978-0-070-65931-5

- Wierda, 1991** Wierda, Leo S., 1991.
Linking Design, Process Planning and Cost Information by Feature-based Modelling.
Journal of Engineering Design, **2** (1), pp. 3–19
- W3C, 2004** World Wide Web Consortium (W3C), 2004.
RDF/XML syntax specification (revised) <http://www.w3.org/TR/rdf-syntax-grammar/>
Visited on: 10/12/2017
- W3C, 2008** World Wide Web Consortium (W3C), 2008.
SPARQL Query Language for RDF <http://www.w3.org/TR/rdf-sparql-query/>
Visited on: 10/06/2016
- W3C, 2012** World Wide Web Consortium (W3C), 2012.
OWL 2 Web Ontology Language <http://www.w3.org/TR/owl2-overview/>
Visited on: 10/12/2017
- W3C, 2015** World Wide Web Consortium (W3C), 2015.
RDF Schema 1.1 <http://www.w3.org/TR/rdf-schema/>
Visited on: 10/12/2017
- Wygant et al., 1987** Wygant, Robert M.; Donaghey, Charles E., 1987.
Ergonomic Considerations in Robot Selection and Safety.
Proceedings of the Human Factors and Ergonomics Society Annual Meeting, **31** (2), pp. 181–185

Xu et al., 2006

Xu, Dong-Ling; Yang, Jian-Bo; Wang, Ying-Ming, 2006.

The evidential reasoning approach for multi-attribute decision analysis under interval uncertainty.

European Journal of Operational Research, **174** (3), pp. 1914–1943

Zachariasen et al., 2011

Zachariasen, Frederik; Arlbjørn, Jan Stentoft, 2011.

Exploring a differentiated approach to total cost of ownership.

Industrial Management & Data Systems, **111** (3), pp. 448–469

Zadeh, 1978

Zadeh, Lotfi Asker, 1978.

Fuzzy sets as a basis for a theory of possibility.

Fuzzy sets and systems, **1** (1), pp. 3–28

Robot systems promise high potential for improvement of production processes in small and medium-sized enterprises (SME). Often, reliable data for the assessment of costs and benefits of robot systems is missing, because robot systems are special machinery and because the required information is distributed among component manufacturer, system integrator and end-user. This thesis addresses the problem of cost-benefit assessment of industrial robot systems in the face of uncertain and incomplete information by combining methods from business economics and from knowledge management in robotics in a cost-benefit model. The resulting method allows efficient decision support in the planning and realization of SME-type robot systems. A test implementation is used to evaluate the method for two realistic use cases.

ISBN 978-3-8396-1437-2



FRAUNHOFER VERLAG